Improving Generalization in Imitation Learning with Adversarial Learning

Nguyen Duc Tho

Supervisor: Prof. Eiji Kamioka

Functional Control Systems
Shibaura Institute of Technology

This dissertation is submitted for the degree of
Doctor of Philosophy

September 2023

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university.

Nguyen Duc Tho
September 2023

Acknowledgements

I would like to express deep and sincere gratitude to my supervisor, Prof. Kamioka Eiji, for his patience and motivation, for the continuous support and indispensable guidance which led me to the completion of this dissertation.

I would also like to thank Dr. Phan Xuan Tan, for inspiring and guiding me throughout my studies. It would have been more difficult without his invaluable advice and suggestions.

I want to thank my family for their love and hard work in making it possible for me to complete this dissertation. I especially want to thank my parent for being there for me always. To my little brother, you have been my main inspiration all these years.

Last but not least, I would be remiss in not mentioning Kem. Without her understanding and encouragement, it would be impossible for me to complete my study. I also have been very thankful to my friend, Tran Minh Chanh, for his support in brainstorming new ideas and encouragement to complete this dissertation.

Abstract

The demand for autonomous agents capable of mimicking human behaviors has grown significantly in recent years. For example, self-driving vehicles, assistive robots, and human-computer interaction fields rely on the ability of agents that can not only make optimal decisions but also behave like humans, which can enable the agents' actions to be believable and appear natural. In order for autonomous agents to acquire such human complex behaviors, imitation learning, also known as learning from demonstration, has been widely used for training autonomous agents in complex environments. Unfortunately, traditional imitation learning algorithms only focus on training an agent to perform a single task well in a given domain, which limits the ability of these agents to generalize on future unseen situations. Thus, they are still far from being comparable with humans due to the lack of the following abilities:

- Humans can recognize structural differences among the task's environments in order to adapt their behaviors accordingly. On the other hand, the performance of the learned agents tends to drop significantly even with a slight amount of distribution shift between training and testing environments (i.e., domain shift).
- Humans do not learn a new task from scratch. Instead, humans often leverage the knowledge learned from previous tasks to build up the necessary skills for performing the new task.

The problem is formalized as domain and task adaptations in imitation learning, which is the main focus of this dissertation. This thesis advances generalization in imitation learning by presenting three novel agents.

- First, to overcome the domain shift problem, the DAIL–GAN agent is proposed. The agent is trained using adversarial learning under a specific objective func-

tion in order to learn both domain-shared and domain-specific features. These features enable the learned agent to perform the task effectively across different domains.

- Second, the TAIL–GAN agent is presented in order to address the task adaptation challenge in imitation learning by utilizing a similar adversarial learning process. Experimental results show that the agent can leverage the knowledge learned from a source task to accelerate the learning process of a new target task.
- Third, to further extend the potential of adversarial learning in tackling the generalization challenge in imitation learning, the DTAIL–GAN agent is introduced as a step toward a more domain- and task- generalizable. The training and adaptation processes leverage the idea of repetition learning in neuroscience in order to help the agent overcome the catastrophic forgetting problem and adapt their previously learned source task's knowledge to a new target task. Moreover, experimental results demonstrate the learned agent can provide a better generalization and consistently perform well on both source and target tasks across different domains.

Table of contents

List of figures

List of tables

List of Abbreviations and Symbols

Symbols

$\Pr(X = x)$	probability that a random variable X takes on the value x
$X \sim p$	random variable X selected from distribution $p(x) = \Pr(X = x)$
$\mathbb{E}[X]$	expectation of a random variable X
e^x	the base of the natural logarithm, $e \approx 2.71828$, carried to power x
$\mathbb{R}$	set of real numbers
$\mathbb{1}(a)$	indicator function ($\mathbb{1}(a) = 1$ if a is true, else 0)
s, s'	states
a	an action
r	a reward
$\mathcal{S}$	set of all states
$\mathcal{A}(s)$	set of all actions available in state s
γ	discount factor
H	final time step of an episode, or the length of an episode
t	discrete time step
s^t	state at time t
a^t	action at time t

r^t	reward at time t

Acronyms / Abbreviations

BC	Behavior cloning
IL	Imitation learning
IRL	Inverse reinforcement learning
RL	Reinforcement learning

Chapter 1

Introduction

1.1 Motivation

Robots play an essential role in humans' lives as they can enhance human productivity and quality of life in everyday activities. Moreover, it is used in many fields, from industry, agriculture, and transportation to healthcare and surgery. For instance, during the COVID-19 pandemic, due to the difficulties brought by the highly contagious SARS-CoV-2 virus and its associated lockdowns, robots have been widely applied in almost every aspect to manage the crisis and overcome the challenges of the pandemic, such as disinfecting hospital rooms, handling infectious materials, delivery to quarantined, etc. [1]–[5]

However, before robots can be considerably helpful in practical day-to-day tasks in human-centered spaces — spaces specifically designed for humans, not machines — they need to be able to provide assistance to humans effectively. In order to allow robots to be more helpful, there are three main challenges that need to be considered.

1. Enabling robots to understand and apply their learned skills in order to solve a given task.
2. Allowing robots and human interaction to be more efficient and natural.
3. Generalizing the number of low-level robots' skills, like manipulation, to effectively perform other tasks in unseen situations.

The first challenge can be tackled using *reinforcement learning* (RL). RL is an effective method for solving sequential decision-making tasks by training an agent. During the learning process, the RL agent interacts with the environment through trial and error to collect experiences and improve its performance in order to solve a given task [6]. For example, to teach a robot to walk using RL, the agent initially may decide to take a big step forward, then fall. The next time the agent takes a minor step and is able to maintain its balance. The RL agent improves its walking performance by trying variations like that a number of times. Finally, it can succeed in learning to walk steadily. In addition, the example shows that the agent learns how to walk based on its received feedback after taking action. The feedback can be a reward (maintaining balance) or a penalty (falling). In RL, this feedback is called a reward function. For each task that the agent has to accomplish, a carefully designed reward function must be provided.

However, designing a hand-crafted reward function may require too much time or expense, especially in complex tasks. Moreover, in order to address the second challenge, which is improving human-robot interaction, the RL agent has to perform naturally and provide human-like actions. Including these nonlinear and dynamic constraints in the reward function makes the challenge more difficult. Therefore, this problem has motivated a number of research studies on *imitation learning* (IL), where expert-generated demonstration data are provided instead of a reward function in order to help the agent learn how to perform a task.

In imitation learning, an expert (commonly a human) provides a set of demonstrations on a given task. The IL agent then tries to learn by following and imitating the expert's behaviors. IL is beneficial when it is easier for an expert to demonstrate the desired behavior than to specify a reward function. Moreover, since the agent is trained to imitate the expert's behaviors, it can behave like humans, which allows the agent's actions to be believable and appear natural. This characteristic of IL agents can significantly enhance human-robot interaction.

Regardless, traditional IL agents are not able to overcome the third challenge, which is generalizing their learned behaviors to tackle new situations. The problem is referred to as *generalization* in imitation learning and can be divided into two categories.

Domain adaptation IL agents often overfit to the expert demonstrations by learning task-irrelevant features, thus suffering from generalization to different structures in the task's environment (e.g., different object positions, size, etc.) from the ones seen in the demonstrations. These environment variations are called domain shifts or distribution shifts. Humans can recognize structural differences among the task's environments in order to adapt their behaviors accordingly. On the other hand, IL agents deterministically copying the expert's behaviors tend to fail miserably, even with a slight amount of domain shift between the expert and agent environments.

Task adaptation IL agents are designed to focus on accomplishing only a single, narrowly defined task. Therefore, when given a new task, the agent has to start the learning process again from the ground up, even if it has already learned a task that is related to and shares the same structure with the new one. This learning process can be redundant and expensive. On the contrary, humans do not learn a new task from scratch. Instead, humans often leverage the knowledge learned from previous tasks to solve such related tasks in a short time.

The need for generalizable IL agents is ubiquitous, given the dynamic of the real-world environment. Still, generalization remains a fundamental challenge for modern imitation learning. Therefore, **the primary goal of this dissertation is to improve the generalization of IL agents by tackling the domain and task adaptation problems**. This dissertation addresses the problem by utilizing adversarial learning to design objective functions for the agent's learning process. Adversarial learning allows the agent interacts with the environment and discriminates the expert's behaviors from its own interactions. Leveraging this unique learning method, the agent can effectively extract the underlying structure of the task from the expert demonstrations instead of only directly imitating the expert's behaviors. These extracted features later help the agent adapt its previously learned knowledge to a new domain or new task effectively.

1.2 Research Contributions

Based on the primary goal, the work present in the dissertation resulted in three novel imitation learning agents. These agents are summarized as follows:

- The DAIL–GAN agent is presented to address the domain adaptation in imitation learning. In order to overcome the domain shift problem, an objective function is designed based on adversarial learning in order to allow the agent to extract domain-shared and domain-specific features. These features are utilized to improve the agent's performance across different domains.
- The task adaptation problem is then considered by introducing TAIL–GAN agent. The agent is trained using a similar objective function as DAIL–GAN. The experiment results demonstrate the agent's effectiveness in adapting the learned knowledge from a source task to a new target task.
- The result of DAIL–GAN and TAIL–GAN highlights the potential of adversarial learning in improving generalization in imitation learning. Thus, to further extend its capability, the DTAIL–GAN agent and an adaptation method are proposed to tackle both domain and task adaptation problems in imitation learning. Being inspired by the idea of repetition learning in neuroscience, the proposed adaptation method enables the DTAIL–GAN agent to repeatedly review the learned knowledge of the source task while learning the new knowledge of the target task. This ensures that the learning performance on the target task is high while the deterioration of the learning performance on the source task is small. A comprehensive evaluation over several simulated tasks with varying difficulty levels shows that the proposed method can provide high and consistent performance on both source and target tasks across different domains.

1.3 Dissertation Organization

The remainder of this dissertation is structured as follows:

- Chapter 2 provides essential background information that is relevant to all subsequent chapters and a literature review of the methods used for domain and task adaptations in imitation learning.
- Chapters 3, 4, and 5 introduces three novel imitation learning agents. The work presented in these chapters has been published in [7] and [8].
- Chapter 6 discusses the potential of applying adversarial learning to improve generalization in imitation learning and the feasibility of utilizing the proposed agents in real-world tasks.
- Chapter 7 concludes the dissertation and presents potential future research work.

Chapter 2

Background and Literature Review

This chapter introduces the necessary background information and discussion of related work relevant to subsequent chapters of this dissertation.

2.1 Reinforcement Learning

Reinforcement learning (RL) is an approach to solving sequential decision-making tasks in complex and stochastic environments. In RL, the learner or the decision-maker is called the *agent*. The thing it interacts with, including everything outside the agent, is called the *environment* [6]. A key aspect of RL is that it tries to mimic how humans learn new things from interaction with the environment. In other words, the RL agent has to actively interact with the environment through *trial and error* in order to gather experiences and find a sequence of actions that can solve a given task. RL is one of three basic machine learning paradigms, as depicted in Figure 2.1.

2.1.1 Episodic Finite-horizon Markov Decision Processes

Consider a sequential decision-making problem in which an agent is faced with a task of influencing an environment through the actions it takes. At each time step, the agent observes the environment and must decide on which action to perform. The agent affects the environment through these actions. This action alters the environment and determines the immediate reward the agent receives. The interaction

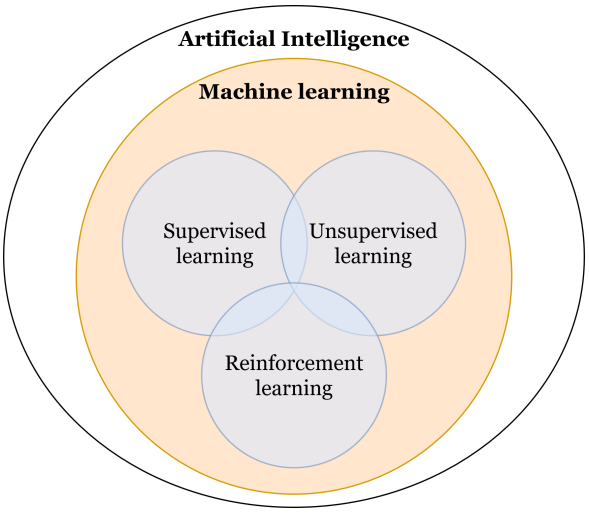


Figure 2.1. Reinforcement learning is a machine learning approach to artificial intelligence.

between the agent and its environment is formalized as a *Markov decision process* (MDP) [9]. This dissertation focuses on the episodic finite-horizon MDP, defined as follows.

$$\mathbb{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma, H)$$

- The state space $\mathcal{S}$ is a finite set of states that the environment can have.
- The action space $\mathcal{A}$ is a finite set of actions the agent is allowed to take after observing a state.
- The transition function $P(\cdot|s, a)$ gives the probability of transition $P(s'|s, a) = \Pr(s^{t+1} = s'|s^t = s, a^t = a)$ at time step t from state s to state s' under action a.
- The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ gives the immediate reward $R(s, a)$ for taking an action a after observing state s.
- $\gamma \in [0, 1)$ is the discount factor.
- The horizon $H \in \mathbb{N}$ is the number of time steps in each episode.

The decision rules that determine the agent's action for state s is a policy, denoted by $\pi(a|s)$. In other words, the policy $\pi(a|s)$ determines the agent's behaviors.

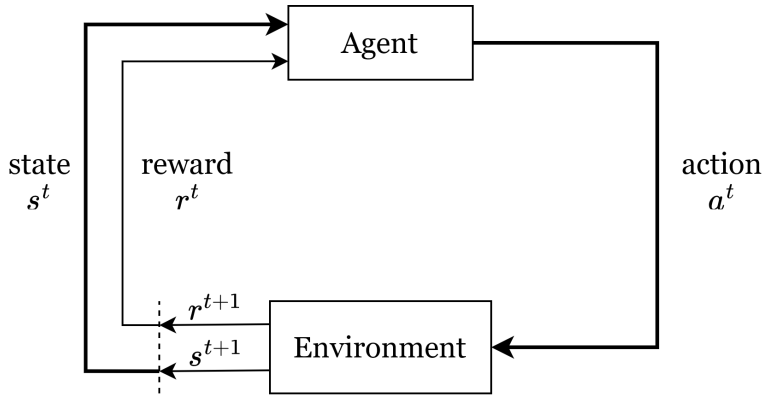


Figure 2.2. The agent-environment interaction in Markov decision process.

Figure 2.2 illustrated the interaction between the agent and its environment. At each time step t, the agent observes state s^t and takes an action a^t according to its policy $\pi(a|s)$. The environment transitions to s^{t+1} according to the transition distribution $P(\cdot|s^t, a^t)$. The agent then received a reward $r^t = R(s^t, a^t)$ and observes the state s^{t+1} of the next time step $t + 1$. This interaction loop between the agent and environment continues for a total of H time steps and generates a trajectory $\tau = (s^0, a^0, s^1, a^1, \ldots, s^H, a^H)$

2.1.2 Reinforcement Learning in Episodic Fixed-horizon MDPs

In an episodic fixed-horizon MDP setting, an RL agent is assumed to

- know the state space $\mathcal{S}$ and action space $\mathcal{A}$
- know the immediate reward $R(s, a)$ when taking action a after observing state s
- does not know the transition function $P(\cdot|s, a)$

The agent can only learn about $P(\cdot|s, a)$ through interaction with the environment. Given an MDP process, the agent's goal is to find an optimal policy $\pi^*(a|s)$ that maximizes the expected discounted sum of rewards along trajectories $J(\pi)$.

$$J(\pi) = \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t R(s^t, a^t)] = \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t r^t]$$

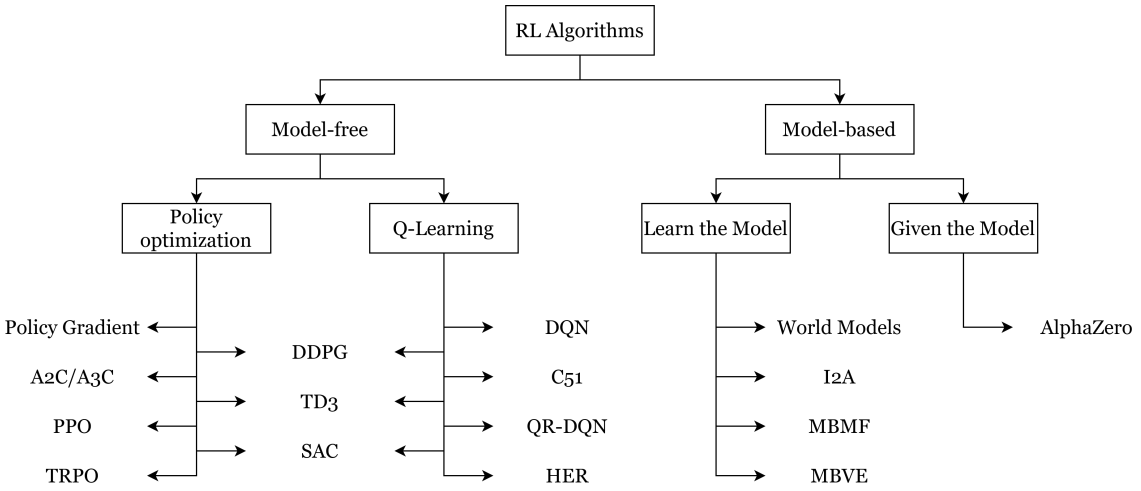


Figure 2.3. Several examples of reinforcement learning algorithms.

That is, $\pi^*(a|s) = \text{argmax}_\pi J(\pi)$. The discount factor γ values the immediate reward above delayed rewards. The following subsection briefly reviews several approaches to finding $\pi^*(a|s)$ in RL.

2.1.3 Reinforcement Learning Algorithms

An overview of RL algorithms is illustrated in Figure 2.3. RL algorithms can be divided into two main categories:

Model-based algorithms Model-based algorithms aim to find the environment dynamics model. In other words, it tries to establish a complete MDP by estimating the transition and reward functions. When the dynamics model is available, the problem of finding an optimal policy is called *planning*. Several model-based algorithms include *World models* [10], *Imagination-Augmented Agents* (I2A) [11], *Model-Based Priors for Model-Free Reinforcement Learning* (MBMF) [12], *Model-Based Value Expansion* (MBVE) [13], and *AlphaZero* [14].

Model-free algorithms Model-free algorithms are data-driven and rely on trial-and-error experiences to find the optimal policy. Due to the difficulties of establishing a complete MDP, model-free algorithms have been the main focus of research compared to model-based algorithms. Several model-free algorithms include *Policy Gradient*, *Asynchronous Advantage Actor-Critic* (A3C) [15], *Proximal Policy*

Optimization (PPO) [16], *Trust Region Policy Optimization* (TRPO) [17], *Deep Deterministic Policy Gradients* (DDPG) [18], *Soft Actor-Critic* (SAC) [19], *Twin Delayed Deep Deterministic Policy Gradients* (TD3) [20], *Deep Q Neural Network* (DQN) [21], *C51* [22], *Distributional Reinforcement Learning with Quantile Regression* (QR-DQN) [23], and *Hindsight Experience Replay* (HER) [24].

There are two main approaches to training an RL agent with model-free algorithms:

Policy optimization Policy optimization seeks to directly learn $\pi^*(a|s)$ by applying gradient ascent on the objective $J(\pi)$.

Q-learning Q-learning methods aim to discover the optimal policy by finding a policy that maximizes the action-value function $q_\pi(s, a)$. The $q_\pi(s, a)$ is defined as the expectation of cumulative discounted rewards.

$$q_\pi(s, a) = \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t r^t | s^t = s, a^t = a]$$

The optimal action-value function $q_{\pi^*}(s, a)$ is the maximum action-value function over all policies. It indicates the maximum possible reward the agent can extract from the environment starting at state s and taking action a.

$$q_{\pi^*}(s, a) = \max_\pi q_\pi(s, a)$$

In case $q_{\pi^*}(s, a)$ is known, the agent can decide which action to take to maximize the return. Thus, it can behave optimally in the MDP and therefore solve the MDP task.

Policy optimization methods are more stable than Q-learning methods since they directly optimize the policy using numerical optimization techniques such as gradient ascent. In contrast, Q-learning methods indirectly find the optimal policy by optimizing $q_\pi(s, a)$. However, Q-learning has the advantage of being more data efficient since it can reuse data more effectively than policy optimization. Fortunately, many algorithms have been designed utilizing both methods to carefully trade-off between the strengths and weaknesses of either policy optimization or Q-learning methods, such as DDPG, TD3, and SAC.

2.2 Imitation Learning

RL has been primarily limited to tasks in which a well-defined reward function is given. However, the assumption of having access to a reward function is not feasible in complex tasks or the real-world environment. For example, consider the task of learning a good policy for autonomous driving. While a human driver is able to drive safely on roads, he/she may not be able to mathematically formulate a reward function that accurately represents good driving behaviors. Moreover, it may be impossible to design such a reward function given the dynamics of the environment (e.g., stoplights, safety signs, traffic jams, pedestrians, etc.) Without a good reward function, RL is not practical for the autonomous driving problem.

Fortunately, a good policy can still be learned by directly imitating demonstrations provided by an expert. This approach to learning a policy by mimicking an expert's behaviors to accomplish a task is called imitation learning. In IL, the expert can be referred to as the teacher, while the agent is the learner. IL does not require the explicit design of a reward function. Moreover, since the expert demonstrations directly provide rich information regarding how to perform the task optimally, IL typically requires fewer interactions with the environment than RL.

In imitation learning setting, a task can be formulated as an MDP without a reward function, i.e., $\mathbb{M}^- = \mathbb{M} \setminus R = (\mathcal{S}, \mathcal{A}, P, \gamma, H)$ Thus, in an episodic fixed-horizon MDP setting, an IL agent is assumed to

- know the state space $\mathcal{S}$, action space $\mathcal{A}$
- does not know the immediate reward $R(s, a)$ when taking action a after observing a state s
- does not know the transition distribution $P(\cdot|s, a)$ and the reward function R

Formally, the IL agent is provided with a set of expert demonstration $\mathcal{D}_\mathcal{E} = \{\tau_\mathcal{E}^i : i \in [1, N]\}$ that is collected by letting the expert interact in the task MDP $\mathbb{M}^-$. Each $\tau_\mathcal{E}^i = \{(s^t, a^t) : t \in [0, H]\} = (s^0, a^0, \ldots, s^H, a^H)$ denotes a single demonstration or an episode, which is a sequence of state-action pairs. The goal of the IL agent is to learn an optimal policy $\pi^*(a|s)$ against unknown reward function R, given the expert demonstrations $\mathcal{D}_\mathcal{E}$. Since the agent does not have access to immediate rewards

during the learning process, IL is considered to be more challenging compared to reinforcement learning.

There are two approaches to training an IL agent.

Direct This approach utilizes supervised learning to learn a policy from expert demonstrations. An example of this approach is *Behavior cloning* (BC).

Indirect This approach learns the unknown reward function using expert demonstrations and derives an optimal policy from it. An example of this approach is *Inverse reinforcement learning* (IRL).

In the following subsections, BC and IRL are introduced in detail.

2.2.1 Behavior Cloning

Behavioral cloning directly learns a policy using supervised learning on state-action pairs from expert demonstrations. The IL agent is trained to reproduce the expert behaviors: for a given state, the action taken by the agent must be the one taken by the expert. In other words, for any given expert state-action pair (s, a), BC treat the state s as the label and the action a as the target. Then, IL becomes a classification or regression problem with state as the input and action as the output.

BC is a simple approach and does not require any interaction with the MDP, but the learned policy often poorly generalizes. The main reason for this is the assumption: supervised learning assumes that the state-action pairs are independent. However, in MDP, an action in a given state induces the next state, which breaks the previous assumption. Moreover, a mistake made by the agent can easily add up and put it into a state that the expert has never visited and the agent has never trained on. In such states, the behavior is undefined, leading to catastrophic failures. Therefore, although the main advantage of BC is its simplicity, it is unsuitable for tasks requiring long-term planning.

2.2.2 Inverse Reinforcement Learning

Inverse reinforcement learning is a different approach to imitation learning. Its main idea is to learn a representation of the underlying reward function R of the environment based on expert demonstrations. IRL parameterizes the reward function R as a linear combination of features:

$$R(s,a) = \mathbf{w}^T \phi(s,a)$$

where $\mathbf{w} \in \mathbb{R}^n$ is a weight vector and $\phi(s,a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^n$ denotes a feature vector.

For a given feature map ϕ, the goal of IRL is to determining the optimal weights $\mathbf{w}$ that maximizes the expected discounted sum of rewards along trajectories $J(\pi)$.

$$\begin{aligned} J(\pi) &= \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t R(s^t, a^t)] \\ &= \mathbf{w}^T \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t \phi(s^t, a^t)] \end{aligned}$$

It is important to note that, the optimal policy π^* always produce a better $J(\pi)$:

$$\begin{aligned} & J(\pi^*) \geq J(\pi) && ,\forall \pi \\ \Leftrightarrow\ & \mathbf{w}^T \mathbb{E}_{\pi^*}[\sum_{t=0}^{H} \gamma^t \phi(s^t, a^t)] \geq \mathbf{w}^T \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t \phi(s^t, a^t)] && ,\forall \pi \end{aligned}$$

Given the above constraint, it can be seen that $\mathbf{w} = 0$ satisfies the condition. This problem is called reward ambiguity, one of IRL's main challenges. Besides, for most expert's behavior, there are many fitting reward functions. Thus, finding an optimal reward function can be quite challenging. Despite that, IRL can find a better policy than BC and can be applied in complex and long-term planning tasks.

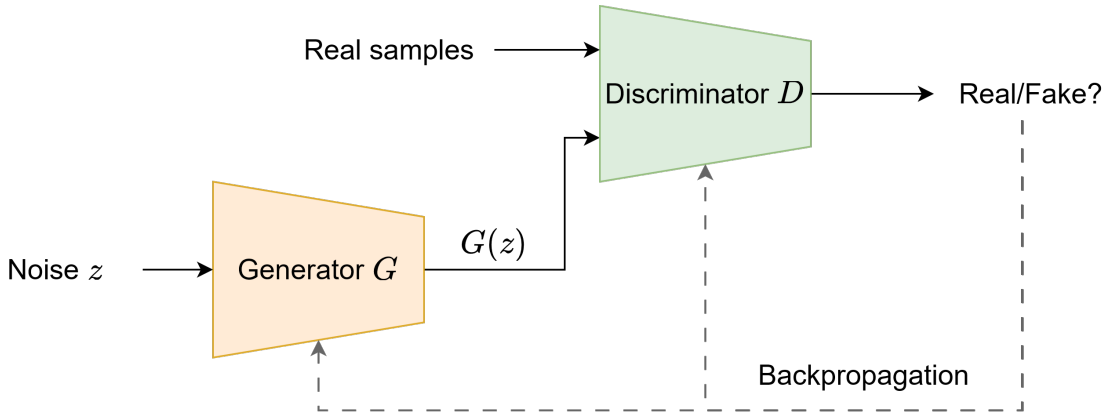


Figure 2.4. An illustration of a general GAN.

2.3 Adversarial Learning in Imitation Learning

2.3.1 Adversarial Learning

The objective of adversarial learning is to generate new samples that are indistinguishable from the training data. Adversarially learned models such as generative adversarial networks (GAN) [25] are a combination of two neural networks:

- The generator G tries to learn a data distribution and produces realistic fake data from a random noise z. The fake examples produced by the generator are used as negative examples for training the discriminator.
- The discriminator D estimates the probability that the sample comes from the real data distribution or not. In other words, it learns to distinguish fake data from real data. The discriminator also sends feedback to the generator so that the generator can improve the fake examples.

The optimal solution is a Nash equilibrium, which happens when the discriminator is no better than a random guess, meaning that the discriminator cannot discriminate between real and fake examples. The objective function is defined as follows:

$$\min_G \max_D L(G, D) = \mathbb{E}[logD(x)] + \mathbb{E}[log(1 - D(G(z)))]$$

where, $D(x)$ is probability that an input example x is real estimated by the discriminator, $G(z)$ is the fake example produced by the generator.

Both networks G and D are trained by backpropagating the loss for each network. The training alternates between training the generator and the discriminator. For each training epoch, the generator generates fake examples. These fake examples are then combined with the real training data and used for training the discriminator as a binary classifier. The discriminator is trained to classify training data as real and the generated data as fake. The goal for the generator is to minimize $log(1 - D(G(z)))$, so it fools the discriminator.

Generative adversarial network (GAN) is the most popular in adversarially learned models. The architecture of GANs is illustrated in Figure 2.4. Although it is originally proposed for unsupervised learning [26]–[28], GAN has also proved its capability in supervised learning, reinforcement learning, and imitation learning fields.

2.3.2 Generative Adversarial Imitation Learning

Inspired by GAN, Generative Adversarial Imitation Learning (GAIL) was proposed to train an IL agent. The architecture of GAIL also consists of two neural networks: generator G and discriminator G. The generator G aims to generate actions similar to the expert's behaviors as closely as possible. The discriminator D tries to distinguish between expert actions and generator actions. GAIL optimizes the following objective function:

$$\min G \max DL(G, D) = \mathbb{E}_{\pi}[\log D(s, a)] + \mathbb{E}_{\pi}[\log (1 - D(s, a))]$$

GAIL uses TRPO [17] to find an optimal policy The GAIL algorithm is presented in Algorithm 1.

Algorithm 1 GAIL

Input
  $\mathcal{D}_{\mathcal{E}} = \{\tau_{\mathcal{E}}^{i} : i \in [1, N]\}$ A set of expert demonstrations
Randomly initialize generator G and discriminator D
for i = 0,1,2,... **do**
  Sample trajectories $\tau^{i} \sim \pi$
  Update the discriminator with the gradient

$$\mathbb{E}_{\tau_{\mathcal{E}}}[\nabla_{D} \log D(s,a)] + \mathbb{E}_{\tau^{i}}[\nabla_{D} \log (1 - D(s,a))]$$

  Update the policy π using TRPO with the cost function $\log(1 - D(s,a))$.
end for
Output
  π^{*} Learned optimal policy

2.4 Literature Review on Domain and Task Adaptation in Imitation Learning

2.4.1 Imitation Learning Approaches

Reinforcement learning (RL) is an effective method to solve sequential decision-making tasks, where a learning agent interacts with the environment to improve its performance through trial and error [29]. RL has achieved exceptional success in challenging tasks, such as object manipulation [30]–[33], game playing [34]–[37], and autonomous driving [38]–[41]. Despite its remarkable advancement, RL still faces appealing difficulties caused by the need of a reward function [42], [43]. For each task that the agent has to accomplish, a carefully designed reward function must be provided. However, designing a hand-crafted reward function may require too much time or expense, especially in complex tasks. This problem has motivated a number of research studies on imitation learning (IL), where expert-generated demonstration data are provided instead of a reward function in order to help the agent learn how to perform a task [44], [45]. For this reason, IL has been growing in popularity and achieved some successes in numerous tasks, including robotics control [46]–[48] and autonomous driving [49]–[52].

A simple approach to imitation learning is Behavioral Cloning (BC) [53], mimics such demonstrations by learning the policy through supervised learning. Despite

successfully applied in many control problems [53]–[55], BC was found vulnerable to cascading errors [56]. On the other hand, Inverse Reinforcement Learning (IRL) [57] methods try to recover a reward function from the expert demonstrations [57]–[60]. This reward function is then used to optimize an imitation policy by running a standard reinforcement learning [61], [62]. Accordingly, IRL has succeeded in a wide range of tasks [63]–[66]. However, in order to train an IRL model, it requires iterations of reinforcement learning, which can be extremely computational expensive for high-dimensional tasks. Recently, Generative Adversarial Network [25] has been introduced and successfully employed to tackle complex challenges in image generation, translation and enhancement [26]–[28], [67]. Inspired by the great ability of GAN, recent studies [68]–[74] have applied it in imitation learning to define expert behaviors by fitting the distributions of states and actions. These models outperform competing methods when applying to complex high-dimensional tasks over various amounts of expert data.

2.4.2 Domain Adaptation Approaches in Imitation Learning

The common major weakness of the above-mentioned agents is that they require the experts to provide demonstrations in the same configuration and domain as the learners (i.e., the agents' domain). Thus, the presence of a shift between the expert and learner domains may lead to a significant performance deterioration of those agents. A popular approach is to employ a domain adaptation, which attempts to recover the learned policy from one domain and to adapt it to a different domain. The work in [75] proposed an agent to recover domain-agnostic features and utilized it to find optimal policies in the setting of third person imitation, in which the expert and learner observations come from different views. Furthermore, the authors in [76] introduced a two-step approach that could be applied to imitate demonstrations observed from a distinct domain. They proposed to find a state-action mapping between the expert and learner domains. After that, the learned mapping was utilized to adapt the learned policy to the learner domain. Although achieving high performance on low-dimensional tasks, the effectiveness of their methods on more complex high-dimensional tasks was not fully inspected yet.

2.4.3 Task Adaptation Approaches in Imitation Learning

Transfer learning (TL) aims to accelerate, adapt, and improve the agent's learning process on a new target task by transferring knowledge learned from the previous source task. Whereas TL has been intensively studied and shown appealing performance in supervised learning [77]–[83], it remains an open question in reinforcement learning and imitation learning fields. Fine tuning is the most explored approach for transfer learning in both RL and IL settings [84]–[86]. In fine tuning, the RL/IL agent is pre-trained on a source task and then retrained to a new target task. Fine tuning does not require strong assumptions about the target domain, making it an easily applicable approach. There are different approaches to transfer learning that have been proposed, such as reward shaping [87]–[89], inter-task mapping [90]–[92], representation learning [93]–[95], etc. However, these methods were designed for RL agents; directly applying them to transfer an IL agent does not necessarily lead to successful results since RL and IL differ in many factors. Moreover, the key challenge in transfer learning is catastrophic forgetting, in which the agent tends to unexpectedly lose the knowledge that was learned from the source task while transferring to the new target task. The reason is due to the changes in the agent's network parameters that are related to the source task getting overwritten to fulfill the target task's objectives [96]. Therefore, TL methods are not suitable for an agent that revisits the earlier task.

Chapter 3

Domain Adaptation in Imitation Learning with DAIL-GAN Agent

In this chapter, the DAIL–GAN agent is presented in order to address the domain adaptation problem in imitation learning.

3.1 Introduction

Imitation learning works by extracting information about the behavior of the expert and learning a mapping between the observation state and demonstrated behavior [97], [98]. Unfortunately, the traditional imitation learning algorithms are still far from being comparable with the human imitation due to the lack of the following abilities:

1. Humans tend to imitate the goal of a task rather than a particular behavior of the expert [99], [100].
2. Humans can recognize structural differences (i.e., domain shift) and similarities between the expert and themselves in order to adapt their behaviors accordingly [101].

The first aspect of human imitation can be modeled using Inverse Reinforcement Learning (IRL) [57], [58]. IRL seeks to estimate a reward function to explain an

expert behavior from demonstrations and subsequently train an agent on it [57]–[60]. Recent studies [68]–[74], [76] utilize Generative Adversarial Network (GAN) [25], which has a discriminator to judge whether a given behavior is from an expert or agent, and then a policy is trained using the discriminator as a reward. However, these approaches do not take into account the second aspect of human learning: imitation with the presence of domain shift between the expert and the agent. Such domain shift can mislead the feature learning, resulting in poor task performance.

The problem is formalized as domain adaptation in imitation learning, which is a process of learning how to perform a task optimally in the learner agent domain, given demonstrations of the task in a distinct expert domain [76]. In order to solve this problem, the authors in [76] proposed a two-step approach: alignment followed by adaptation. Firstly, the Generative Adversarial MDP Alignment (GAMA) was introduced to learn the state – action maps from demonstrations. Then, in the adaptation step, an optimal policy for the learner domain was obtained using the learned alignment from the first step. Despite showing a promising result, their agent was evaluated only in low-dimensional tasks. Besides, they updated the learned policy by using behavioral cloning, which was vulnerable to cascading errors. This could lead to poor adaptation performance in more complex high-dimensional tasks.

Unlike previous studies, a novel agent, namely DAIL–GAN, is proposed that aims to learn both domain-shared and domain-specific features. Such features enable the agents to learn optimal policies without being affected by the shift between two domains. The learning procedure can be achieved within one training process by utilizing adversarial learning [25]. It enables the agent to learn the extracted features, while at the same time, seeking for an optimal learner domain policy. A comprehensive experiment on both low and high-dimensional tasks is conducted to evaluate the performance of the proposed DAIL–GAN agent.

3.2 Problem Formulation

The domain adaptation in imitation learning can be formalized as a Markov decision problem which was introduced in Chapter 2. As a brief reminder, an episodic Markov Decision Process (MDP) $\mathbb{M}$ with finite time horizon H [29] is represented as the following equation:

$$\mathbb{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma, H)$$

where $\mathcal{S}$ and $\mathcal{A}$ represent the state and action spaces, respectively; $P(s'|s,a)$ denotes the transition function, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor—whereas, a policy $\pi(s|a) : \mathcal{S} \to \mathcal{A}$ for $\mathbb{M}$ describes a mapping from states $\mathcal{S}$ to actions $\mathcal{A}$.

It should be noted that in the domain adaptive imitation learning setting, the reward function is not given beforehand. Therefore, the MDP for a domain x without reward is defined as $\mathbb{M}_x^- = (\mathcal{S}_x, \mathcal{A}_x, P_x, \gamma_x, H_x)$. All examined domains are assumed to be alignable. That is, if considering two domains x and y, $\mathbb{M}_x^-$ can be reduced to $\mathbb{M}_y^-$, denoted as $\mathbb{M}_x^- \geq \mathbb{M}_y^-$, or vice versa [76]. An example is illustrated in Figure 3.1. Based on this expression, let $\mathcal{E}$ and $\mathcal{L}$ be the expert and the learner (agent) domain, respectively, $\mathbb{M}_\mathcal{E}^-$ and $\mathbb{M}_\mathcal{L}^-$ are said to be alignable if and only if $\mathbb{M}_\mathcal{E}^- \geq \mathbb{M}_\mathcal{L}^-$ or $\mathbb{M}_\mathcal{L}^- \geq \mathbb{M}_\mathcal{E}^-$ [76].

Furthermore, $\tau_x = \{(s_x^t, a_x^t) : t \in [0, \mathcal{H}]\}$ denotes a demonstration in the domain x, which is a sequence of state–action pairs. Then, a set of demonstrations $\mathcal{D}_\mathcal{E} = \{\tau_\mathcal{E}^i : i \in [1, N]\}$ from $\mathcal{E}$ is assumed to be available at the training time. With those assumptions, the main objective is being able to learn an optimal learner domain policy $\pi_\mathcal{L}^*$ against unknown reward functions $\mathcal{R}_\mathcal{E}$ and $\mathcal{R}_\mathcal{L}$, given the expert demonstrations $\mathcal{D}_\mathcal{E}$.

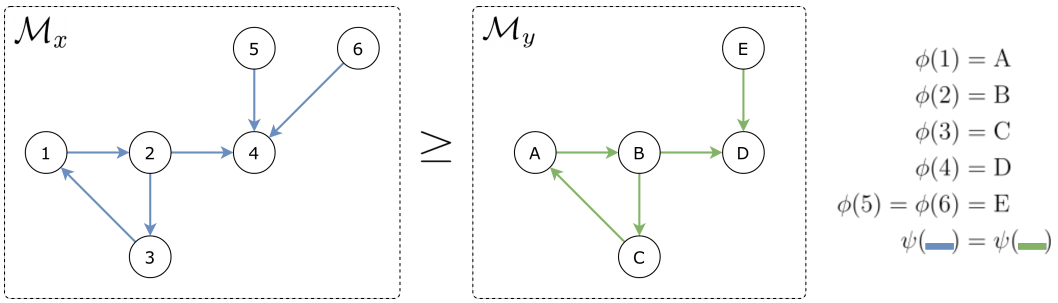


Figure 3.1. An example of two MDPs for x and y domains where $\mathbb{M}_x \geq \mathbb{M}_y$. $\phi : \mathcal{S}_x \to \mathcal{S}_y$ and $\psi : \mathcal{A}_x \to \mathcal{A}_y$ are state and actions maps, respectively. States correspond to nodes and actions to colors. States 5, 6 in $\mathcal{S}_x$ are merged to state e in $\mathcal{S}_y$ and blue actions in $\mathcal{A}_x$ are mapped to green actions in $\mathcal{A}_y$.

3.3 The Proposed DAIL–GAN Agent

In this section, the proposed DAIL–GAN agent is introduced. The agent relies on learning the domain-shared and domain-specific features in order to recover expert behaviors and adapt them to the learner agent domain. The architecture of the proposed agent is illustrated in Figure 3.2. The agent includes three deep feed-forward networks F, G, and D that holds different responsibilities.

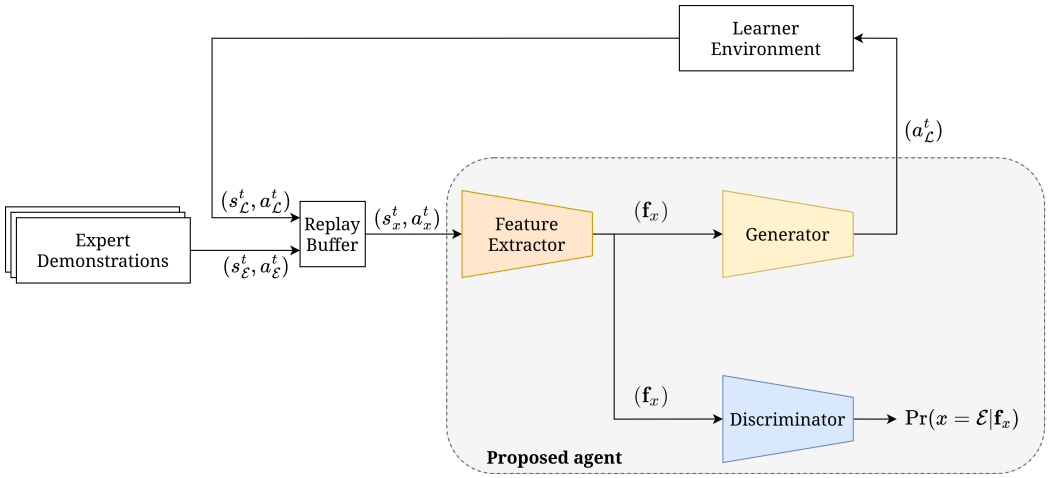


Figure 3.2. The neural network architecture of the proposed DAIL–GAN agent.

3.3.1 Feature Extractor Network F

A state-action pair (s^t_x, a^t_x) in domain x is sampled from a replay buffer [102] and input to the feature extractor F to produce a feature vector $\mathbf{f}_x = F(s^t_x, a^t_x)$. F is trained to capture the structural similarities or the shared features between $\mathcal{E}$ and $\mathcal{L}$ domains by minimizing the distance between two features $\mathbf{f}_{\mathcal{E}}$ and $\mathbf{f}_{\mathcal{L}}$. Therefore, the loss function of F is defined as:

$$\mathfrak{L}_F(F, G) = \mathbb{E}\left[\left\|F(s^t_{\mathcal{E}}, a^t_{\mathcal{E}}) - F(s^t_{\mathcal{L}}, a^t_{\mathcal{L}})\right\|\right] \tag{3.1}$$

$$= \mathbb{E}\left[\left\|F(s^t_{\mathcal{E}}, a^t_{\mathcal{E}}) - F(s^t_{\mathcal{L}}, G(F(s^t_{\mathcal{E}}, a^t_{\mathcal{E}})))\right\|\right] \tag{3.2}$$

3.3.2 Discriminator Network D and Generator Network G

The discriminator D is designed to distinguish between expert feature vector $\mathbf{f}_{\mathcal{E}}$ and learner feature vector $\mathbf{f}_{\mathcal{L}}$. Specifically, D receives a feature vector $\mathbf{f}_x$ outputs a probability $\Pr(x = \mathcal{E}|\mathbf{f}_x)$ to classify whether $\mathbf{f}_x$ is from $\mathcal{E}$ or $\mathcal{L}$. Meanwhile, the generator G aims to generate an action a_L^t so that $\mathbf{f}_{\mathcal{L}} = F(s_{\mathcal{L}}^t, a_{\mathcal{L}}^t)$ looks as similar as possible to $\mathbf{f}_{\mathcal{E}}$. In the proposed DAIL–GAN agent, the adversarial loss [25] is applied for both networks:

$$\mathfrak{L}_{GAN}(G, D) = \mathbb{E}[\log D(F(s_{\mathcal{E}}^t, a_{\mathcal{E}}^t))] + \mathbb{E}[\log(1 - D(F(s_{\mathcal{L}}^t, a_{\mathcal{L}}^t)))] \tag{3.3}$$

$$= \mathbb{E}[\log D(F(s_{\mathcal{E}}^t, a_{\mathcal{E}}^t))] + \mathbb{E}[\log(1 - D(F(s_{\mathcal{L}}^t, G(F(s_{\mathcal{E}}^t, a_{\mathcal{E}}^t)))))] \tag{3.4}$$

The optimal policy is achieved using a RL-based policy gradient, which relies on reward signal $r = -\log D(F(s_{\mathcal{E}}^t, a_{\mathcal{E}}^t))$ provided by the learned discriminator.

3.3.3 Full Objective

During the learning phase, in order to learn domain-shared features between $\mathcal{E}$ and $\mathcal{L}$ domains, the feature extractor F and the generator G are optimized to minimize the feature extractor loss $\mathfrak{L}_F$. At the same time, given a feature vector $\mathbf{f}_x$ of domain x, we want to judge whether $\mathbf{f}_x$ is from $\mathcal{E}$ or $\mathcal{L}$ by minimizing the domain classification loss $\mathfrak{L}_{GAN}$. This encourages domain-specific features to be captured by F. Overall, the full objective function is:

$$\underset{F,G}{max}\underset{D}{min}\mathfrak{L}(F, G, D) \tag{3.5}$$

$$\text{subject to } \mathfrak{L}(F, G, D) = \mathfrak{L}_{GAN}(G, D) - \lambda\mathfrak{L}_F \tag{3.6}$$

The goal is to find a saddle point, where:

$$(\hat{F}, \hat{G}) = \underset{F,G}{argmax}\mathfrak{L}(F, G, \hat{D}) \tag{3.7}$$

$$\hat{D} = \underset{D}{argmin}\mathfrak{L}(\hat{F}, \hat{G}, D) \tag{3.8}$$

$$\tag{3.9}$$

At the saddle point, the $\hat{D}$ minimizes the domain classification loss. The feature extractor $\hat{F}$ and the generator $\hat{G}$ minimize the distance between both domains (i.e. the features are shared between domains), while maximizing the domain classification loss (i.e. the features are specific to each domain). The parameter λ controls the trade-off between domain-shared features and domain-specific features should be learned by F.

The learning algorithm of the proposed agent is outlined in Algorithm 2.

Algorithm 2 DAIL–GAN

Input
    $\mathcal{D}_{\mathcal{E}}$ A set of expert demonstrations
Randomly initialize feature extractor network F, generator G and discriminator D
for i = 0, 1, 2, ... **do**
    Sample an expert demonstration $\tau_{\mathcal{E}}^{i} \sim \mathcal{D}_{\mathcal{E}}$
    Update the parameters of feature extractor network F with the gradient

$$\mathbf{E}[\nabla_F log(D(\mathbf{f}_{\mathcal{E}}))] + \mathbf{E}[\nabla_F log(1 - D(\mathbf{f}_{\mathcal{L}}))] - \lambda \mathbf{E}[\nabla_F \left\| \mathbf{f}_{\mathcal{E}} - \mathbf{f}_{\mathcal{L}} \right\|]$$

    Update the discriminator parameters with the gradient

$$\mathbf{E}[\nabla_D log(D(\mathbf{f}_{\mathcal{E}}))] + \mathbf{E}[\nabla_D log(1 - D(\mathbf{f}_{\mathcal{L}}))]$$

    Update policy π_L with the reward signal $r = -logD(\mathbf{f}_{\mathcal{E}})$
end for
Output
    π_L Learned policy for learner domain

3.4 Performance Evaluation

In this section, the performance of the proposed DAIL–GAN agent is evaluated by comparing with various baseline agents on a number of tasks ranging from low to complex high-dimensional. The details of the experimental settings and evaluation results are presented in the following subsections.

3.4.1 Experimental Settings

Environments

In this experiment, five simulated environments were considered: Pendulum [103], Acrobot [103]–[105], CartPole [103], [106], Door [107], and Hammer [107]. The detailed descriptions and visualizations of these environment are shown in Table 3.1 and Figure 3.3, respectively. From such environments, five domain adaptive tasks were decided, each of which included two different environments - an expert domain and a learner domain. These tasks can be divided into 2 categories as follows:

- **Low-dimensional tasks**:
  - Pendulum-Acrobot: Expert domain is Pendulum and learner domain is Acrobot.
  - Pendulum-CartPole: Expert domain is Pendulum and learner domain is CartPole.
  - Acrobot-CartPole: Expert domain is Acrobot and learner domain is CartPole.

  To provide expert demonstrations, for each task, the Trust Region Policy Optimization method [108] is first trained on the expert domain using the shaped reward signal. Then, 20 expert demonstrations are collected by executing the learned policies in the expert domain simulator. Each demonstration includes a sequence of state-action pairs. It should be noted that only successful demonstrations where the learned policies can accomplish the task are collected.
- **High-dimensional tasks**:

- Door-Door: The expert and learner domains have different friction parameter. The friction parameter in expert domain is $[1, 1, 1]$, while in learner domain is $[4.0, 4.0, 4.0]$.
- Hammer-Hammer: The expert and learner domains have different mass of the hammer. The mass of the hammer in expert domain is 0.253442, while in learner domain is 1.0.

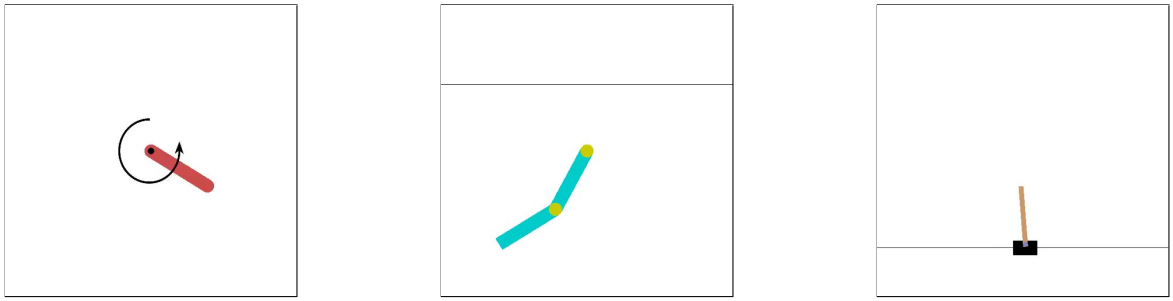
Twenty expert demonstrations are used for each task. The demonstrations are collected from humans using the Mujoco HAPTIX system [109] and publicly available [107].

Table 3.1. Description of five simulated environments used in the experiment.

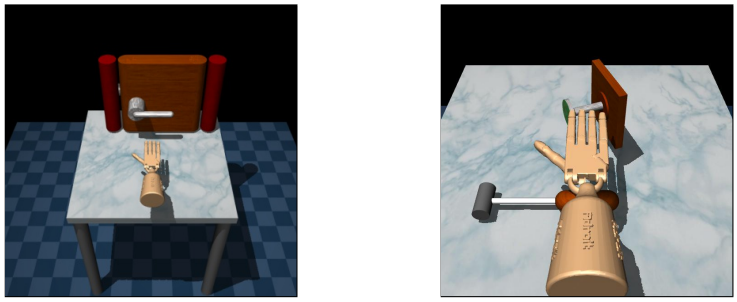
Task	State space	Action space	Description
Pendulum [103]	3 (continuous)	1 (continous)	Swinging up a pendulum.
Acrobot [103]–[105]	6 (continuous)	3 (discrete)	Swinging the end of the lower link up to a given height
CartPole [103], [106]	4 (continuous)	2 (discrete)	Preventing the pendulum from falling over by applying a force to the cart.
Door [107]	39 (continuous)	28 (continuous)	A 24-DoF hand attempts to undo the latch and swing the door open.
Hammer [107]	46 (continuous)	26 (continous)	A 24-DoF hand attempts to use a hammer to drive the nail into the board.

Table 3.2. DAIL–GAN hyperparameters used in the experiment. Each number corresponds to the number of nodes in a network layer.

	Feature Extractor F	**Generator** G	**Discriminator** D
Low-dimensional Tasks	(s_x^t, a_x^t) - 32 - 32 - 16	$(\mathbf{f}_x)$ - 32 - 32 - $(a_{\mathcal{L}}^t)$	$(\mathbf{f}_x)$ - 32 - 32 - 1
High-dimensional Tasks	(s_x^t, a_x^t) - 128 - 128 - 64	$(\mathbf{f}_x)$ - 128 - 64 - $(a_{\mathcal{L}}^t)$	$(\mathbf{f}_x)$ - 128 - 64 - 1



(a) Pendulum **(b)** Acrobot **(c)** CartPole



(d) Door **(e)** Hammer

Figure 3.3. Visual rendering of five simulated environments used in the experiment.

Baselines

The performance of the proposed DAIL–GAN agent was evaluated in comparison with the following baseline methods:

- Trust Region Policy Optimization (TRPO) [108] is a Reinforcement learning-based agent. The agent was trained directly on the learner domain and had access to the shaped reward function. This baseline set an upper bound for the performance of domain adaptation algorithms.
- GAMA-PA [76]: The agent introduced a two-step approach for domain adaptation in imitation learning. It first learns the state-action maps between expert and learner domains, and then utilizes it to learn an optimal policy. The agent parameters are employed as reported in [76] in order to ensure a fair comparison.

Network Structure and Hyperparameters

Deep feed-forward networks with 2 hidden layers are used for three F, G, D networks of the proposed agent. Grid search was utilized in order to find an optimal set of network hyperparameters. Since high-dimensional tasks are more challenging compared to low-dimensional tasks, DAIL-GAN requires a higher number of nodes in each network layer in order to provide a high and consistent performance. The network hyperparameters are shown in Table 3.2. In this experiment, the learning rate was 0.0003. Adam was used as an optimizer.

3.4.2 Results

In this subsection, the evaluation results of the proposed DAIL–GAN agent on low- and high-dimensional tasks are presented to highlight its superior capability in tackling domain adaptation problem in imitation learning.

Low-dimensional Tasks

Table 3.3 reports the quantitative evaluations of the proposed DAIL–GAN agent on low-dimensional tasks, in terms of average cumulative rewards. The numerical results clearly indicate that, for all evaluated tasks, TRPO [108] provided the best performance as its average cumulative rewards were at the highest. This was actually predictable because TRPO [108] had direct access to states and shaped rewards of the learner domain. On the other hand, inputs of GAMA-PA [76] and DAIL–GAN were limited to expert demonstrations only. As a result, their performances deteriorated compared to TRPO [108]. However, Table 3.3 also determines that the proposed DAIL–GAN outperformed GAMA-PA [76] across all three tasks. Additionally, for the Pendulum-Acrobot task, the proposed agent almost achieved as high performance as TRPO [108]. In order to understand the observed results more deeply, Figures 3.4, 3.5 and 3.6 visualize the behaviors of learned policies provided by the evaluated agents when performing the Pendulum-Acrobot, Pendulum-CartPole and Acrobot-CartPole tasks, respectively.

Table 3.3. The performance of the proposed DAIL–GAN agent on low-dimensional tasks. These scores represent the cumulative rewards obtained from executing a learned policy in the simulator, averaged over 100 episodes.

Task	**DAIL–GAN**	**GAMA-PA[76]**	**TRPO** [108]
Pendulum-Acrobot	-83.31 $\pm$ 32.61	-386.31 $\pm$ 49.20	-63.18 $\pm$ 7.05
Pendulum-CartPole	289.74 $\pm$ 171.21	144.03 $\pm$ 89.09	497.13 $\pm$ 28.56
Acrobot-CartPole	153.86 $\pm$ 81.79	93.05 $\pm$ 88.97	497.13 $\pm$ 28.56

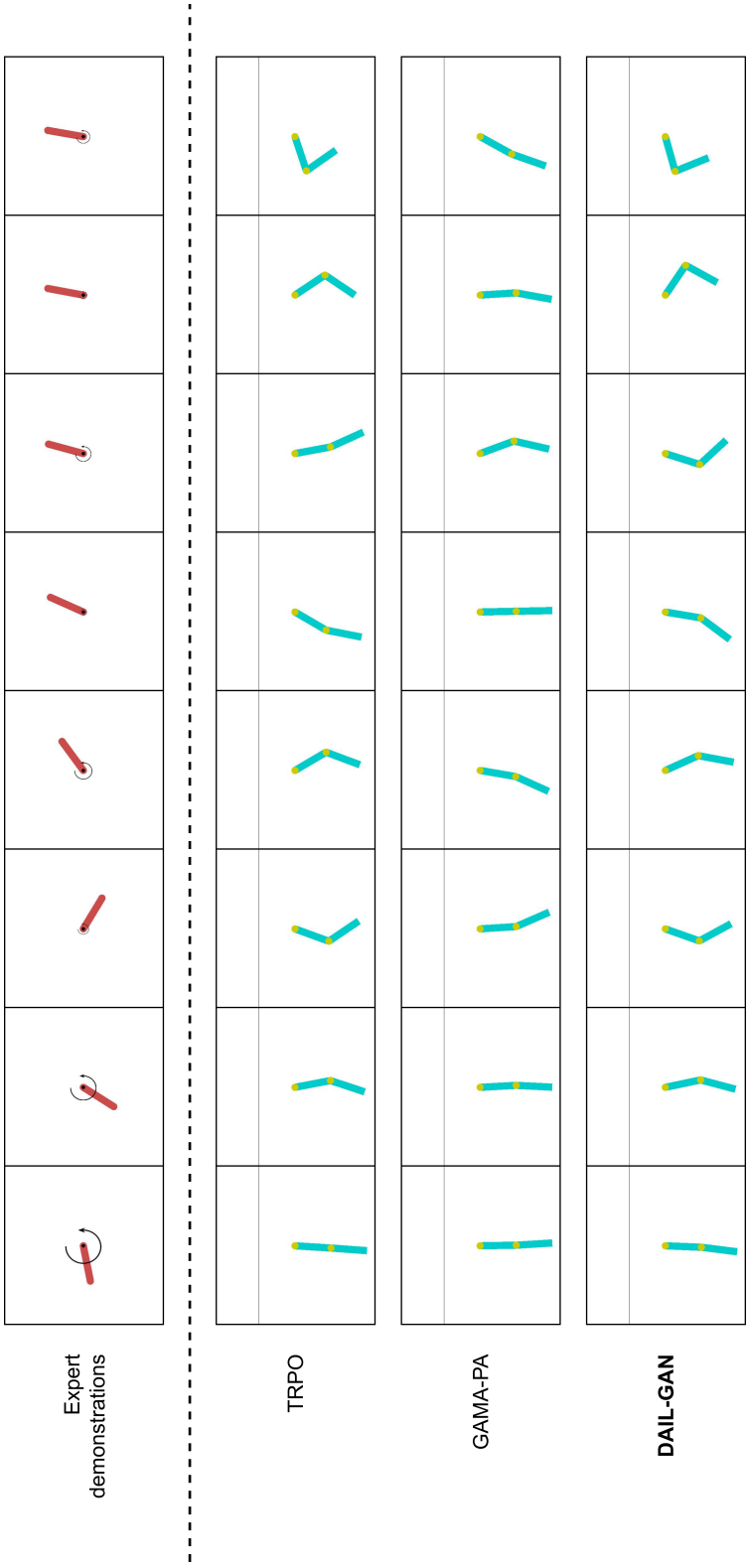


Figure 3.4. Pendulum-Acrobot.

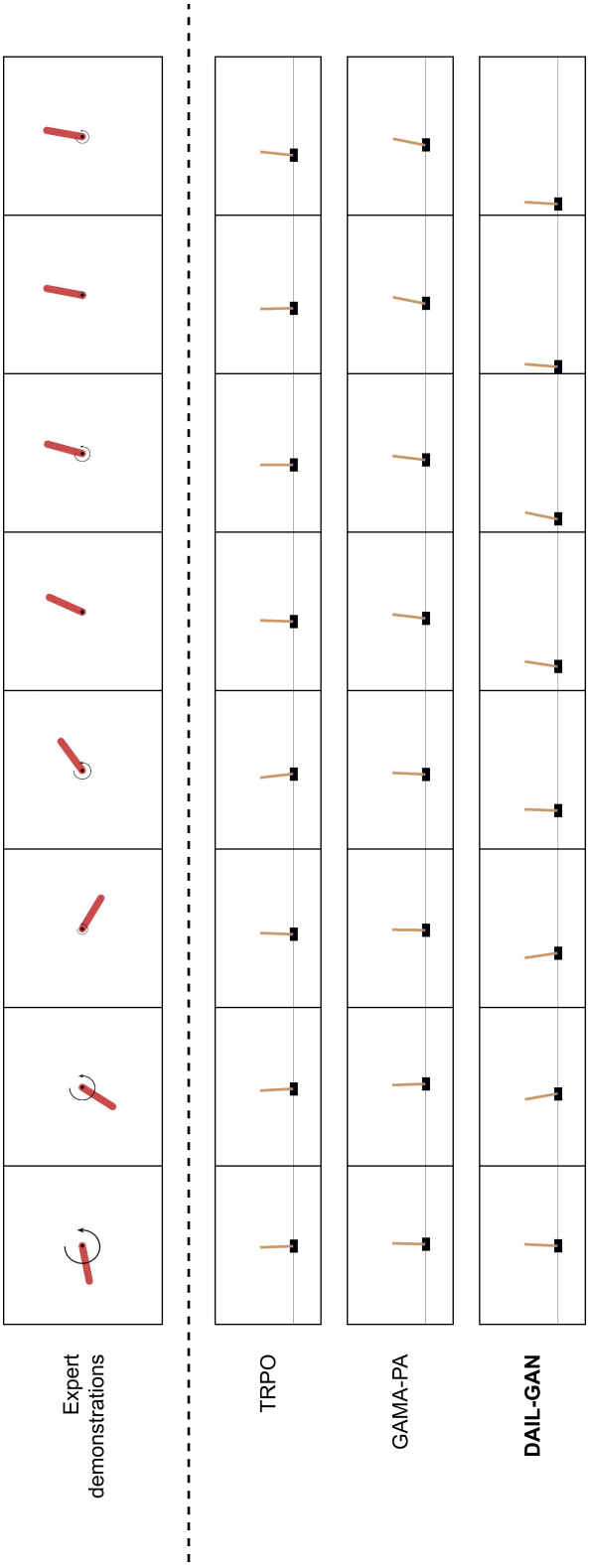


Figure 3.5. Pendulum-CartPole.

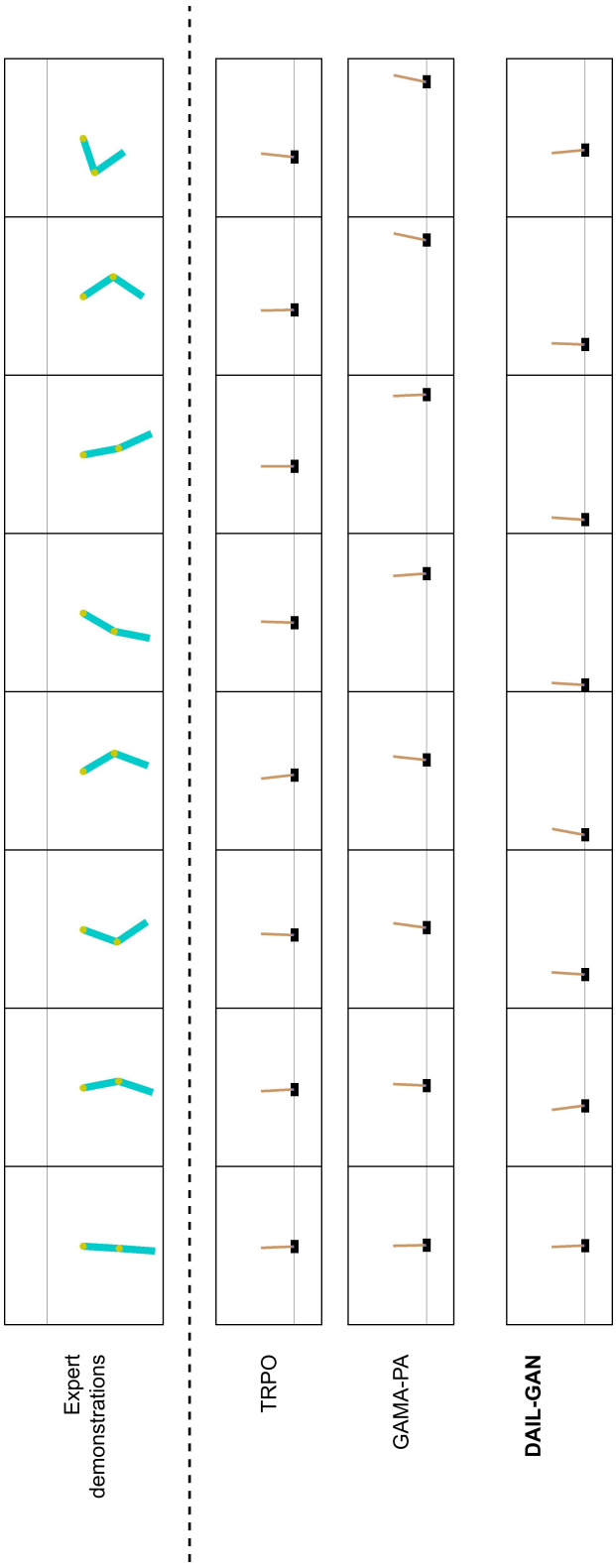


Figure 3.6. Acrobot-CartPole.

In the expert demonstration of the Pendulum-Acrobot task in Figure 3.4 and the Pendulum-CartPole task in Figure 3.5, expert behaviors were to apply a strong force, expressed by a rotation velocity, at first to make the pendulum swing upright. After that, a few light forces were applied to maintain it vertically. Observing from Figure 3.4, the policies trained with GAMA-PA [76] failed to apply strong enough forces to swing the lower link as high as the proposed DAIL. Also, Figure 3.5 expresses that the GAMA-PA [76] could not move the cart at an appropriate velocity to keep the pole stay vertical. It was because the expert demonstration also did not show much movement after successfully swinging the pendulum upright as it only applied light forces. Meanwhile, the policies learned by the DAIL–GAN agent could accomplish the task. Interestingly, it can be observed that the learned policies are able to produce behaviors that relatively similar to the expert: the cart was first pushed to the left by a strong force, then small forces are applied to prevent the pole from falling over.

For the Acrobot-CartPole task in Figure 3.6, the behaviors of the expert were that the link was swung back and forth to gain enough velocity to reach a higher height. Similarly, the GAMA-PA's learned policy could move the cart faster compared to the Pendulum-CartPole task. Yet, it still failed to maintain appropriate velocity to keep the pole standing. On contrary, the proposed DAIL–GAN was able to remain the pole vertical. It is important to note that the learned policy of DAIL–GAN could move the cart in both directions which is also similar to the expert behaviors.

The above observations show that the proposed DAIL–GAN agent not only succeeded in imitating the expert behaviors but also adapted the learned policies well to a distinct learner domain. Meanwhile, although the GAMA-PA [76] could learn the state – action maps from expert to learner domain, its adaptation algorithm was inefficient to help it accomplish the tasks.

High-dimensional Tasks

In this subsection, the performance of the proposed DAIL–GAN versus the referenced agents on the high-dimensional task is assessed. The average cumulative rewards of the evaluated agents are shown in Table 3.4. As expected, the TRPO agent achieved the highest average cumulative reward since it was trained directly on the learner domain. It is also revealed that DAIL–GAN outperformed GAMA-PA, although they were both unable to accomplish the Door-Door task. In addition, Figures 3.7 and 3.8

depict the policies learned by TRPO [108], GAMA-PA [76], and the DAIL–GAN agent, from which some interesting behaviors were observed.

Table 3.4. The performance of the proposed DAIL–GAN agent on high-dimensional tasks. These scores represent the cumulative rewards obtained from executing a learned policy in the simulator, averaged over 100 episodes

Task	**DAIL–GAN**	**GAMA-PA** [76]	**TRPO** [108]
Door-Door	-33.51 $\pm$ 8.87	-65.19 $\pm$ 0.77	2449.06 $\pm$ 1175.25
Hammer-Hammer	-78.84 $\pm$ 19.28	-252.52 $\pm$ 4.91	17030.25 $\pm$ 4357.23

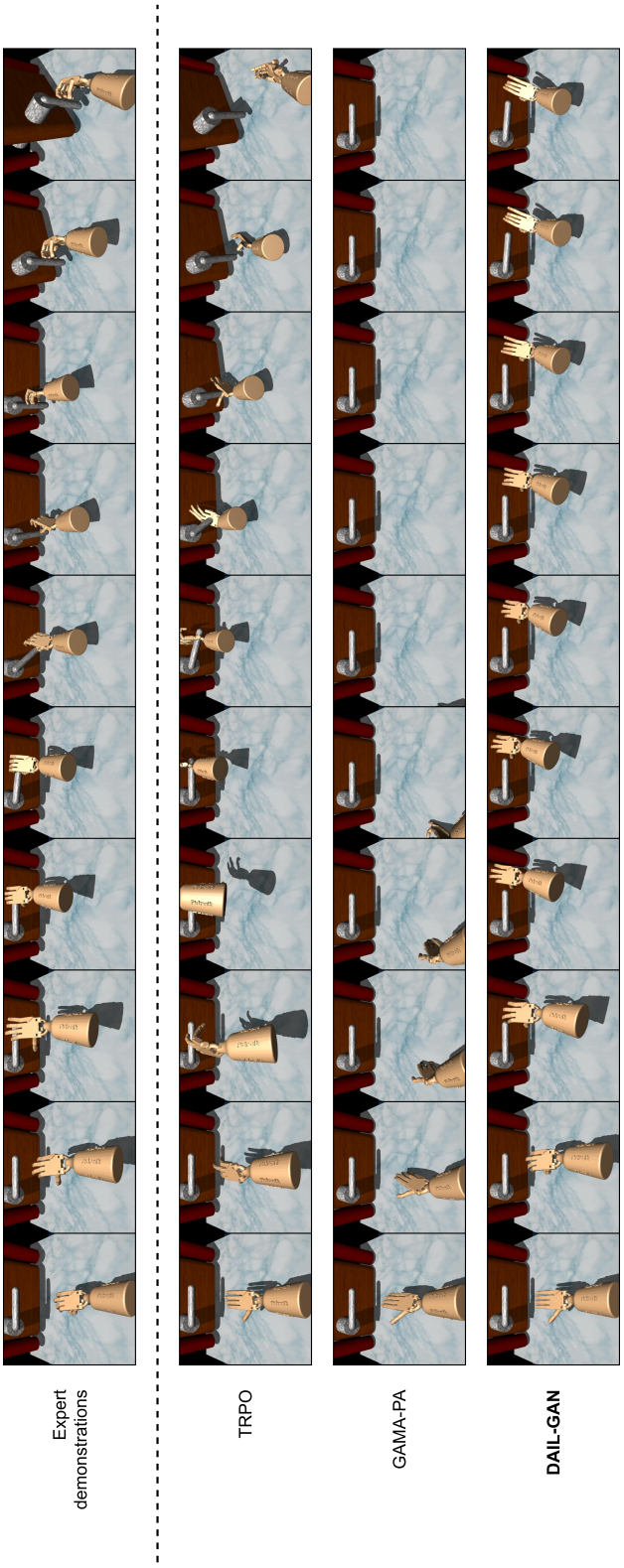


Figure 3.7. Door-Door.

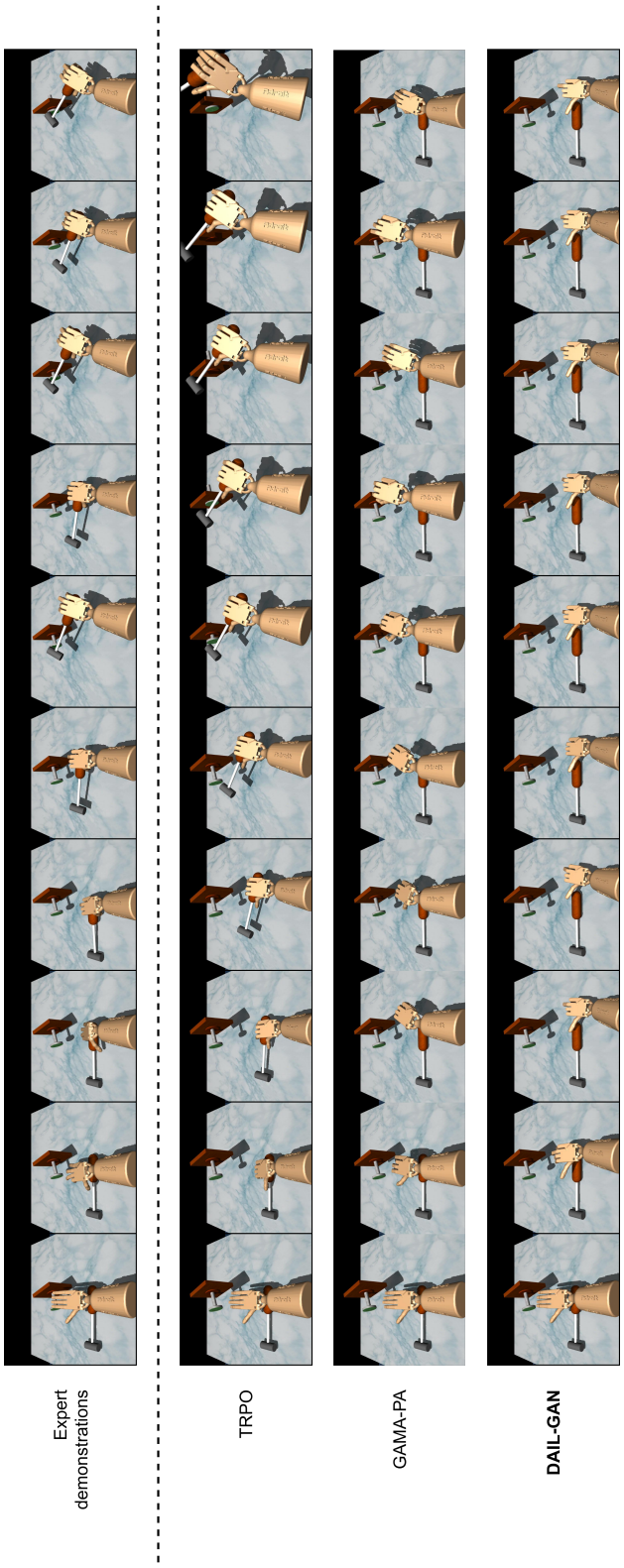


Figure 3.8. Hammer-Hammer.

As illustrated in Figure 3.7, the expert behaviors were understandable since their demonstrations were collected from humans: grab the handle, rotate it, then open the door. In Figure 3.8, the expert behaviors were to pick up and hammer multiple times in order to drive the nail into the board. While the policy trained with the TRPO could accomplish the task, it produced behaviors that were not human-like, i.e. unnatural use of wrist to rotate the handle. The main reason behind these unnatural behaviors was that the TRPO depended on a carefully reward shaping and it was challenging to formalize human-like behaviors into a mathematical reward function. On the other hand, with the use of expert demonstrations, the GAMA-PA and the proposed DAIL–GAN were expected to generate human-like behaviors. Yet, the policy learned by GAMA-PA failed to control the hand properly, as shown in Figures 3.7 and 3.8, due to the failure of the adaptation step in high-dimensional task. Meanwhile, it can be observed from Figures 3.7 and 3.8 that the policy trained with DAIL–GAN could produce more natural and human-like behaviors to move the robot hand closer to the door handle or the hammer. Unfortunately, DAIL–GAN could not rotate the handle or pick up the hammer in order to accomplish the task. Nevertheless, the human-like behaviors of the trained policies proved that DAIL–GAN could effectively extract and imitate expert behaviors from their demonstrations.

3.5 Discussion

This section discusses the overall performance of the proposed DAIL–GAN agent, followed by the importance of the feature extractor.

The quantitative and qualitative results assessed from the previous section have shown the potential of the proposed DAIL–GAN agent in addressing the domain adaptation problem in imitation learning. On both low- and high-dimensional tasks, DAIL–GAN could imitate expert behaviors from their demonstrations. Especially, the policies acquired by DAIL–GAN could even generate natural and human-like behaviors despite the high complexity of the Door-Door and Hammer-Hammer tasks. This indicates that the proposed DAIL–GAN could scale up to a complex manipulation task with a high-dimensional state and action space. Furthermore, the proposed agent could adapt the learned policies to a distinct learner domain and accomplish low-dimensional tasks without being affected by the presence of domain

shift between expert and learner domains. Although the success rate remained limited and depended on the complexity of the tasks, the proposed agent can be improved to provide a better performance toward practical real-world imitation learning tasks.

The promising performance of the proposed DAIL–GAN also praises the effectiveness of applying adversarial learning to train the agent. The adversarial learning enabled the feature extractor F to learn both domain-shared and domain-specific features between expert and learner domains. In Figure 3.5, the learned policy tended to move the cart to the left by a strong force initially, then followed by small forces; this behavior was similar to that of the expert demonstration. Such a similarity indicated that the feature extractor could extract the structural similarities or domain-shared features between expert and learner domain, resulting in comparable behaviors between them. Furthermore, it can also be observed in Figure 3.5 that although strong forces were applied, the learned policies still managed to keep the pole stay upright. This showed that the feature extractor was able to learn the differences between the expert and learner domains so that it could adapt the learned policies to the learner domain and accomplish the task. In summary, adversarial learning has proven its important role in DAIL–GAN. It could allow the agent to acquire shareable behaviors in both domains by learning the domain-shared features and adapting those behaviors to the learner domain regardless of the domain shift by learning the domain-specific features.

3.6 Summary

In this chapter, a novel DAIL–GAN agent was introduced to address the domain adaptation problem in imitation learning. The agent was able to extract the domain-shared and domain-specific features using adversarial learning. The comprehensive evaluation on both low and high-dimensional tasks demonstrates that the policies learned by the proposed agent can imitate expert behaviors and adapt them to a distinct learner domain. Thus, the potential of the proposed agent was verified.

Chapter 4

Task Adaptation in Imitation Learning with TAIL–GAN Agent

The previous chapter introduced DAIL–GAN agent to tackle the domain adaptation in imitation learning. In this chapter, the task adaptation problem is then focused on. The TAIL–GAN agent is introduced to address the problem.

4.1 Introduction

Imitation learning has been growing in popularity and has achieved some successes in numerous tasks, including robotics control [46]–[48] and autonomous driving [49]–[52]. Despite certain achievements, IL agents are designed to focus on accomplishing only a single, narrowly defined task. Therefore, when given a new task, the agent has to start the learning process again from the ground up, even if it has already learned a task that is related to and shares the same structure with the new one. On the other hand, humans possess an astonishing ability in the learning process, where the knowledge learned from source tasks can be leveraged for learning a new task. For example, an infant can reuse and augment the motor skills obtained when he learns to walk or uses his hand, for more complex tasks later in his life (e.g., riding a bike). Transfer learning (TL) is a technique based on this idea. TL enables the agent to reuse its knowledge learned from a source task in order to facilitate learning a new target task, resulting in a more generalized agent.

Recent studies have applied TL to RL/IL agents and achieved some success, especially in robot manipulation tasks since these tasks usually share a common structure (i.e., robot arm) [110]–[112]. However, transfer learning requires training the agent on the source task first. Then, the trained agent is adapted to the target task using fine tuning. Unlike transfer learning, the TAIL–GAN agent is proposed, which utilizes adversarial learning to train and adapt the agent simultaneously. The evaluation results show that the proposed agent has a better learning performance compared to existing transfer learning approaches.

4.2 Problem Formulation

The task problem in imitation learning can be formalized as an episodic Markov decision process (MDP). A MDP $\mathbb{M}_x^-$ for a task x with finite time horizon H_x [29] is represented as the following equation:

$$\mathbb{M}_x^- = (\mathcal{S}_x, \mathcal{A}_x, P_x, \gamma, H_x) \tag{4.1}$$

where $\mathcal{S}_x$ and $\mathcal{A}_x$ represent the continuous state and action spaces, respectively; $P_x(s'|s,a)$ denotes the transition probability function; γ is the discount factor. A stochastic policy $\pi_x(s,a)$ for $\mathbb{M}_x^-$ describes a mapping from each state to the probability of taking each action. The goal of an IL agent is to learn an optimal policy π_x^* that imitates the expert policy $\hat{\pi}_x$ given demonstrations from that expert. An expert demonstration for a task x is defined as a sequence of state–action pairs $\tau_x = \{(\hat{s}_x^t, \hat{a}_x^t) : t \in [0, H_x]\}$. It should be noted that the set of expert demonstrations is collected under different domain and settings.

Let $\mathbb{M}_S^-$ denote a source task, which provides prior knowledge $\mathcal{K}_S$ that is accessible by the target task $\mathbb{M}_T^-$, such that by leveraging $\mathcal{K}_S$, the target agent learns better in the target task $\mathbb{M}_T^-$. The main objective is to learn an optimal policy $\pi_T^*(\mathcal{K}_S, \mathcal{K}_T)$ for target tasks, by leveraging $\mathcal{K}_S$ from $\mathbb{M}_S^-$.

4.3 The Proposed TAIL–GAN Agent

In this section, the proposed TAIL–GAN agent is introduced. The architecture of the proposed agent is the same as DAIL–GAN, which is illustrated in Figure 4.1. It is important to note that the loss functions for three deep feed-forward networks F, G, and D are similar to the DAIL–GAN agent. The only difference is that the TAIL–GAN agent is trained on both source and target tasks simultaneously and under the same domain. Thus, instead of extracting domain-shared and domain-specific features, TAIL–GAN is trained to learn the similarities and differences between source and target tasks.

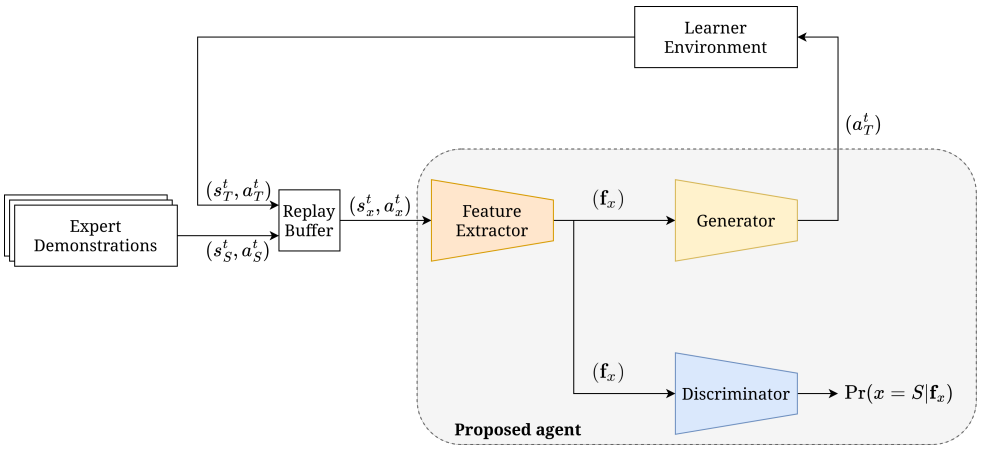


Figure 4.1. The neural network architecture of the proposed TAIL–GAN agent.

4.3.1 Feature Extractor Network F

A state-action pair (s_x^t, a_x^t) in task x is sampled from a replay buffer [102] and input to the feature extractor F to produce a feature vector $\mathbf{f}_x = F(s_x^t, a_x^t)$. F is trained to capture the structural similarities or differences between source $\mathbb{M}_S^-$ and target $\mathbb{M}_T^-$ tasks by minimizing the distance between two features $\mathbf{f}_S$ and $\mathbf{f}_T$. Therefore, the loss function of F is defined as:

$$\mathfrak{L}_F(F, G) = \mathbb{E}\left[\left\|F(s_S^t, a_S^t) - F(s_T^t, a_T^t)\right\|\right] \tag{4.2}$$

$$= \mathbb{E}\left[\left\|F(s_S^t, a_S^t) - F(s_T^t, G(F(s_S^t, a_S^t)))\right\|\right] \tag{4.3}$$

4.3.2 Discriminator Network D and Generator Network G

The discriminator D is designed to distinguish between $\mathbf{f}_S$ and $\mathbf{f}_T$. Specifically, D receives a feature vector $\mathbf{f}_x$ outputs a probability $\Pr(x = S|\mathbf{f}_x)$ to classify whether $\mathbf{f}_x$ is from source $\mathbb{M}_S^-$ or target $\mathbb{M}_T^-$ tasks. Meanwhile, the generator G aims to generate an action a_T^t so that $\mathbf{f}_T = F(s_T^t, a_T^t)$ looks as similar as possible to $\mathbf{f}_S$. In the proposed DAIL–GAN agent, the adversarial loss [25] is applied for both networks:

$$\mathfrak{L}_{GAN}(G, D) = \mathbb{E}[\log D(F(s_S^t, a_S^t))] + \mathbb{E}[\log(1 - D(F(s_T^t, a_T^t)))] \tag{4.4}$$

$$= \mathbb{E}[\log D(F(s_S^t, a_S^t))] + \mathbb{E}[\log(1 - D(F(s_T^t, G(F(s_S^t, a_S^t)))))] \tag{4.5}$$

The optimal policy is achieved using a RL-based policy gradient, which relies on reward signal $r = -\log D(F(s_S^t, a_S^t))$ provided by the learned discriminator.

4.3.3 Full Objective

During the learning phase, in order to capture the similarities and differences between source and target tasks, the feature extractor F and the generator G are optimized to minimize the feature extractor loss $\mathfrak{L}_F$. At the same time, given a feature vector $\mathbf{f}_x$ of task x, we want to judge whether $\mathbf{f}_x$ is from $\mathbb{M}_S^-$ or $\mathbb{M}_T^-$ by minimizing the task classification loss $\mathfrak{L}_{GAN}$. This encourages task-specific features to be captured by F. Overall, the full objective function is:

$$\max_{F,G} \min_{D} \mathfrak{L}(F, G, D) \tag{4.6}$$

$$\text{subject to } \mathfrak{L}(F, G, D) = \mathfrak{L}_{GAN}(G, D) - \lambda \mathfrak{L}_F \tag{4.7}$$

The learning algorithm of the proposed agent is outlined in Algorithm 3.

Algorithm 3 TAIL–GAN

Input
  $\mathcal{D}_S$ A set of expert demonstrations on source tasks
Randomly initialize feature extractor network F, generator G and discriminator D
for i = 0, 1, 2, ... **do**
  Sample an expert demonstration $\tau_S^i \sim \mathcal{D}_S$
  Update the parameters of feature extractor network F with the gradient

$$\mathbf{E}[\nabla_F log(D(\mathbf{f}_S))] + \mathbf{E}[\nabla_F log(1 - D(\mathbf{f}_T))] - \lambda \mathbf{E}[\nabla_F \|\mathbf{f}_S - \mathbf{f}_T\|]$$

  Update the discriminator parameters with the gradient

$$\mathbf{E}[\nabla_D log(D(\mathbf{f}_S))] + \mathbf{E}[\nabla_D log(1 - D(\mathbf{f}_T))]$$

  Update policy π_L with the reward signal $r = -logD(\mathbf{f}_S)$
end for
Output
  π_T Learned policy for target task

4.4 Performance Evaluation

In this section, the performance of the proposed TAIL–GAN is evaluated in comparison with baselines. To support the evaluation, different simulated tasks with varying difficulty levels ranging from simple to complex ones were utilized. The details of these tasks are described in the next subsection.

4.4.1 Experimental Settings

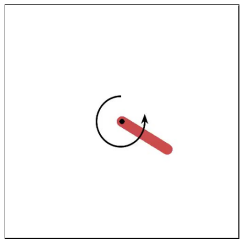
Simulated Tasks

In order to examine the effectiveness of the proposed method, six simulated tasks with varying difficulties were considered: Pendulum [113], CartPole [113], [114], WindowOpen [115], WindowClose [115], Door [116], and Hammer [116]. The task difficulty is varied along two axes; the size of the state space and the size of the action space. The detailed descriptions and visualizations of these tasks are shown in Table 4.1 and Figure 4.2. From such tasks, three experiments were conducted, each

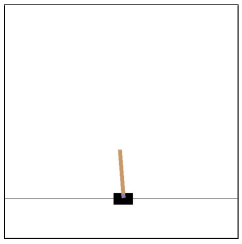
of which included two different tasks—a source task and a target task. The detailed descriptions of these experiments are shown in Table 4.2.

Table 4.1. Description of six simulated tasks used in the experiment.

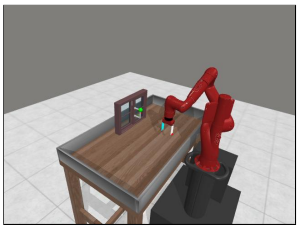
Task	Size of State Space	Size of Action Space	Difficulty Level	Description
Pendulum [113]	3 (continuous)	1 (continuous)	Easy	Swinging up a pendulum.
CartPole [113], [114]	4 (continuous)	1 (continuous)	Easy	Preventing the pendulum from falling over by applying a force to the cart.
WindowOpen [115]	39 (continuous)	4 (continuous)	Medium	Opening a window.
WindowClose [115]	39 (continuous)	4 (continuous)	Medium	Closing a window.
Door [116]	39 (continuous)	28 (continuous)	Hard	A 24-DoF hand attempts to undo the latch and swing the door open.
Hammer [116]	46 (continuous)	26 (continuous)	Hard	A 24-DoF hand attempts to use a hammer to drive the nail into the board.



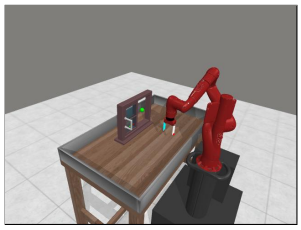
(a) Pendulum



(b) CartPole

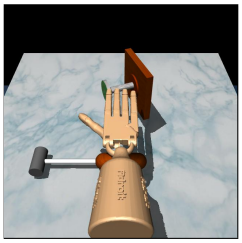


(c) WindowOpen



(d) WindowClose

(e) Door



(f) Hammer

Figure 4.2. Visual rendering of five simulated tasks used in the experiment.

Table 4.2. Description of three experiments conducted to evaluate the performance of the proposed method.

Experiment	Source Task	Target Task	Difficulty Level	Description
Pendulum–CartPole	Pendulum	CartPole	Easy	A simple experiment in which both source and target tasks have small state and action spaces.
WindowOpen–WindowClose	WindowOpen	WindowClose	Medium	Both source and target tasks have a large state space but small action space.
Door–Hammer	Door	Human	Hard	A challenging experiment in which both source and target tasks have large state and action spaces.

In order to train and adapt the proposed TAIL–GAN agent, expert demonstrations for both source and target tasks must be provided. In this experiment, the proximal policy optimization (PPO) method was chosen to be trained on each task in order to create an expert RL agent. The reason behind this decision was that PPO was recently showing the best result for many complex tasks. After that, the demonstrations were collected by executing the trained PPO expert agent in the simulated task. For the source task, 30 demonstrations were collected to provide sufficient data for training the proposed agent [68]. In the adaptation process, the proposed agent already learned the knowledge of the source task, thus, a smaller number of demonstrations for the target task is required. Therefore, only 15 demonstrations were collected for the target task.

Baselines

To evaluate the performance of the proposed agent, two baselines were considered.

PPO + Fine-tuning The agent is trained on the source task using Proximal Policy Optimization (PPO) [117], which is a policy optimization method. Then, fine-tuning is applied to adapt the trained agent to the target task. Fine-tuning is a common transfer learning technique that simply re-trains the agent on a new target task.

NFQI + TA-TL NFQI + TA-TL is a policy adaptation method, where first it utilizes Neural Fitted Q-iteration (NFQI) [118] to find an optimal policy on a source task, then that policy is transferred to a new target task. NFQI is a Q-learning method that tries to estimate the Q-function using a deep feed-forward network.

In order to provide a fair comparison, each baseline was evaluated for 100 trials. The success rate and average cumulative reward were used as performance metrics. The success rate indicates the percentage of trials in which the baseline can successfully complete a task. The average cumulative reward measures how well the baseline performed in a trial.

4.4.2 Results

The result is tabulated in Table 4.3. The behavior of those agents when performing target tasks is visualized in Figure 4.3. It can be seen that the proposed TAIL–GAN and baselines provide comparably similar behaviors in order to solve target tasks. This result indicated that the proposed agent successfully adapted and transferred the agent's knowledge to the new target task. Moreover, it can be observed from Table 4.3 that TAIL–GAN outperformed other baselines. In addition, it performed highly well and consistently on the complex WindowClose and Hammer tasks. Applying fine tuning to the PPO agent also provided a consistent performance across all three tasks. At the same time, applying TA-TL to the NFQI agent was not able to produce a high success rate due to the high complexity of the WindowClose and Hammer tasks.

The results demonstrated that the proposed agent not only outperformed baselines in terms of success rate on all target tasks, but notably produced a consistently high performance, even on the most difficult task. This proved the potential of the proposed agent and adversarial learning in order to tackle the task adaptation problem in imitation learning.

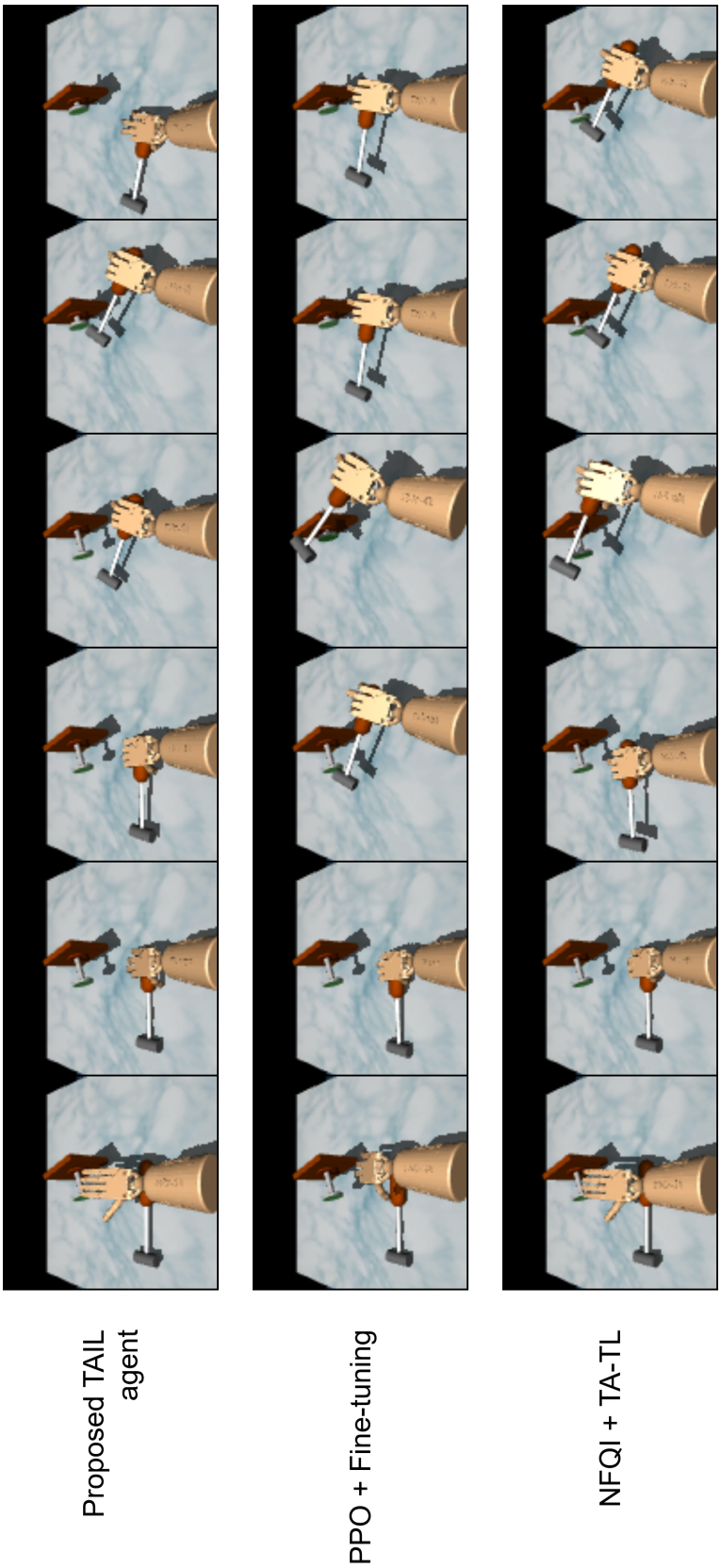
Table 4.3. The performance of the proposed agent on target tasks after adaptation.

		CartPole	**WindowClose**	**Hammer**
Success rate	TAIL–GAN	100%	85%	80%
	PPO [117] + Fine-tuning	100%	82%	76%
	NFQI + TA-TL [119]	100%	77%	70%
Average cumulative reward	TAIL–GAN	500.00 ± 0.0	2473.10 ± 619.13	3351.69 ± 1957.30
	PPO [117] + Fine-tuning	500.00 ± 0.0	2402.65 ± 638.54	3165.05 ± 1134.02
	NFQI + TA-TL [119]	500.00 ± 0.0	1457.53 ± 621.37	2840.35 ± 1036.76

(a) CartPole

(b) WindowClose

Figure 4.3. *Cont.*



(c) Hammer

Figure 4.3. A visualization of the behavior of the proposed agent and baselines on target tasks.

4.5 Discussion

In this section, the effects of adversarial learning on training the TAIL–GAN agent are discussed in detail.

The experimental results assessed in the previous section have shown the potential of TAIL–GAN in tackling the task adaptation problem in imitation learning. As shown in Tables 4.3, TAIL–GAN could provide consistent and high performance in terms of success rate target tasks with varying difficulty levels. This promising result demonstrates the effectiveness of applying adversarial learning, in which the agent can extract the similarities and differences between source and target tasks to adapt its learned knowledge effectively. Although the performance of TAIL–GAN remained limited, it has demonstrated the potential of adversarial learning in improving generalization in imitation learning.

4.6 Summary

In this chapter, the TAIL–GAN agent was proposed to tackle the task adaptation problem in imitation learning. The experimental results demonstrated that the agent was able to extract the similarity and differences between source and target tasks in order to provide high performance on the target task in terms of success rate.

Chapter 5

Repetition-Based Approach for Domain and Task Adaptation in Imitation Learning

In this chapter, the DTAIL–GAN agent is introduced to further extend the potential of adversarial learning in addressing the generalization challenge in imitation learning.

5.1 Introduction

In Chapters 3 and 4, the domain and task adaptation problems in IL were addressed individually. Although DAIL–GAN and TAIL–GAN agents can provide a competitive performance compared to other baselines, there is still an enormous difference between human ability and IL agents. DAIL–GAN and TAIL–GAN are designed to leverage the learned knowledge to accelerate the acquisition of the new target domain/task. Thus, the learning performance on the target domain/task may be improved in exchange for the deterioration of the source domain/task's performance. In other words, the agent forgets how to perform the previously learned domain/task when learning a new one, which is described as the catastrophic forgetting problem [96], [120]. On the contrary, humans can perform well on both source and target tasks on different domains.

To address the aforementioned gap, a novel challenge on domain and task adaptation in imitation learning is discussed in this chapter, in which a trained agent on a source task faces a new target task and must optimize its overall performance on both tasks. In order words, the main goal is to help the agent achieve high learning performance on the target task, while avoiding performance deterioration on the source task. It is important to note that in this chapter, the agent is trained and performed on different domains.

The problem can be served as a step toward building a general-purpose agent. As one illustrative example, consider a household robot learning to assist its human owner. Initially, the human might want to teach the robot to load clothes into the washer by providing demonstrations of the task. At a later time, the user could teach the robot to fold clothes. These tasks are related to each other since they involve manipulating clothes, hence the robot is expected to perform well on both tasks and leverage any relevant knowledge obtained from loading the washer while folding clothes. In order to achieve such a knowledge transfer ability, a task adaptation method for imitation learning is proposed in this chapter. Being inspired by the idea of repetition learning in neuroscience [121]–[123], the general idea of the proposed method is to make the agent repeatedly review the learned knowledge of the source task while learning the target task at the same time. Accordingly, the proposed method is two-fold. Firstly, to allow the agent to repeatedly review the learned knowledge of the source task, a task adaptation algorithm is proposed. In the adaptation process, the learned knowledge is expanded by adding the knowledge of the target task. Secondly, a novel IL agent which is capable of finding an optimal policy using expert-generated demonstrations, is proposed. This agent allows the learned knowledge of the source task to be encoded into a high-dimensional vector, namely task embedding, which then supports the knowledge expansion in the adaptation process. The evaluation results show that the proposed method has a better learning ability compared to existing transfer learning approaches.

5.2 Problem Formulation

The task and domain adaptation problem in imitation learning can be formalized as an episodic Markov decision process (MDP). A MDP $\mathbb{M}_x^-$ for a task x with finite time

horizon H_x [29] is represented as the following equation:

$$\mathbb{M}_x^- = (\mathcal{S}_x, \mathcal{A}_x, P_x, \gamma, H_x) \tag{5.1}$$

where $\mathcal{S}_x$ and $\mathcal{A}_x$ represent the continuous state and action spaces, respectively; $P_x(s'|s,a)$ denotes the transition probability function; γ is the discount factor. A stochastic policy $\pi_x(s,a)$ for $\mathbb{M}_x^-$ describes a mapping from each state to the probability of taking each action. The goal of an IL agent is to learn an optimal policy π_x^* that imitates the expert policy $\hat{\pi}_x$ given demonstrations from that expert. An expert demonstration for a task x is defined as a sequence of state–action pairs $\tau_x = \{(\hat{s}_x^t, \hat{a}_x^t) : t \in [0, H_x]\}$. It should be noted that the set of expert demonstrations is collected under different domain and settings.

Let $\mathbb{M}_S^-$ denote a source task, which provides prior knowledge $\mathcal{K}_S$ that is accessible by the target task $\mathbb{M}_T^-$, such that by leveraging $\mathcal{K}_S$, the target agent learns better in the target task $\mathbb{M}_T^-$. The main objective is to learn an optimal policy $\pi_{ST}^*(\mathcal{K}_S, \mathcal{K}_T)$ for both source and target tasks, by leveraging $\mathcal{K}_T$ from $\mathbb{M}_T^-$ as well as $\mathcal{K}_S$ from $\mathbb{M}_S^-$.

5.3 The Proposed Adaptation Method

The proposed method presented in this section involves two main processes: learning from a source task and adapting to a new target task. The main objective is to build an agent that can perform consistently well on both source and target tasks. In order to achieve this, the general of this novel idea is to allow the agent to repeatedly review the knowledge learned from the source task, while learning the new knowledge of the target task. The idea is inspired by a human learning effect, which is repetition learning. Prior studies in neuroscience have proved that when humans learn by repetition, their memory performance can be enhanced and retained for a longer time [121]–[123], giving humans the unique ability to perform most sophisticated tasks with ease. Therefore, developing a similarly intelligent method is focused on in order to achieve the main research objective and to tackle the task adaptation problem in imitation learning.

Accordingly, the proposed method is two-fold. Firstly, an adaptation algorithm is proposed to allow the agent to learn the new target task by expanding its knowledge.

More concretely, on top of the knowledge that the agent has learned from a source task, the knowledge of a target task is added. In addition, the agent repeatedly uses such knowledge to learn the target task and review the previously learned source task to ensure that the learning performance on the target task is high, while the deterioration of the learning performance on the source task is small. Secondly, to support the expansion of the to-be-learned knowledge, a novel imitation learning (IL) agent is proposed. This agent encodes the learned knowledge into a latent space, namely task embedding space, in which the learned knowledge from task x at time step t can be represented by a high-dimensional vector $z_x^t \in \mathbb{R}^n$. Figure 5.1 illustrates the task embedding space before and after applying the proposed task adaptation algorithm. The task embedding space allows the proposed adaptation algorithm to add the new knowledge of the target task while minimizing its impacts on the source task's knowledge. In addition, since the source and target tasks are related to each other, there are some common knowledge between those two tasks. This shared common knowledge can be captured by the task embedding that helps accelerate the adaptation process. The details of the proposed method are provided in the following sub-sections.

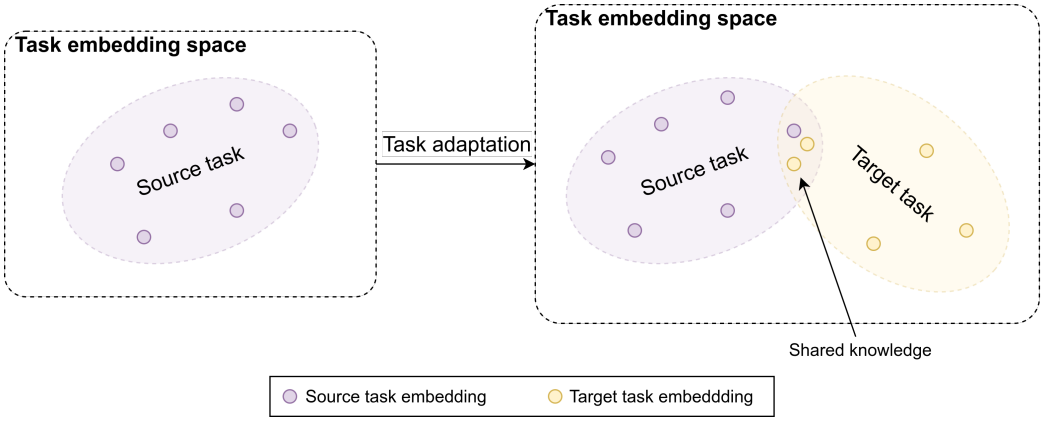


Figure 5.1. An illustration of the task embedding space. Purple and yellow regions denote the knowledge learned from the source and target tasks, respectively. Applying the proposed task adaptation algorithm will lead to the expansion of the task embedding space due to the acquisition of the knowledge of the target task. In addition, the intersection between those two regions indicates the shared common knowledge between the two tasks.

5.3.1 The Proposed DTAIL–GAN Agent

In this subsection, the proposed agent is described in detail. The proposed agent is an imitation learning method that finds an optimal policy for the source task using expert-generated demonstration data. The agent is capable of encoding the learned knowledge into a task embedding in order to support the later adaptation progress. The architecture of the proposed agent is illustrated in Figure 5.2. The proposed agent is a combination of three deep feed-forward networks E, G, and D, which have different responsibilities.

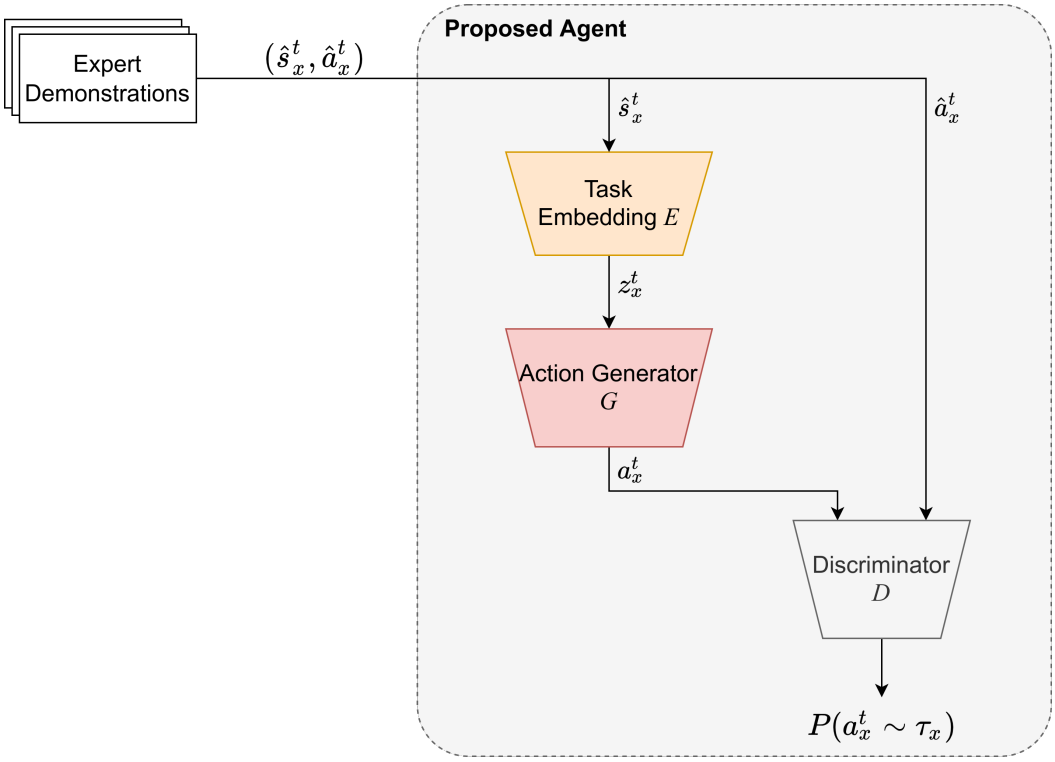


Figure 5.2. The neural network architecture of the proposed agent.

Task-Embedding Network E

The task-embedding network E is designed to encode the learned knowledge into a high-dimensional task embedding space. Specifically, E maps a state s_x^t of task x at time step t into a task embedding $z_x^t = E(s_x^t)$, $z_x^t \in \mathbb{R}^n$. Since z_x^t contains the information of the task, it is expected that z_x^t can capture the similarities and differences between source and target tasks. In order to achieve that, contrastive

learning is introduced to train E. Contrastive learning aims to bring task embeddings of the same task close to each other in the task embedding space and to push dissimilar ones far apart. In order words, E is trained to minimize distance $d(z_S^t, z_S^t)$ and maximize distance $d(z_S^t, z_T^t)$, where $d(\cdot)$ is a negative cosine similarity function defined as

$$d(z_x^t, z_y^t) = -\frac{z_x^t \cdot z_y^t}{||z_x^t|| * ||z_y^t||} \tag{5.2}$$

where x and y can be the same or different task.

The optimization function $\mathcal{L}_E$ to train E is defined as follows:

$$\min_E \mathcal{L}_E(z_x^t, z_y^t) = \mathbb{1}[x = y]d(z_x^t, z_y^t) + \mathbb{1}[x \neq y](-d(z_x^t, z_y^t)) \tag{5.3}$$

where $\mathbb{1}(\cdot) \in \{0, 1\}$ is an indicator function.

Action Generator Network G and Discriminator Network D

The action generator network G aims to generate an optimal action a_x^t using the input task embedding z_x^t. The discriminator network D is designed to distinguish between expert action $\hat{a}_x^t$ and the training agent's action a_x^t. The intuition behind this is that the expert actions are assumed to be optimal in the imitation learning setting, thus, G are trained to minimize the difference between $\hat{a}_x^t$ and a_x^t. In order to achieve that, the adversarial loss [68] is applied for both networks:

$$\min_G \max_D \mathcal{L}_{GD}(\hat{a}_x^t, a_x^t) = \mathbb{E}[logD(a_x^t)] + \mathbb{E}[log(1 - D(\hat{a}_x^t))] \tag{5.4}$$

The optimal policy is achieved using a RL-based policy gradient method, which relies on reward signal $r = -logD(\hat{a}_x^t)$ provided by the discriminator.

Full Objective

During the source task's learning process, a set of expert-generated demonstrations $\{\tau_S^1, \tau_S^2, ...\}$ is provided where each demonstration is a sequence of state-actions pairs $\tau_S^i = \{(\hat{s}_S^t, \hat{a}_S^t), ...\}$. The task embedding for each demonstration state z_S^t at time step t can be computed using $z_S^t = E(\hat{s}_S^t)$. It should be noted that the contrastive loss

function $\mathcal{L}_E$ used to train E requires two inputs z_x^t and z_y^t, where x and y can be of the same or different task. In this source task learning process, the target task demonstrations are not provided yet, thus, the second task embedding input $z_S'^t$ is generated by introducing the Gaussian noise $\mu \sim \mathcal{N}(0,1)$ to augment $\hat{s}_x^t$ as follows:

$$z_S'^t = E(\hat{s}_S'^t) \tag{5.5}$$

where $\hat{s}_S'^t = \hat{s}_S^t + \mu$. In addition, since $\hat{s}_S'^t$ is an augmentation of $\hat{s}_S^t$, it might not belong to the state space $\mathcal{S}_S$ of the source task. Thus, the resulting $z_S'^t$ is not used as an input to G to generate an action, but it is used to help compute the loss $\mathcal{L}_E$ only. This means that $z_S'^t$ can be treated as a constant. In other words, the gradient flows back from $z_S'^t$ is unnecessary in the backpropagation. This can be indicated using the stop-gradient operation $stopgrad(\cdot)$ as follows [124], [125]:

$$z_S'^t = stopgrad(E(\hat{s}_S'^t)) \tag{5.6}$$

With the generated action $a_S^t = G(z_S^t)$, the full objective function to train the proposed agent on the source task is

$$\min_{E,G} \max_{D} \mathcal{L} = \mathcal{L}_E(z_S^t, z_S'^t) + \mathcal{L}_{GD}(\hat{a}_S^t, a_S^t) \tag{5.7}$$

The algorithm to train the proposed agent on the source task is outlined in Algorithm 4.

Algorithm 4 Training the proposed agent on the source task.

Input
  $\{\tau_S^1, \tau_S^2, ...\}$ A set of expert demonstrations on the source task
Randomly initialize task embedding network E, generator G and discriminator D
for k = 0, 1, 2, ... **do**
  Sample an expert demonstration τ_S^i
  Sample state-action pairs $(\hat{s}_S^t, \hat{a}_S^t) \sim \tau_S^i$
  Compute $z_S^t = E(\hat{s}_S^t)$
  Compute $z_S'^t = stopgrad(E(\hat{s}_S^t + \mu))$
  Generate action $a_S^t = G(z_S^t)$
  Compute the loss $\mathcal{L} = \mathcal{L}_E(z_S^t, z_S'^t) + \mathcal{L}_{GD}(\hat{a}_S^t, a_S^t)$
  Update the parameters of F, G, and D
  Update policy π_S with the reward signal $r = -logD(\hat{a}_S^t)$
end for
Output
  π_S Learned policy for source task

5.3.2 The Proposed Task Adaptation Algorithm

Leveraging the task embedding space learned by the proposed agent, a novel adaptation algorithm is presented in order to adapt the agent to a new target task by adding the knowledge of the target task to the task-embedding space as shown in Figure 5. In addition, to prevent losing the previously learned knowledge to perform the source task, a novel idea based on repetition learning is applied in the proposed adaptation algorithm. The idea can be illustrated as shown in Figure 5.3. The intuition behind this idea is that during the adaptation process, the agent is allowed to repeatedly review how to perform the previously learned source task while learning the target task. Each time the agent switches to a different task, its performance drops, but then it recovers. This distinctive learning process allows the agent to continuously review its learned knowledge and generalize to both source and target tasks, resulting in an agent that can perform well on both tasks. It is similar to humans; when humans repeatedly practice an action, it leads to better performance. In addition, the process enables the agent to surpass the performance of an agent that is adapted using transfer learning. As shown in Figure 5.3, using transfer learning, the adapted agent

completes its adaptation process right after adapting the source task to the target task. For this reason, when facing the source task again after adaptation, the performance of the agent deteriorates due to the catastrophic forgetting problem.

It is important to note that, theoretically, the more knowledge the agent gains, the higher performance the agent can provide on both source and target tasks. As shown in Figure 5.3, after facing the source task again, the performance of the agent on the source task increases. However, in practice, there is still an amount of performance deterioration on the source task since the agent is not able to fully utilize the learned knowledge. This observation is further discussed in the evaluation and discussion sections.

A hyperparameter $\lambda \in [0, 1]$ is introduced, which denotes the probability that the agent repeatedly reviews the source task's knowledge. With λ, the balance between the performance on the target task and the performance deterioration on the source task can be controlled. For instance, the higher the value of λ, the higher the probability that the agent can review the previously learned source task, resulting in a smaller deterioration of the source task's performance in exchange for low performance on the target task. It should be noted that if $\lambda \approx 0$, the proposed task adaptation algorithm can be seen as a transfer learning method where it is only focused on improving the target task's performance. The task adaptation algorithm is outlined in Algorithm 5.

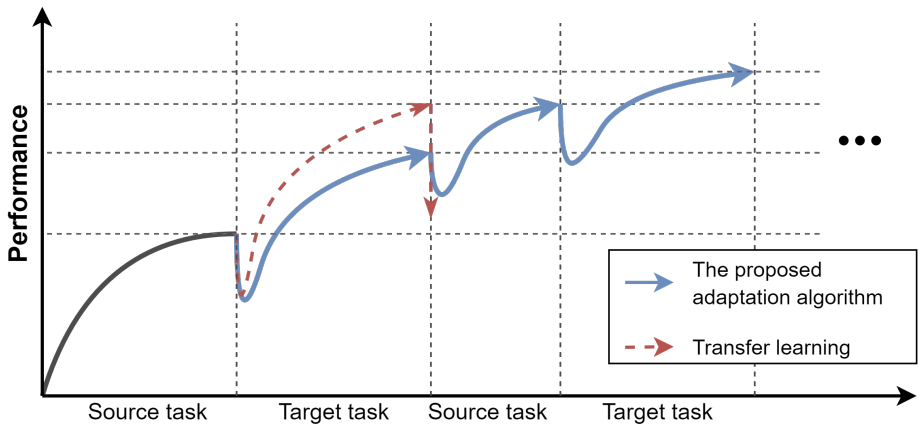


Figure 5.3. An illustration of the performance of an agent on the source and target tasks over adaptation time.

Algorithm 5 The proposed adaptation algorithm.

Input
  $\{\tau_T^1, \tau_T^2, ...\}$ A set of expert demonstrations on the target task
  $\{\tau_S^1, \tau_S^2, ...\}$ A set of expert demonstrations on the source task
Randomly initialize task embedding network E, generator G and discriminator D
for k = 0, 1, 2, ... **do**
  Sample an expert demonstration on the target task τ_T^i
  Sample an expert demonstration on the source task τ_S^i
  Sample state-action pairs $(\hat{s}_S^t, \hat{a}_S^t) \sim \tau_S^i$ and $(\hat{s}_T^t, \hat{a}_T^t) \sim \tau_T^i$
  $n \leftarrow$ uniform random number between 0 and 1
  if $n < \lambda$ **then** ▷ Review source task's learned knowledge
    Compute $z_S^t = E(\hat{s}_S^t)$
    Compute $z_T^t = stopgrad(E(\hat{s}_T^t))$
    Generate action $a_S^t = G(z_S^t)$
    Compute the loss $\mathcal{L} = \mathcal{L}_E(z_S^t, z_T^t) + \mathcal{L}_{GD}(\hat{a}_S^t, a_S^t)$
  else ▷ Learn target task
    Compute $z_T^t = E(\hat{s}_T^t)$
    Compute $z_S^t = stopgrad(E(\hat{s}_S^t))$
    Generate action $a_T^t = G(z_T^t)$
    Compute the loss $\mathcal{L} = \mathcal{L}_E(z_T^t, z_S^t) + \mathcal{L}_{GD}(\hat{a}_T^t, a_T^t)$
  end if
  Update the parameters of F, G, and D
  Update policy π_S with the reward signal $r = -logD(\hat{a}_S^t)$
end for
Output
  π_{ST} Learned policy for both source and target task

5.4 Performance Evaluation

In this section, the performance of the proposed method is evaluated in comparison with baselines. To support the evaluation, different simulated tasks with varying difficulty levels ranging from simple to complex ones were utilized. The details of

these tasks are described in the next subsection. A set of experiments are designed in order to answer the following essential questions:

- Can the proposed IL agent provide a competitive performance on the source task?
- Can the adaptation algorithm enable the agent to adapt its learned knowledge to the target task in order to outperform the baselines?
- By leveraging the repetition learning to expand the agent's knowledge, can the adaptation algorithm reduce the deterioration of the agent's performance on the source task?

5.4.1 Experimental Settings

Simulated Tasks

In order to examine the effectiveness of the proposed method, six simulated tasks with varying difficulties were considered: Pendulum [113], CartPole [113], [114], WindowOpen [115], WindowClose [115], Door [116], and Hammer [116]. The task difficulty is varied along two axes; the size of the state space and the size of the action space. The detailed descriptions and visualizations of these tasks are shown in Table 5.1 and Figure 5.4. From such tasks, three experiments were conducted, each of which included two different tasks—a source task and a target task. The detailed descriptions of these experiments are shown in Table 5.2.

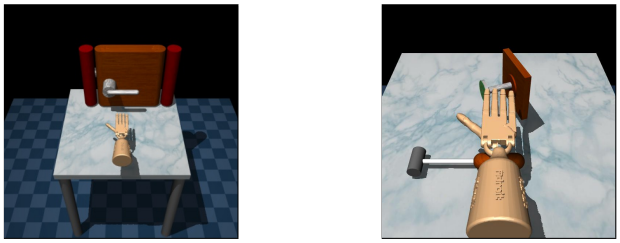
Table 5.1. Description of six simulated tasks used in the experiment.

Task	Size of State Space	Size of Action Space	Difficulty Level	Description
Pendulum [113]	3 (continuous)	1 (continuous)	Easy	Swinging up a pendulum.
CartPole [113], [114]	4 (continuous)	1 (continuous)	Easy	Preventing the pendulum from falling over by applying a force to the cart.
WindowOpen [115]	39 (continuous)	4 (continuous)	Medium	Opening a window.
WindowClose [115]	39 (continuous)	4 (continuous)	Medium	Closing a window.
Door [116]	39 (continuous)	28 (continuous)	Hard	A 24-DoF hand attempts to undo the latch and swing the door open.
Hammer [116]	46 (continuous)	26 (continuous)	Hard	A 24-DoF hand attempts to use a hammer to drive the nail into the board.

Table 5.2. Description of three experiments conducted to evaluate the performance of the proposed method.

Experiment	**Source Task**	**Target Task**	**Difficulty Level**	**Description**
Pendulum–CartPole	Pendulum	CartPole	Easy	A simple experiment in which both source and target tasks have small state and action spaces.
WindowOpen–WindowClose	WindowOpen	WindowClose	Medium	Both source and target tasks have a large state space but small action space.
Door–Hammer	Door	Human	Hard	A challenging experiment in which both source and target tasks have large state and action spaces.

(a) Pendulum **(b)** CartPole **(c)** WindowOpen **(d)** WindowClose



(e) Door **(f)** Hammer

Figure 5.4. Visual rendering of five simulated tasks used in the experiment.

In order to train and adapt the proposed IL agent, expert demonstrations for both source and target tasks must be provided. In this experiment, the proximal policy optimization (PPO) method was chosen to be trained on each task in order to create an expert RL agent. The reason behind this decision was that PPO was recently showing the best result for many complex tasks. After that, the demonstrations were collected by executing the trained PPO expert agent in the simulated task under different configurations. For the source task, 30 demonstrations were collected to provide sufficient data for training the proposed agent [68]. In the adaptation process, the proposed agent already learned the knowledge of the source task, thus, a smaller number of demonstrations for the target task is required. Therefore, only 15 demonstrations were collected for the target task.

Baselines

To evaluate the performance of the proposed method, a number of baselines were considered. Firstly, to assess the performance of the proposed agent on a source task, two RL baselines were used, which are proximal policy optimization (PPO) [117] and neural fitted Q-iteration (NFQI) [118]. PPO is a policy gradient method, while NFQI is a value-based method that tries to estimate the Q-function using a deep feed-forward network. Secondly, after training the agent on the source task,

the proposed adaptation algorithm was applied in order to adapt the trained agent to a new target task. The performance of the agent after adaptation was evaluated through the comparison with transfer learning-based baselines, which are fine-tuning and TA-TL [119]. Fine-tuning is a common transfer learning technique that simply re-trains the agent on a new target task. Fine-tuning was applied to both the proposed agent and PPO, resulting in two baselines for the evaluation. Meanwhile, TA-TL is a policy adaptation method, where first it utilizes the NFQI agent to find an optimal policy on a source task, then that policy is transferred to a new target task. In order to provide a fair comparison, each baseline was evaluated for 100 trials. The success rate and average cumulative reward were used as performance metrics. The success rate indicates the percentage of trials in which the baseline can successfully complete a task. The average cumulative reward measures how well the baseline performed in a trial.

Implementation and Training Details

In order to perform the experiments, a personal computer running Ubuntu 20.04 with an Intel i7-8750H @ 2.20GHz, 16 GB RAM, and NVIDIA GTX 1080 Ti was used. PyTorch [126] and Tianshou [127] were utilized as deep learning frameworks to implement the proposed adaptation method and baselines. Adam optimizer with an initial learning rate of 10^{-4} was used for training the proposed agent. The dimension n of the task embedding z_x^t and the value of λ were set to 64 and 0.1, respectively.

5.4.2 Results

In this subsection, the evaluation results of the proposed agent and adaptation algorithm are presented to highlight their effectiveness in tackling the domain and task adaptation problem in imitation learning.

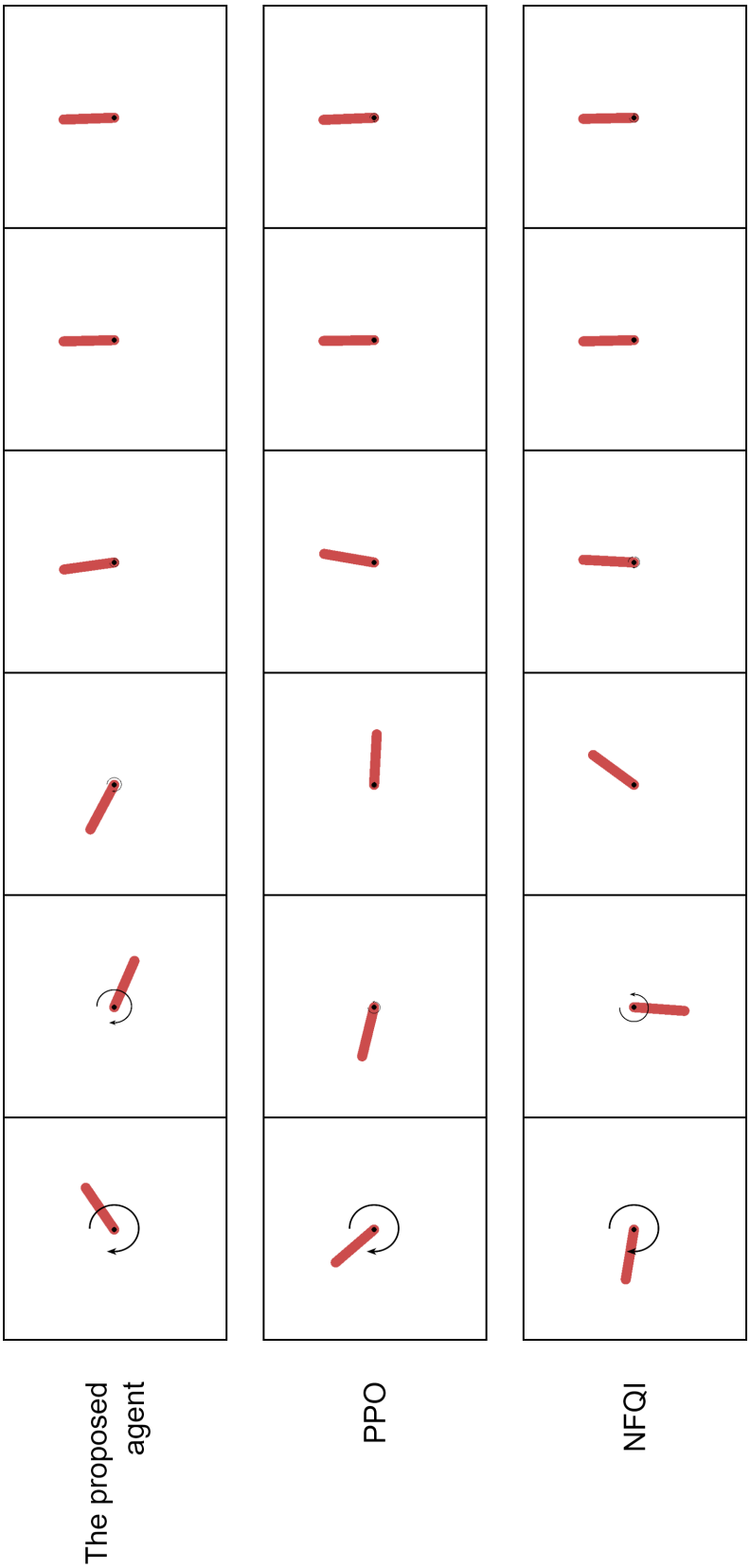
Performance of the Proposed Agent on the Source Task

Table 5.3 reports the performance of the proposed agent on the source tasks (i.e., Pendulum, WindowOpen, and Door) against two RL baselines: PPO and NFQI. In addition, Figure 5.5 visualizes their behaviors when performing the source tasks.

It can be observed that the proposed agent and two baselines could accomplish source tasks by keeping the pendulum vertical (Figure 5.5a), successfully opening the window and the door (Figure 5.5b,c). The proposed imitation learning agent was able to produce relatively similar behaviors to PPO. This result demonstrated that the proposed agent was trained successfully in order to imitate the expert behaviors. Table 5.3 shows that PPO always provided the best performance in terms of success rate and average cumulative reward on three different source tasks. This result was reasonable since PPO is a reinforcement learning method, thus, it has a direct access to the task environment, including states and the reward signal. On the other hand, the proposed agent is an imitation learning method that learns to perform the task using only expert demonstrations. Despite that disadvantage, the proposed agent could consistently perform well on all source tasks with varying difficulties and almost achieved similarly high performance to PPO. It should be noted that the performance of all agents always decreased when being tested on a more complicated task with more extensive state and action spaces, especially the Door task. However, the reduction in performance between the proposed agent and PPO was comparable. On the other hand, there was a significant gap between the proposed agent and the NFQI performance. The NFQI agent showed the largest reduction in terms of success rate, i.e., from 100% success rate on the simple Pendulum task to only 65% on the challenging Door task. This was because the Q-function approximation in NFQI did not work well with the task with large state and action spaces [118]. In summary, the results showed that the proposed agent could provide relatively high and consistent performance that is close to the expert PPO on different source tasks with various difficulty levels.

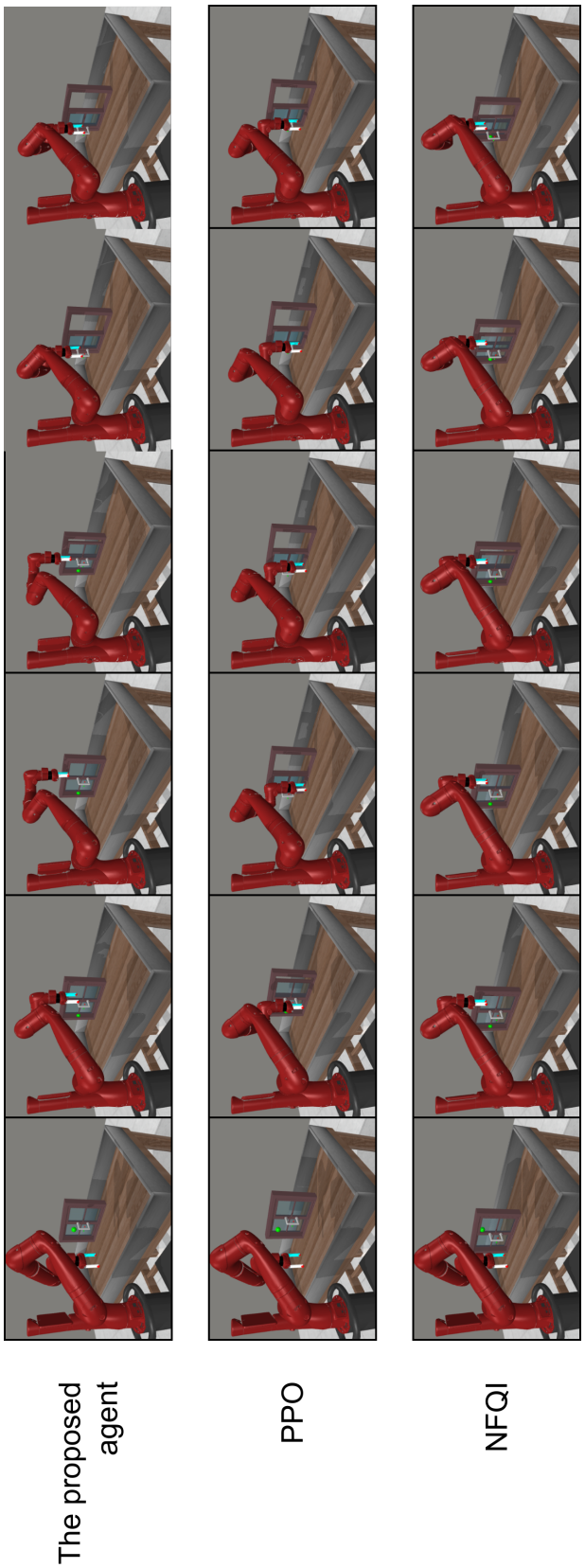
Table 5.3. The performance of the proposed agent on source tasks.

		Pendulum	**WindowOpen**	**Door**
Success rate	DTAIL–GAN	100%	94%	87%
	PPO [117]	100%	97%	91%
	NFQI [118]	100%	76%	65%
Average cumulative reward	DTAIL–GAN	-146.51 ± 85.24	1586.38 ± 229.00	2250.04 ± 1428.60
	PPO [117]	-134.77 ± 93.59	1827.56 ± 410.98	2450.42 ± 1303.48
	NFQI [118]	-189.01 ± 87.09	752.00 ± 476.77	1252.55 ± 1213.15



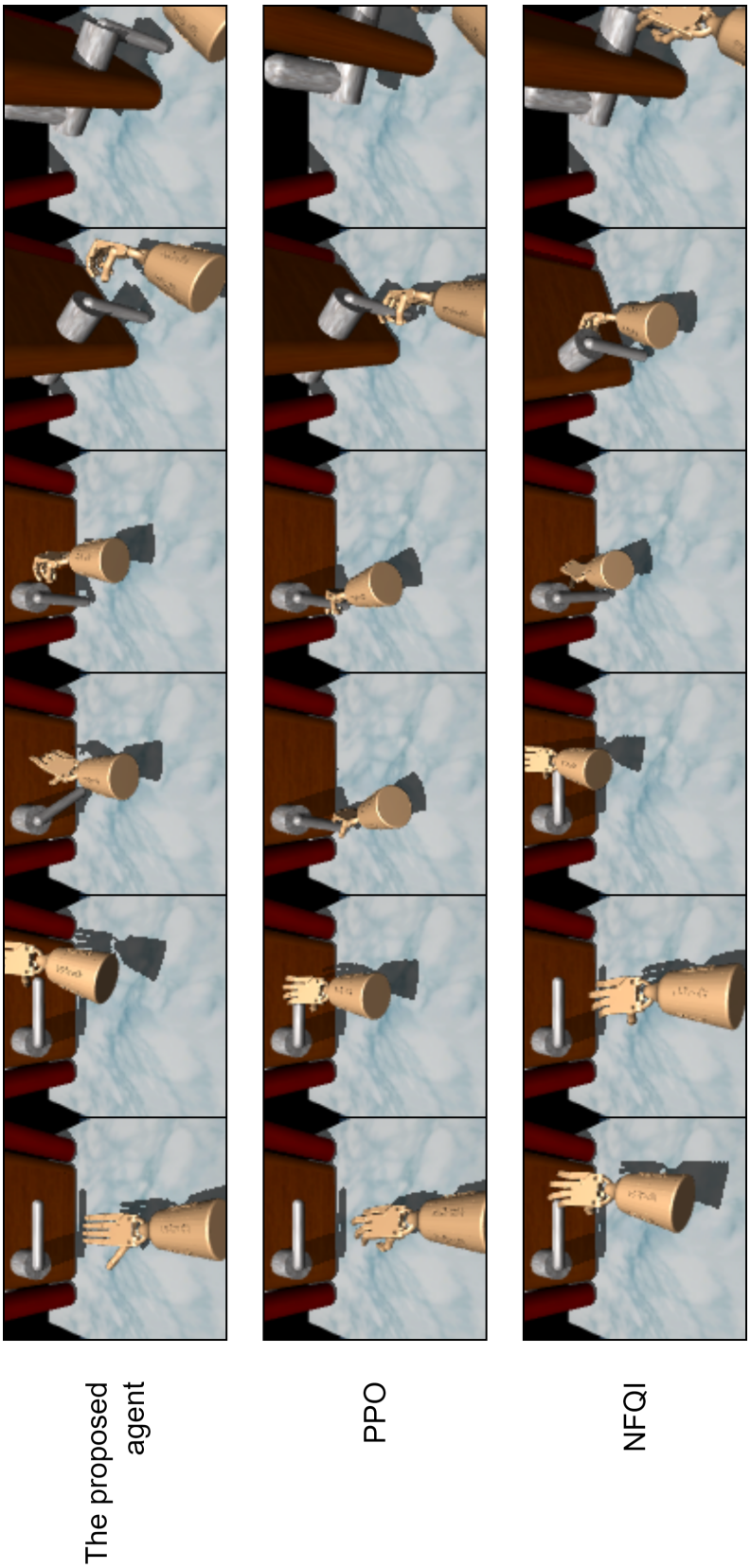
(a) Pendulum

Figure 5.5. *Cont.*



(b) WindowOpen

Figure 5.5. *Cont.*



(c) Door

Figure 5.5. A visualization of the behavior of the proposed agent and baselines on source tasks.

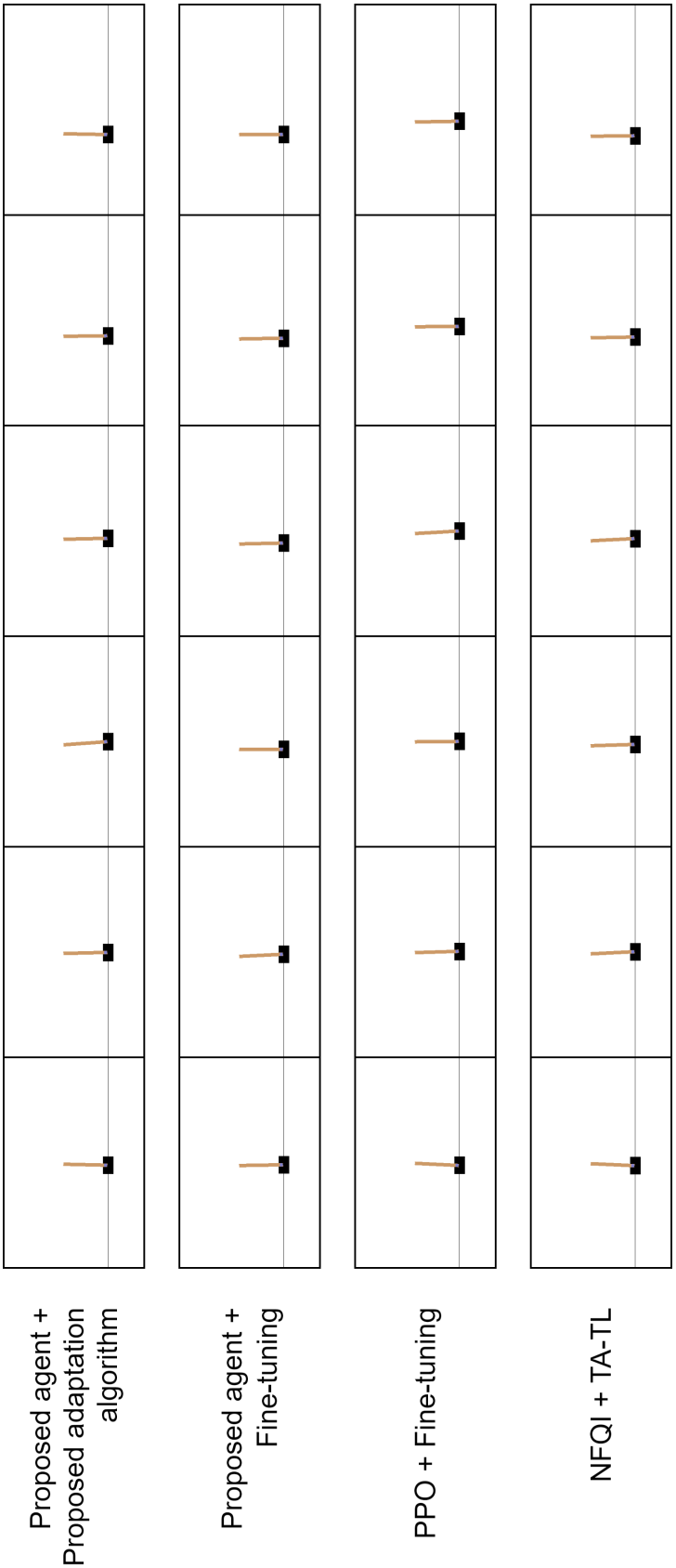
Performance of the Proposed Agent on the Target Task after Adaptation

All agents trained on the source task were adapted to the target task in order to evaluate the performance of the proposed adaptation algorithm in comparison with other transfer learning baselines. The result is tabulated in Table 5.4. The behavior of those agents when performing target tasks is visualized in Figure 5.6. It can be seen that the proposed adaptation method and baselines provide comparably similar behaviors in order to solve target tasks. This result indicated that the proposed method successfully adapted and transferred the agent's knowledge to the new target task. Moreover, it can be observed from Table 5.4 that the proposed method, which is a two-fold method, including the proposed agent and the adaptation algorithm, outperformed other transfer learning-based baselines. In addition, it performed highly well and consistently on the complex WindowClose and Hammer tasks. On the other hand, applying fine tuning to the proposed agent led to a significant reduction in the adapted agent's performance, especially on the complex Hammer task which achieved only a 50% success rate. Moreover, its performance was the lowest compared to other transfer learning baselines. This indicated that the trained agent on the source task (i.e., Door) failed to transfer its learned knowledge to the target task (i.e., Hammer). The reason could be because the adapted agent using fine tuning failed to learn state and action mappings from the source to the target task due to the size of the state and action spaces of those two tasks being different as shown in Table 5.1. This observation indicates that fine tuning was not suitable for the proposed agent. On the other hand, applying fine tuning to the PPO agent provided a consistent performance across all three tasks. At the same time, applying TA-TL to the NFQI agent was not able to produce a high success rate due to the high complexity of the WindowClose and Hammer tasks.

The results demonstrated that the proposed method not only outperformed baselines in terms of success rate on all target tasks, but notably produced a consistently high performance, even on the most difficult task. This proved the potential of the proposed method in order to tackle the domain and task adaptation problem in imitation learning. However, it should be noted that the research objective is not only to achieve high performance on the target task, but also to avoid the performance deterioration on the source task. Therefore, the performance of the adapted agent on source tasks will be assessed next in order to evaluate the decline of the agent's performance after adaptation.

Table 5.4. The performance of the proposed agent on target tasks after adaptation.

		CartPole	**WindowClose**	**Hammer**
Success rate	DTAIL–GAN+ Proposed adaptation algorithm	100%	83%	82%
	DTAIL–GAN+ Fine-tuning	77%	72%	50%
	PPO [117] + Fine-tuning	87%	80%	77%
	NFQI + TA-TL [119]	80%	63%	67%
Average cumulative reward	DTAIL–GAN+ Proposed adaptation algorithm	500.00 ± 0.0	2340.59 ± 642.69	$13,137.42 \pm 2709.57$
	DTAIL–GAN+ Fine-tuning	433.44 ± 86.52	1513.07 ± 566.09	1741.76 ± 1035.17
	PPO [117] + Fine-tuning	487.63 ± 32.74	2215.98 ± 608.33	3022.64 ± 1115.92
	NFQI + TA-TL [119]	476.63 ± 61.84	1447.53 ± 641.16	2591.46 ± 1231.70



(a) CartPole

Figure 5.6. *Cont.*

(b) WindowClose

Figure 5.6. *Cont.*

(c) Hammer

Figure 5.6. A visualization of the behavior of the proposed agent and baselines on target tasks.

Performance of the Proposed Agent on the Source Task after Adaptation

Table 5.5 shows the deterioration in success rate of the adapted agent on source tasks compared to the one before the adaptation. The lower value of the deterioration illustrates a better result. It can be observed that as the difficulty level of the target task increased, the deterioration became more notable. In addition, three baselines were not able to maintain high performance on the source task. Even on the simple Pendulum task, the deterioration was extremely high compared to the proposed adaptation algorithm. This was due to the fact that those transfer learning baselines were designed to optimize the performance of the agent only on the target task. Thus, it was obvious that the performance of those adapted agents dropped significantly on the source task. On the other hand, the deterioration of the proposed method was the lowest compared to other baselines, which indicated that the proposed adaptation algorithm successfully retained the learned knowledge from the source tasks and reduced the negative effect of catastrophic forgetting.

Table 5.5. The performance of the proposed agent on source tasks after adaptation. These scores represent the deterioration in success rate compared to the one before the adaptation.

	Pendulum	**WindowOpen**	**Door**
DTAIL–GAN+ Proposed adaptation algorithm	18%	32%	44%
DTAIL–GAN+ Fine-Tuning	41%	73%	74%
PPO [117] + Fine-tuning	32%	58%	83%
NFQI + TA-TL [119]	24%	62%	51%

Computational Complexity

Besides evaluating the performance of the proposed task adaptation method in terms of success rate, its computational cost was also assessed in order to provide an adequate study of its overall performance. Table 5.6 shows the training time required to adapt a trained agent to a new target task in each experiment. It can be observed that the training time of the proposed adaptation method was slightly better than the training time when applying fine tuning to PPO, especially on two complex WindowOpen-WindowClose and Door–Hammer experiments. On the other hand, compared to TA-TL, the proposed adaptation method required a higher training time on all three experiments. This result was expected since, during the proposed adaptation process, the agent had to not only learn the new task, but also review the previously learned source task. However, it should be noted that the training time of the proposed adaptation method can be further improved by leveraging the parallel training process [127], [128].

Table 5.6. The training time (s/epoch) of the proposed task adaptation algorithm.

	Pendulum–CartPole	**WindowOpen–WindowClose**	**Door–Hammer**
DTAIL–GAN+ Proposed adaptation algorithm	87.051	163.768	503.19
DTAIL–GAN+ Fine-tuning	74.680	114.290	321.87
PPO [117] + Fine-tuning	86.801	184.472	557.416
NFQI + TA-TL [119]	58.499	121.510	352.53

5.5 Discussion

In this section, the effects of applying repetition learning on the performance of the proposed method and the important role of the task embedding network E are discussed in detail.

The experimental results assessed in the previous section have shown the potential of the proposed adaptation method in tackling the task adaptation problem in imitation learning. As shown in Tables 5.3 and 5.4, the proposed method could provide consistent and high performance in terms of success rate and average cumulative reward on both source and target tasks with varying difficulty levels. This indicates that the proposed method can be applied to more challenging tasks with larger state and action spaces. Moreover, Table 5.5 shows that the performance deterioration on the source task was also reduced compared to transfer learning baselines. This promising result demonstrates the effectiveness of the proposed adaptation method, in which the idea of repetition learning was leveraged in order to allow the agent to review the previously learned source task. Although the success rate and training time remained limited, the proposed method presents a plausible approach to tackle the task adaptation problem in imitation learning. It can be further improved in order to provide better overall performance toward practical imitation learning tasks.

In order to support the idea of repetition learning, an imitation learning agent was proposed, which was able to encode its learned knowledge into a task-embedding space. To provide an ablation study of the task embedding network E in the proposed agent, a small experiment was conducted, where a number of task embeddings z_S^t and z_T^t were collected by executing the adapted agent in the WindowOpen–WindowClose experiment on both source task (i.e., WindowOpen) and target task (i.e., WindowClose). The WindowOpen–WindowClose was chosen because both source and target tasks are similar and have a large and equal size of the state space, which can provide a sufficient ablation result. In each task, the adapted agent was run in the simulation over 100 trials. After that, t-distributed stochastic neighbor embedding (t-SNE) was applied in order to project the collected high-dimensional task embeddings to a two-dimensional space for visualization as shown in Figure 5.7. t-SNE captures the distance relation between task embeddings. If two embeddings were close in the task-embedding space, they stay close in the resulting visualization, and vice versa. Therefore, from Figure 5.7, it can be seen that task embeddings of the

source and target tasks were well separated. Moreover, Figure 5.7 also shows that some target task embeddings were mixed with the source task embeddings. This was expected since the WindowOpen and WindowClose tasks shared the same structure (i.e., robot hand and window), thus, these target task embeddings represented the shared knowledge between the source and target tasks. This result indicates that the proposed adaptation method not only successfully expands the task embedding space without forgetting the previously learned knowledge, but also leverages the source task's knowledge in order to accelerate and adapt to the new target task. This leads to high performance on the target task shown in Table 5.4 and a low performance deterioration on the source task shown in Table 5.5.

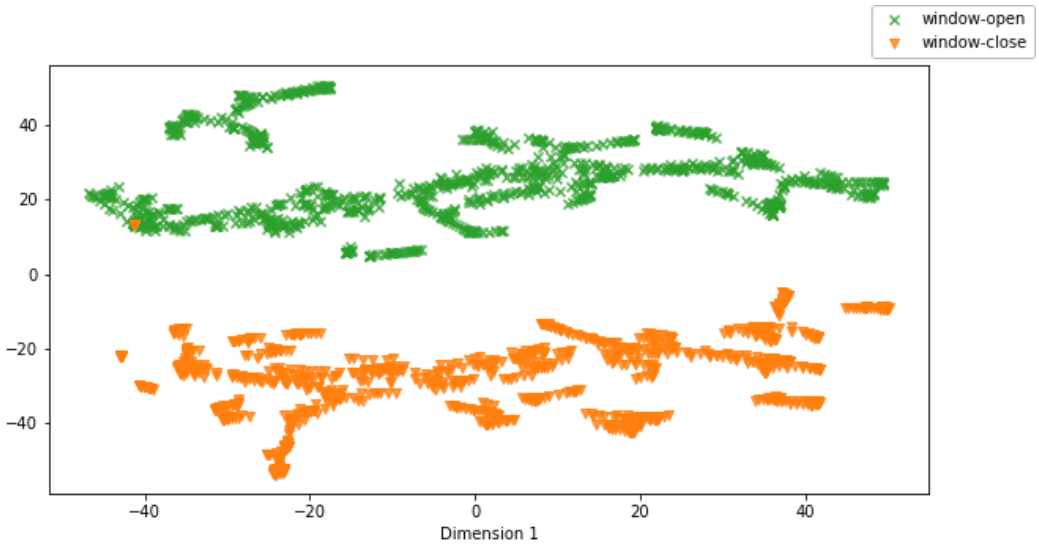


Figure 5.7. Visualization of clustering results on task embedding vectors z_S^t and z_T^t. Different colors mark different tasks.

Although the novel idea of applying repetition learning and encoding the task knowledge into a task embedding has significantly improved the adapted agent on both tasks, there is still one limitation. As shown in Figure 5.3, ideally, the adapted agent should be able to perform both source and target tasks better over time and eventually surpass its performance on the source task before being adapted. However, as shown in the experimental results, there was an amount of deterioration in the source task's performance, thus, the proposed method is still limited compared to human learning ability. Overcoming this problem can be served as a key step toward building a continual learning agent, where the agent can learn and adapt to not only one but multiple target tasks. In future work, this will be the main focus of

the authors in order to provide a general-purpose agent that can become a better learner over time, i.e., learning new tasks better and faster, and performing better on previously learned tasks.

5.6 Summary

In this chapter, a novel domain and task adaptation method for imitation learning was proposed. The proposed adaptation method leverages the idea of repetition learning in neuroscience allowing the agent to repeatedly review the previously learned source task while learning a new target task. The experimental results on simulated tasks with varying difficulties show that the proposed method is able to consistently provide high performance on the target task and minimizes the deterioration of the source task's performance. Moreover, it demonstrates the effectiveness of the proposed method compared to transfer learning in enabling the agent to expand its knowledge without forgetting the knowledge learned from the source task, resulting in an adapted agent that is able to perform well on both tasks. Despite some limitations in the success rate and computational cost, the results indicate the potential of the proposed method to be applied in practical imitation learning tasks.

Chapter 6

Discussion

This chapter discusses the potential of adversarial learning in improving imitation learning generalization. In addition, it also summarized the advantages and remaining issues of the proposed DAIL–GAN, TAIL–GAN, and DTAIL–GAN agents.

In Chapter 3, the adversarial learning process allowed the DAIL–GAN agent to extract domain-shared and domain-specific features for adapting its knowledge from the expert domain to the learner domain. The experiment results showed that it could provide high performance on low-dimensional tasks but was unable to complete high-dimensional tasks. Then, a similar learning process is applied to train the TAIL–GAN agent by learning the similarities and differences between source and target tasks. The results revealed that TAIL–GAN can accomplish low- and high-dimensional tasks with a relatively high success rate. This difference is because adversarial learning relies on the discriminator's learning performance. When training DAIL–GAN on high-dimensional tasks, the differences between the expert and learner domains are minor, making it challenging for the discriminator to distinguish both domains at the beginning of the training process. This results in poor domain-specific feature extraction and reduces the agent's performance on high-dimensional tasks. On the other hand, the differences between source and target tasks when training TAIL–GAN are significant. Thus, the discriminator can efficiently extract the task-specific features resulting in a high agent's performance on low- and high-dimensional tasks.

Despite that, the results of DAIL–GAN, TAIL–GAN agents have proved the capability and flexibility of applying adversarial learning to training imitation learning agents. With the same objective function, adversarial learning can help the agents extract

different features depending on the training data. For instance, in DAIL–GAN agent, adversarial learning enables it to extract domain-shared and domain-specific features. On the other hand, in TAIL–GAN agent, the similarity and differences between source and target tasks are extracted. Moreover, in Chapter 5, a more challenging problem was considered, in which a trained agent on a source task faces a new target task and must optimize its overall performance on both tasks. The DTAIL–GAN agent was proposed to address this problem. Although its performance remains limited, the promising results of DTAIL–GAN agent also highlight the potential of adversarial learning in developing a more general-purpose agent.

The main limitation of the proposed agents is that they were only evaluated in simulated environments. Thus, their performance in real-world situations still needs to be discovered. It would be beneficial to evaluate them in physical robots in order to verify if they can maintain high performance in the complex physics of the real world.

Chapter 7

Conclusion

This dissertation has presented novel methods for improving generalization in imitation learning. This chapter concludes the thesis with a final summary of the key insights and future research possibilities.

7.1 Conclusion

Generalizing between domains and tasks remains a challenging problem for imitation learning agents. Although an IL agent can solve a given task under a specific domain, they struggle to transfer their knowledge to new, unseen situations or environments. This dissertation attempts to address this shortcoming by utilizing adversarial learning in order to allow the agent to extract domain- and task-related features during its learning process. These features helped the agent to be able to transfer and adapt its learned knowledge from one source task to a new target task across different domains.

In Chapter 3, DAIL–GAN was first introduced to overcome the domain adaptation problem in imitation learning. The agent was able to learn the domain-shared and domain-specific features to adapt the learned knowledge from the expert domain to a distinct learner domain. It provided a high success rate on low-dimensional tasks without being affected by the presence of domain shift between expert and learner domains. Although its performance highly depended on the complexity of the tasks,

DAIL–GAN had proved the effectiveness of applying adversarial learning in imitation learning.

In Chapter 4, the task adaptation problem was then focused on. The TAIL–GAN agent was trained under a similar adversarial learning process compared to DAIL–GAN̈The experimental results demonstrated that TAIL–GAN can achieve high performance in a number of low-dimensional tasks.

Finally, in Chapter 5, in order to further exploit the potential of adversarial learning, the DTAIL–GAN agent and an adaptation method were introduced to address both domain and adaptation problems under a more challenging setting in which the adapted agent has to perform consistently well on both source and target tasks. The proposed adaptation method consisted of two phases, including training and adaptation processes. The first training process allowed the agent to capture the underlying structure of the task and encode these features into a high-dimensional latent space. The second adaptation process utilized adversarial learning to expand the learned knowledge by adding the knowledge of the target task. Moreover, leveraging the idea of repetition learning in neuroscience, the agent was allowed to repeatedly review the previously learned source task while learning a new target task. A comprehensive evaluation over several simulated tasks with varying difficulty levels showed that the proposed method could provide high and consistent performance on both source and target tasks across different domains.

In summary, the most significant contribution of this dissertation was demonstrating the effectiveness of adversarial learning in improving imitation learning generalization. From the experimental results, it could be observed that applying adversarial learning in the imitation learning agent's learning process can significantly improve generalization, allow the agent to adapt its learned behaviors, and produce a higher performance in unseen situations. Despite some limitations in the performance of the proposed agents, the results indicated their potential to be applied in practical imitation learning tasks. To conclude, while much work remains, this dissertation provides several contributions toward generalizable imitation learning, which can accelerate the widespread adoption of imitation learning in the real-world environment.

7.2 Future Research Possibilities

This section outlines several directions for future research opportunities based on this dissertation.

Non-episodic MDPs As discussed in Chapter 2, all proposed agents in this dissertation are designed for the episodic MDPs setting. For other important non-episodic settings, such as infinite time horizon MDPs, the performance of those agents is unknown. It would be beneficial to investigate to what extent the proposed agents developed in this dissertation can be applied to this setting.

Sim-to-Real Transfer A limitation of this dissertation is that all the experiments were conducted in simulated environments. It would be interesting to observe the performance of the proposed agents on real robots. Moreover, in a sim-to-real transfer setting, the simulation and the physical robot can be considered as two different domains. Therefore, the adaptation algorithm proposed in Chapter 5 can be applied in which the agent is first trained in simulation and then adapted to the real environment. In other words, the physical robot can go through the same adaptation process as the DTAIL–GAN agent but learn to use the physics and dynamics of the real world. This approach could provide a very useful tool to make imitation learning agents more practical.

Neuroscience By leveraging the idea of repetition learning in neuroscience, the DTAIL–GAN agent has achieved high performance on both source and target tasks. Moreover, allowing the agent to repeatedly review its knowledge while learning a new task reduced the deterioration of the learning performance on the source task while ensuring that the learning performance on the target task is high. The result has proved the potential of applying neuroscience ideas in training an imitation learning agent. It is, therefore, an interesting challenge to leverage other neuroscience ideas, such as forgetting curve, recency effect, etc., to develop agents with better generalization. For example, the forgetting curve characteristic is a natural process, describing the exponential loss of memory over time [129]. It can be formulated as the following equation [129]:

$$R = e^{-\frac{t}{S}}$$

where R is memory retention, S is the relative strength of memory, and t is the time. The proposed adaptation algorithm in Chapter 5 uses probability to decide, at time step t, whether the agent needs to review its learned knowledge on the source task or continue learning the new target task. Instead, using the above equation, the algorithm can decide based on the memory retention of the source task's knowledge. This approach may allow the agent to reduce repetition times while maintaining the same performance on both source and target tasks. This could be an interesting idea to reduce the adaptation time and provide better feature learning.

Lifelong (Continual) learning The DTAIL–GAN agent is able to generalize the knowledge learned from a source task to a new target task and perform well on both tasks. A logical next step is to adapt the learned agent to not only one but multiple novel target tasks, which is the goal of lifelong learning. However, there are several challenges that need to be considered, such as maintaining performance on previously learned tasks while learning new tasks, reusing past knowledge to speed up learning new tasks, and improving performance on previously learned tasks from newly acquired knowledge. Existing approaches in lifelong learning can be combined with the adaptation method proposed in Chapter 5 in order to address those challenges while improving the agent's generalization. It is exciting to develop such a method to provide a general-purpose agent that can become a better learner over time, i.e., learning new tasks better and faster, and performing better on previously learned tasks.

References

[1] S. Raje, N. Reddy, H. Jerbi, P. Randhawa, G. Tsaramirsis, N. V. Shrivas, A. Pavlopoulou, M. Stojmenović, and D. Piromalis, "Applications of Healthcare Robots in Combating the COVID-19 Pandemic," eng, *Applied Bionics and Biomechanics*, vol. 2021, p. 7 099 510, 2021, ISSN: 1176-2322. DOI: 10.1155/2021/7099510.

[2] *How Robots Became Essential Workers in the COVID-19 Response - IEEE Spectrum*, en. [Online]. Available: https://spectrum.ieee.org/how-robots-became-essential-workers-in-the-covid19-response (visited on 03/01/2023).

[3] H. Jiang and L. Cheng, "Public Perception and Reception of Robotic Applications in Public Health Emergencies Based on a Questionnaire Survey Conducted during COVID-19," en, *International Journal of Environmental Research and Public Health*, vol. 18, no. 20, p. 10 908, Jan. 2021, ISSN: 1660-4601. DOI: 10.3390/ijerph182010908. [Online]. Available: https://www.mdpi.com/1660-4601/18/20/10908.

[4] M. Javaid, A. Haleem, A. Vaish, R. Vaishya, and K. P. Iyengar, "Robotics Applications in COVID-19: A Review," *Journal of Industrial Integration and Management*, vol. 05, no. 04, pp. 441–451, Dec. 2020, ISSN: 2424-8622. DOI: 10.1142/S2424862220300033. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/S2424862220300033.

[5] Y. Shen, D. Guo, F. Long, L. A. Mateos, H. Ding, Z. Xiu, R. B. Hellman, A. King, S. Chen, C. Zhang, and H. Tan, "Robots Under COVID-19 Pandemic: A Comprehensive Survey," eng, *IEEE access: practical innovations, open solutions*, vol. 9, pp. 1590–1615, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3045792.

[6] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018, ISBN: 9780262039246.

[7] T. Nguyen Duc, C. M. Tran, P. X. Tan, and E. Kamioka, "Domain Adaptation for Imitation Learning Using Generative Adversarial Network," en, *Sensors*, vol. 21, no. 14, p. 4718, Jan. 2021, ISSN: 1424-8220. DOI: 10.3390/s21144718. [Online]. Available: https://www.mdpi.com/1424-8220/21/14/4718.

[8] T. Nguyen Duc, C. M. Tran, N. G. Bach, P. X. Tan, and E. Kamioka, "Repetition-Based Approach for Task Adaptation in Imitation Learning," en, *Sensors*, vol. 22, no. 18, p. 6959, Jan. 2022, ISSN: 1424-8220. DOI: 10.3390/s22186959. [Online]. Available: https://www.mdpi.com/1424-8220/22/18/6959.

[9] R. Bellman, "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957, ISSN: 0095-9057. [Online]. Available: https://www.jstor.org/stable/24900506 (visited on 03/14/2023).

[10] D. Ha and J. Schmidhuber, "Recurrent World Models Facilitate Policy Evolution," in *Advances in Neural Information Processing Systems*, vol. 31, Curran Associates, Inc., 2018. [Online]. Available: https://papers.nips.cc/paper/2018/hash/2de5d16682c3c35007e4e92982f1a2ba-Abstract.html (visited on 02/16/2023).

[11] S. Racanière, T. Weber, D. P. Reichert, L. Buesing, A. Guez, D. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, R. Pascanu, P. Battaglia, D. Hassabis, D. Silver, and D. Wierstra, "Imagination-augmented agents for deep reinforcement learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 5694–5705, ISBN: 9781510860964. (visited on 02/16/2023).

[12] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural Network Dynamics for Model-Based Deep Reinforcement Learning with Model-Free Fine-Tuning," arXiv, Tech. Rep., Dec. 2017, arXiv:1708.02596 [cs] type: article. DOI: 10.48550/arXiv.1708.02596. [Online]. Available: http://arxiv.org/abs/1708.02596 (visited on 02/16/2023).

[13] V. Feinberg, A. Wan, I. Stoica, M. I. Jordan, J. E. Gonzalez, and S. Levine, "Model-Based Value Estimation for Efficient Model-Free Reinforcement Learning," arXiv, Tech. Rep., Feb. 2018, arXiv:1803.00101 [cs, stat] type: article. DOI: 10.48550/arXiv.1803.00101. [Online]. Available: http://arxiv.org/abs/1803.00101 (visited on 02/16/2023).

[14] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," arXiv, Tech. Rep., Dec. 2017, arXiv:1712.01815 [cs] type: article. DOI: 10.48550/arXiv.1712.01815. [Online]. Available: http://arxiv.org/abs/1712.01815 (visited on 02/16/2023).

[15] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Harley, T. P. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16, New York, NY, USA: JMLR.org, Jun. 2016, pp. 1928–1937. (visited on 02/16/2023).

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," arXiv, Tech. Rep., Aug. 2017, arXiv:1707.06347 [cs] type: article. DOI: 10.48550/arXiv.1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347 (visited on 02/16/2023).

[17] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15, Lille, France: JMLR.org, Jul. 2015, pp. 1889–1897. (visited on 02/16/2023).

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," arXiv, Tech. Rep., Jul. 2019, arXiv:1509.02971 [cs, stat] type: article. DOI: 10.48550/arXiv.1509.02971. [Online]. Available: http://arxiv.org/abs/1509.02971 (visited on 02/16/2023).

[19] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," en, PMLR, Jul. 2018, pp. 1861–1870. [Online]. Available: https://proceedings.mlr.press/v80/haarnoja18b.html (visited on 02/16/2023).

[20] S. Fujimoto, H. Hoof, and D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods," en, PMLR, Jul. 2018, pp. 1587–1596. [Online]. Available: https://proceedings.mlr.press/v80/fujimoto18a.html (visited on 02/16/2023).

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," arXiv, Tech. Rep., Dec. 2013, arXiv:1312.5602 [cs] type: article. DOI: 10.48550/arXiv.1312.5602. [Online]. Available: http://arxiv.org/abs/1312.5602 (visited on 02/16/2023).

[22] M. G. Bellemare, W. Dabney, and R. Munos, "A Distributional Perspective on Reinforcement Learning," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17, Sydney, NSW, Australia: JMLR.org, Aug. 2017, pp. 449–458. (visited on 02/16/2023).

[23] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI'18/IAAI'18/EAAI'18, New Orleans, Louisiana, USA: AAAI Press, Feb. 2018, pp. 2892–2901, ISBN: 9781577358008. (visited on 02/16/2023).

[24] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight Experience Replay," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://papers.nips.cc/paper/2017/hash/453fadbd8a1a3af50a9df4df899537b5-Abstract.html (visited on 02/16/2023).

[25] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *arXiv*, Jun. 2014. arXiv: 1406.2661. [Online]. Available: http://arxiv.org/abs/1406.2661.

[26] H. KWON, Y. KIM, H. YOON, and D. CHOI, "Captcha image generation systems using generative adversarial networks," *IEICE Transactions on Infor-*

mation and Systems, vol. E101.D, no. 2, pp. 543–546, 2018. DOI: 10.1587/transinf.2017EDL8175.

[27] J. Song, T. He, L. Gao, X. Xu, A. Hanjalic, and H. T. Shen, "Unified binary generative adversarial network for image retrieval and compression," *International Journal of Computer Vision*, pp. 1–22, 2020.

[28] H. Chen, Y. Wang, H. Shu, C. Wen, C. Xu, B. Shi, C. Xu, and C. Xu, "Distilling portable generative adversarial networks for image translation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 3585–3592.

[29] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[30] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*, PMLR, 2018, pp. 734–743.

[31] M. Q. Mohammed, K. L. Chung, and C. S. Chyi, "Review of deep reinforcement learning-based object grasping: Techniques, open challenges, and recommendations," *IEEE Access*, vol. 8, pp. 178 450–178 481, 2020.

[32] R. Li, A. Jabri, T. Darrell, and P. Agrawal, "Towards practical multi-object manipulation using relational reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 4051–4058.

[33] H. Han, G. Paul, and T. Matsubara, "Model-based reinforcement learning approach for deformable linear object manipulation," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, IEEE, 2017, pp. 750–755.

[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[35] A. Jeerige, D. Bein, and A. Verma, "Comparison of deep reinforcement learning approaches for intelligent game playing," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, IEEE, 2019, pp. 0366–0371.

[36] D. Silver, R. S. Sutton, and M. Müller, "Reinforcement learning of local shape in the game of go.," in *IJCAI*, vol. 7, 2007, pp. 1053–1058.

[37] D. Ye, G. Chen, W. Zhang, S. Chen, B. Yuan, B. Liu, J. Chen, Z. Liu, F. Qiu, H. Yu, *et al.*, "Towards playing full moba games with deep reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 621–632, 2020.

[38] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[39] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[40] B. Osiński, A. Jakubowski, P. Zięcina, P. Miłoś, C. Galias, S. Homoceanu, and H. Michalewski, "Simulation-based reinforcement learning for real-world autonomous driving," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 6411–6418.

[41] M. Zhu, Y. Wang, Z. Pu, J. Hu, X. Wang, and R. Ke, "Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102 662, 2020.

[42] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis," *Machine Learning*, vol. 110, no. 9, pp. 2419–2468, 2021.

[43] P. Kormushev, S. Calinon, and D. G. Caldwell, "Reinforcement learning in robotics: Applications and real-world challenges," *Robotics*, vol. 2, no. 3, pp. 122–148, 2013.

[44] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2008.10.024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889008001772.

[45] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

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

[47] 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," *arXiv preprint arXiv:1802.09564*, 2018.

[48] N. Ratliff, J. A. Bagnell, and S. S. Srinivasa, "Imitation learning for locomotion and manipulation," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, IEEE, 2007, pp. 392–397.

[49] J. Chen, B. Yuan, and M. Tomizuka, "Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 2884–2890.

[50] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 4693–4700.

[51] J. Hawke, R. Shen, C. Gurau, S. Sharma, D. Reda, N. Nikolov, P. Mazur, S. Micklethwaite, N. Griffiths, A. Shah, *et al.*, "Urban driving with conditional imitation learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 251–257.

[52] P. M. Kebria, R. Alizadehsani, S. M. Salaken, I. Hossain, A. Khosravi, D. Kabir, A. Koohestani, H. Asadi, S. Nahavandi, E. Tunsel, *et al.*, "Evaluating architecture impacts on deep imitation learning performance for autonomous driving," in *2019 IEEE International Conference on Industrial Technology (ICIT)*, IEEE, 2019, pp. 865–870.

[53] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE and PSYCHOLOGY . . ., Tech. Rep., 1989.

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

[55] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, "Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 3758–3765.

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

[57] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning.," in *ICML*, vol. 1, 2000, p. 2.

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

[59] S. Levine, Z. Popovic, and V. Koltun, "Nonlinear inverse reinforcement learning with gaussian processes," *Advances in neural information processing systems*, vol. 24, pp. 19–27, 2011.

[60] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning.," in *Aaai*, Chicago, IL, USA, vol. 8, 2008, pp. 1433–1438.

[61] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*, PMLR, 2016, pp. 49–58.

[62] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 1331–1336.

[63] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, p. 103 500, 2021.

[64] M. Naumann, L. Sun, W. Zhan, and M. Tomizuka, "Analyzing the suitability of cost functions for explaining and imitating human driving behavior based on inverse reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 5481–5487.

[65] Z. Bing, C. Lemke, L. Cheng, K. Huang, and A. Knoll, "Energy-efficient and damage-recovery slithering gait design for a snake-like robot based on reinforcement learning and inverse reinforcement learning," *Neural Networks*, vol. 129, pp. 323–333, 2020.

[66] G. J. Zelinsky, Y. Chen, S. Ahn, H. Adeli, Z. Yang, L. Huang, D. Samaras, and M. Hoai, "Predicting goal-directed attention control using inverse-reinforcement learning," *arXiv preprint arXiv:2001.11921*, 2020.

[67] M. Qi, Y. Li, A. Wu, Q. Jia, B. Li, W. Sun, Z. Dai, X. Lu, L. Zhou, X. Deng, *et al.*, "Multi-sequence mr image-based synthetic ct generation using a generative adversarial network for head and neck mri-only radiotherapy," *Medical physics*, vol. 47, no. 4, pp. 1880–1894, 2020.

[68] J. Ho and S. Ermon, "Generative Adversarial Imitation Learning," in *Advances in Neural Information Processing Systems*, vol. 29, Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html.

[69] N. Baram, O. Anschel, I. Caspi, and S. Mannor, "End-to-end differentiable adversarial imitation learning," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 390–399. [Online]. Available: http://proceedings.mlr.press/v70/baram17a.html.

[70] F. Behbahani, K. Shiarlis, X. Chen, V. Kurin, S. Kasewa, C. Stirbu, J. Gomes, S. Paul, F. A. Oliehoek, J. Messias, *et al.*, "Learning from demonstration in the wild," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 775–781.

[71] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, pp. 3815–3825, ISBN: 9781510860964.

[72] X. Zhang, Y. Li, X. Zhou, and J. Luo, "Cgail: Conditional generative adversarial imitation learning—an application in taxi drivers' strategy learning," *IEEE Transactions on Big Data*, 2020.

[73] W. Chi, G. Dagnino, T. M. Kwok, A. Nguyen, D. Kundrat, M. E. Abdelaziz, C. Riga, C. Bicknell, and G.-Z. Yang, "Collaborative robot-assisted endovascular catheterization with generative adversarial imitation learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 2414–2420.

[74] Y. Zhou, R. Fu, C. Wang, and R. Zhang, "Modeling car-following behaviors and driving styles with generative adversarial imitation learning," *Sensors*, vol. 20, no. 18, p. 5034, 2020.

[75] B. C. Stadie, P. Abbeel, and I. Sutskever, "Third-Person Imitation Learning," in *International Conference on Learning Representations (ICLR)*, Toulon, France, Apr. 2017. arXiv: 1703.01703. [Online]. Available: http://arxiv.org/abs/1703.01703.

[76] K. Kim, Y. Gu, J. Song, S. Zhao, and S. Ermon, "Domain Adaptive Imitation Learning," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, PMLR, Nov. 2020, pp. 5286–5295. [Online]. Available: http://proceedings.mlr.press/v119/kim20c.html.

[77] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, "Transfusion: Understanding transfer learning for medical imaging," *Advances in neural information processing systems*, vol. 32, 2019.

[78] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer.," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[79] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, and S. Singh, "Deep transfer learning based classification model for covid-19 disease," *Irbm*, 2020.

[80] M. F. Aslan, M. F. Unlersen, K. Sabanci, and A. Durdu, "Cnn-based transfer learning–bilstm network: A novel approach for covid-19 infection detection," *Applied Soft Computing*, vol. 98, p. 106 912, 2021.

[81] M. Humayun, R. Sujatha, S. N. Almuayqil, and N. Jhanjhi, "A transfer learning approach with a convolutional neural network for the classification of lung carcinoma," in *Healthcare*, MDPI, vol. 10, 2022, p. 1058.

[82] P. Salza, C. Schwizer, J. Gu, and H. C. Gall, "On the effectiveness of transfer learning for code search," *IEEE Transactions on Software Engineering*, 2022.

[83] M. Sharma, K. Nath, R. K. Sharma, C. J. Kumar, and A. Chaudhary, "Ensemble averaging of transfer learning models for identification of nutritional deficiency in rice plant," *Electronics*, vol. 11, no. 1, p. 148, 2022.

[84] V. Campos, P. Sprechmann, S. S. Hansen, A. Barreto, S. Kapturowski, A. Vitvitskyi, A. P. Badia, and C. Blundell, "Beyond fine-tuning: Transferring behavior in reinforcement learning," in *ICML 2021 Workshop on Unsupervised Reinforcement Learning*, 2021. [Online]. Available: https://openreview.net/forum?id=4NUhTHom2HZ.

[85] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 7559–7566.

[86] R. Julian, B. Swanson, G. Sukhatme, S. Levine, C. Finn, and K. Hausman, "Never stop learning: The effectiveness of fine-tuning in robotic reinforcement learning," in *Proceedings of the 2020 Conference on Robot Learning*, J. Kober, F. Ramos, and C. Tomlin, Eds., ser. Proceedings of Machine Learning Research, vol. 155, PMLR, 2021, pp. 2120–2136. [Online]. Available: https://proceedings.mlr.press/v155/julian21a.html.

[87] P. Mannion, S. Devlin, J. Duggan, and E. Howley, "Reward shaping for knowledge-based multi-objective multi-agent reinforcement learning," *The Knowledge Engineering Review*, vol. 33, e23, 2018. DOI: 10.1017/S0269888918000292.

[88] T. Brys, A. Harutyunyan, M. E. Taylor, and A. Nowé, "Policy transfer using reward shaping," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '15, Istanbul, Turkey: International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 181–188, ISBN: 9781450334136.

[89] S. Doncieux, "Transfer learning for direct policy search: A reward shaping approach," in *2013 IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, 2013, pp. 1–6. DOI: 10.1109/DevLrn.2013.6652568.

[90] M. E. Taylor, P. Stone, and Y. Liu, "Transfer learning via inter-task mappings for temporal difference learning," *J. Mach. Learn. Res.*, vol. 8, pp. 2125–2167, Dec. 2007, ISSN: 1532-4435.

[91] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," *arXiv preprint arXiv:1703.02949*, 2017.

[92] H. B. Ammar, K. Tuyls, M. E. Taylor, K. Driessens, and G. Weiss, "Reinforcement learning transfer via sparse coding," in *Proceedings of the 11th international conference on autonomous agents and multiagent systems*, International Foundation for Autonomous Agents and Multiagent Systems . . ., vol. 1, 2012, pp. 383–390.

[93] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 2169–2176.

[94] M. E. Taylor and P. Stone, "Representation transfer for reinforcement learning.," in *AAAI Fall Symposium: Computational Approaches to Representation Change during Learning and Development*, 2007, pp. 78–85.

[95] A. Zhang, H. Satija, and J. Pineau, "Decoupling dynamics and reward for transfer learning," *arXiv preprint arXiv:1804.10689*, 2018.

[96] N. Vithayathil Varghese and Q. H. Mahmoud, "A survey of multi-task deep reinforcement learning," *Electronics*, vol. 9, no. 9, p. 1363, 2020.

[97] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[98] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2008.10.024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889008001772.

[99] C. L. Baker, J. B. Tenenbaum, and R. R. Saxe, "Goal inference as inverse planning," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 29, 2007.

[100] Y. Bao and R. H. Cuijpers, "On the imitation of goal directed movements of a humanoid robot," *International Journal of Social Robotics*, vol. 9, no. 5, pp. 691–703, 2017.

[101] M. S. Tomov, E. Schulz, and S. J. Gershman, "Multi-task reinforcement learning in humans," *Nature Human Behaviour*, pp. 1–10, 2021.

[102] S. Zhang and R. S. Sutton, "A deeper look at experience replay," *arXiv preprint arXiv:1712.01275*, 2017.

[103] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[104] R. S. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," *Advances in neural information processing systems*, pp. 1038–1044, 1996.

[105] A. Geramifard, C. Dann, R. H. Klein, W. Dabney, and J. P. How, "Rlpy: A value-function-based reinforcement learning framework for education and research.," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1573–1578, 2015.

[106] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE transactions on systems, man, and cybernetics*, no. 5, pp. 834–846, 1983.

[107] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2018. [Online]. Available: https://sites.google.com/view/deeprl-dexterous-manipulation.

[108] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning*, PMLR, 2015, pp. 1889–1897.

[109] V. Kumar and E. Todorov, "Mujoco haptix: A virtual reality system for hand manipulation," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, IEEE, 2015, pp. 657–663.

[110] J. Hua, L. Zeng, G. Li, and Z. Ju, "Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning," *Sensors*, vol. 21, no. 4, p. 1278, 2021.

[111] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: A survey," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2020, pp. 737–744.

[112] Y. Liu, Z. Li, H. Liu, and Z. Kan, "Skill transfer learning for autonomous robots and human–robot cooperation: A survey," *Robotics and Autonomous Systems*, vol. 128, p. 103 515, 2020.

[113] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[114] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE transactions on systems, man, and cybernetics*, no. 5, pp. 834–846, 1983.

[115] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Proceedings of the Conference on Robot Learning*, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., ser. Proceedings of Machine Learning Research, vol. 100, PMLR, 2020, pp. 1094–1100. [Online]. Available: https://proceedings.mlr.press/v100/yu20a.html.

[116] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, Pittsburgh, Pennsylvania, Jun. 2018. DOI: 10.15607/RSS.2018.XIV.049. [Online]. Available: https://sites.google.com/view/deeprl-dexterous-manipulation.

[117] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[118] M. Riedmiller, "Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method," in *European conference on machine learning*, Springer, 2005, pp. 317–328.

[119] "Cross-domain transfer in reinforcement learning using target apprentice," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 7525–7532.

[120] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," in *International Conference on Machine Learning*, PMLR, 2018, pp. 4548–4557.

[121] H. Ebbinghaus, "Memory: A contribution to experimental psychology," *Annals of neurosciences*, vol. 20, no. 4, p. 155, 2013.

[122] L. Zhan, D. Guo, G. Chen, and J. Yang, "Effects of repetition learning on associative recognition over time: Role of the hippocampus and prefrontal cortex," *Frontiers in Human Neuroscience*, vol. 12, 2018, ISSN: 1662-5161. DOI: 10.3389/fnhum.2018.00277. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnhum.2018.00277.

[123] T. Uchihara, S. Webb, and A. Yanagisawa, "The effects of repetition on incidental vocabulary learning: A meta-analysis of correlational studies," *Language Learning*, vol. 69, no. 3, pp. 559–599, 2019.

[124] Y. Tian, X. Chen, and S. Ganguli, "Understanding self-supervised learning dynamics without contrastive pairs," in *International Conference on Machine Learning*, PMLR, 2021, pp. 10 268–10 278.

[125] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[126] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[127] J. Weng, H. Chen, D. Yan, K. You, A. Duburcq, M. Zhang, H. Su, and J. Zhu, "Tianshou: A highly modularized deep reinforcement learning library," *arXiv preprint arXiv:2107.14171*, 2021.

[128] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, 2021.

[129] H. Ebbinghaus, “Memory: A contribution to experimental psychology,” eng, *Annals of Neurosciences*, vol. 20, no. 4, pp. 155–156, Oct. 2013, ISSN: 0972-7531. DOI: 10.5214/ans.0972.7531.200408.

Appendix A

Publication List

International Journals (peer reviewed)

1. **Tho Nguyen Duc**, Chanh Minh Tran, Nguyen Gia Bach, Phan Xuan Tan, Eiji Kamioka, "Repetition-Based Approach for Task Adaptation in Imitation Learning" Sensors, Vol. 22, Issue 18, 6959, 2022, pp.1-19.

2. Nguyen Gia Bach, Chanh Minh Tran, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "Novel projection schemes for graph-based Light Field coding" Sensors, Vol. 22, Issue 13, 4948, 2022, pp.1-28.

3. Chanh Minh Tran, **Tho Nguyen Duc**, Nguyen Gia Bach, Phan Xuan Tan, Eiji Kamioka, "Sprinkle Prebuffer Strategy to Improve Quality of Experience with Less Data Wastage in Short-form Video Streaming" Electronics, Vol. 11, Issue 13, 1949, 2022, pp.1-18.

4. Chanh Tran Minh, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "An Experimental Study of the Server-based Unfairness Solutions for the Cross-Protocol Scenario of Adaptive Streaming over HTTP/3 and HTTP/2" Baghdad Science Journal (BSJ), Vol. 18, No. 4, 2021, pp.1441-1447.

5. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan, Eiji Kamioka, "Generative Adversarial Network for Imitation Learning from Single Demonstration" Baghdad Science Journal (BSJ), Vol. 18, No. 4, 2021, pp.1350-1355.

6. Chanh Minh Tran, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "Cross-protocol Unfairness between Adaptive Streaming Clients over HTTP/3 and HTTP/2: A Root-Cause Analysis" Electronics, Vol. 10, Issue 15, 1755, 2021, pp.1-14.

7. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan, Eiji Kamioka, "Domain Adaptation for Imitation Learning using Generative Adversarial Network" Sensors, Vol. 21, Issue 14, 4718, 2021, pp.1-14.

8. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan, Eiji Kamioka, "Convolutional Neural Networks for Continuous QoE Prediction in Video Streaming Services" IEEE Access, Vol. 8, 2020, pp.116268-116278.

9. Chanh Minh Tran, **Tho Nguyen Duc**, Phan Xuan Tan and Eiji Kamioka, "FAURAS: A Proxy-based Framework for Ensuring the Fairness of Adaptive Video Streaming over HTTP/2 Server Push" Applied Science, Vol. 10, Issue 7, 2485, 2020, pp.1-21.

10. Chanh Minh Tran, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "QABR: A QoE-Based Approach to Adaptive Bitrate Selection in Video Streaming Services" International Journal of Advanced Trend in Computer Science and Engineering, Vol. 8, No. 1.4, 2019, pp.138-144.

11. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan, Eiji Kamioka, "Modeling of Cumulative QoE in On-demand Video Services: Role of Memory Effect and Degree of Interest" Future Internet, Vol.11, Issue 8, 171, 2019, pp.1-18.

International Conferences (peer reviewed)

1. Chanh Minh Tran, **Tho Nguyen Duc**, Nguyen Gia Bach, Phan Xuan Tan, Eiji Kamioka, "Reducing QoE Loss and Data Wastage of Short-Form Video Streaming with HTTP/3 Prioritization" Proceedings of the 12th International Conference on Informatics, Environment, Energy and Applications 2023 (IEEA2023), Singapore, February 17-19, 2023 (Accepted).

2. Long Doan, **Tho Nguyen Duc**, Chuanzhe Jing, Eiji Kamioka, Phan Xuan Tan, "Automatic Keyword Extraction for Viewport Prediction of 360-degree Virtual

Tourism Video" Proceedings of the 2nd IEEE International Conference on Computing 2022 (ICOCO2022), Kota, Kinabalu, Sabah, Malaysia, November 14-16, 2022, pp.386-391.

3. Chuanzhe Jing, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "Subtitle-based Viewport Prediction for 360-degree Virtual Tourism Video, " Proceedings of the 13th International Conference on Information, Intelligence, Systems and Applications 2022 (IISA2022), Corfu, Greece, July 18-20, 2022, pp.1-8.

4. Phan Xuan Tan, **Tho Nguyen Duc**, Chanh Minh Tran, Eiji Kamioka, "Continuous QoE Prediction Based on WaveNet" Proceedings of the 12th International Conference on Computer and Automation Engineering (ICCAE2020), Sydney, Australia, February 14-16, 2020, pp.80-84.

5. Yu Wang, Chanh Minh Tran, **Tho Nguyen Duc**, Xiaochun Wu, Phan Xuan Tan, Eiji Kamioka, "An Experimental Study on the Unfairness in Adaptive Streaming with HTTP/2 Server Push" Proceedings of International Conference on Video, Signal and Image Processing 2019 (VSIP2019), Wuhan, China, October 29-31, 2019, pp.94-98.

6. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan and Eiji Kamioka, "Bidirectional LSTM for Continuously Predicting QoE in HTTP Adaptive Streaming" Proceedings of the 2nd International Conference on Information Science and System (ICISS2019), Tokyo, Japan, March 16-19, 2019, pp.156-160.

Domestic Conference (non-reviewed)

1. **Tho Nguyen Duc**, Chanh Minh Tran, Phan Xuan Tan, Eiji Kamioka, "Imitation Learning: Learning Simple Tasks from a Single Demonstration using Generative Adversarial Network" IEICE Technical Report, Vol.121, No.181, CNR2021-6, September 21, 2021, pp.12-15.

2. Chanh Minh Tran, **Tho Nguyen Duc**, Phan Xuan Tan, Eiji Kamioka, "Cross-Protocol Performance Investigation between HTTP/3 and HTTP/2: An Experiment with Adaptive Streaming" IEICE Technical Report, Vol.121, No.167, IA2021-23, September 8, 2021, pp.54-58.