Learning Representations for Efficient Exploration and Goal-Conditioned Reinforcement Learning

by

Jongwook Choi

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2024

Doctoral Committee:

Professor Honglak Lee, Chair
Professor Satinder Singh Baveja
Assistant Professor Nima Fazeli
Professor Benjamin Kuipers

Jongwook Choi
jwook@umich.edu
ORCID iD: 0009-0001-0210-0684

DEDICATION

To my wife and family.

Also, to everyone who has supported me along the way.

ACKNOWLEDGMENTS

This dissertation and my PhD would not have been possible without the support and encouragement from all who have been part of this journey. I am thankful to too many people, beyond words.

First and foremost, I am deeply and sincerely grateful to my advisor, Professor Honglak Lee. It was a great honor to be one of your students and to have had such incredible opportunities to work with you. You have always guided me with support, respect, and patience. I would also like to express my sincere thanks to my dissertation committee members, Professor Benjamin Kuipers, Professor Satinder Singh, and Professor Nima Fazeli, for your valuable feedback, time commitment, and help with my dissertation. In particular, I am very grateful to Professor Benjamin Kuipers for providing me with thoughtful, comprehensive feedback and profound insights that aided me in enhancing my dissertation.

I also wish to extend my thanks to many mentors I've met along my doctoral study journey and during my internship at Google. I thank Professor Gunhee Kim for his help during my undergraduate, which enabled me to begin my graduate studies at the best institution, and thank Professor Joseph Lim for having me as a visiting student, which significantly helped me kickstart my research career. I feel so grateful to Shane Gu, Sergey Levene, Vikash Kumar, Mohammad Norouzi, and Aleksandra Faust for their advice and guidance during my wonderful internship and collaboration with Google, and providing me with exciting and invaluable opportunities.

My sincere thanks also goes to my collaborators at many places: Seunghoon Hong, Junhyuk Oh, Yijie Guo, Marcin Moczulski, Neal Wu, Archit Sharma, Hyunjae Woo, Sungryull Sohn, Lisa Lee, Michael Ahn, Ofir Nachum, Chris Hoang, Wilka Carvalho, Sungtae Lee, Seohong Park, Jaekyeom Kim, Izzedin Gur, Natasha Jaques, Yingjie Miao, Xinyu Wang, Anthony Liu, Violet Fu, and Dong-Ki Kim. I could never have achieved many great works without their efforts and support. My special thanks go to Sungryull Sohn and Sungtae Lee, who have continuously collaborated with me and engaged in in-depth discussions for such a long time. I greatly enjoyed the fun in research we had together and deeply appreciate all the help.

I am so grateful to all our group members I have met throughout my time at the University of Michigan: Yuting Zhang, Seunghoon Hong, Junhyuk Oh, Xinchen Yan, Ruben Villegas, Lajanugen Logeswaran, Kibok Lee, Sungryull Sohn, Yijie Guo, Yunseok Jang, Wilka Carvalho, Kimin Lee, Anthony Liu, Violet Fu, Tiange Luo, Muhammad Khalifa, and Yiwei Lyu. I will never forget the exciting and wonderful times we had together as collaborators and friends.

In addition, I am grateful to some of my friends in Ann Arbor and elsewhere in the US who shared joys and hardships of my graduate school life together: Yunseok Jang, Sungryull Sohn, Juyong Kim, Wonjoon Goo, Hyunjae Woo, Youngwoon Lee, Shao-Hua Sun, Taehoon Kim, Sewon Min, Mina Lee, Zeyu Zheng, Vivek Veeriah, Lisa Lee, GS Oh, Sungmin Oh, Jae-Won Chung, and Santiago Castro.

I would also like to express my appreciation to many staffs and amazing people in the University of Michigan who have supported me in numerous ways. In addition, I am very grateful to KFAS (Korea Foundation for Advanced Studies) for supporting me both materially and spiritually.

Lastly, but not least, I would like to express my deepest gratitude and thanks to my wife, Sooah, for her unwavering support, understanding, encouragement, and love. Without her being by my side during the most arduous and challenging times, I wouldn't have been able to complete my doctoral study. Finally, I deeply thank my mother, father, sister, mother-in-law, and father-in-law for their continuous, unconditional love and support.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

LIST OF ALGORITHMS

ALGORITHM

LIST OF APPENDICES

APPENDIX

ABSTRACT

Deep reinforcement learning (RL) is a general-purpose computational framework for learning sequential decision-making agents, with the promise that agents can learn useful behaviors and solve the task through trial-and-error by maximizing rewards. One key fundamental problem in deep RL is representation learning — discovering and extracting useful or task-relevant information from raw data (e.g., observations and agent's actions) that can make solving downstream tasks more efficient and tractable. In this dissertation, I propose and discuss several methods and principles pertaining to representation learning for RL, with a focus on state and temporal abstraction, enabling more efficient exploration, skill discovery, and learning of goal-conditioned policies for hierarchical agents.

First, I begin with a self-supervised approach to learn a state representation using the idea of contingency-awareness: the agent's knowledge about which aspect of the environment is controllable. I present a novel attentive dynamics model that identifies controllable elements of the environment which can efficiently abstract the search space for exploration, and show that it enables strong exploration performance in difficult, hard-exploration Atari game environments featuring sparse rewards. Next, I discuss a novel perspective and foundation that unifies goal-conditioned RL and variational empowerment methods for unsupervised skill discovery based on the principle of mutual information maximization into a single family of methods. The proposed framework, variational goal-conditioned RL, allows us to interpret variational empowerment methods as a principled approach for learning latent goal representations and goal-reaching reward functions, while also enabling practical techniques and improvements brought from each other. Finally, I present another instance of skill learning for temporal abstraction: entity-centric skill learning in continuous control environments with multiple entities. By utilizing a structured goal representation and a novel intrinsic reward based on counterfactual reasoning and dynamics models, I demonstrate that one can learn pairwise object interaction behaviors without relying on any external rewards. Overall, this dissertation contributes to the advancement of deep RL by addressing state representation learning and skill learning problems, which can help build more autonomous systems for real-world problems with less human supervision.

CHAPTER 1

Introduction

Reinforcement learning (RL) provides a general-purpose computational framework to learn an agent that can make sequential decisions to maximize cumulative rewards, through trial-and-error and the agent's own experiences by interacting with the environment. Deep reinforcement learning (Deep RL) makes use of deep neural networks as non-linear function approximators to parametrize policy and value functions, enabling the handling of complex raw observations such as images and raw sensory information, which cannot be easily dealt with by tabular RL methods. As such, one promise of deep RL is its ability to automatically learn useful behaviors from the agent's experience involving complex and high-dimensional observations, when trained in an end-to-end manner to maximize reward, without requiring extensive supervision such as manually extracting high-level features or the use of human domain-specific knowledge.

Despite the great progress and advances in deep RL and successful applications in the past few years (Mnih et al., 2015; Silver et al., 2017; Andrychowicz et al., 2020), training an intelligent RL agent for real-world problems still faces many practical challenges and difficulties. In many real-world problems and applications, tasks can be *long-horizon* and provide only *sparse reward* signals — the agent would receive meaningful (non-zero) feedback only after taking an appropriate sequence of actions over an extended period of time. In such settings, agents will struggle to learn any meaningful behaviors or representations, as the lack of intermediate rewards makes it extremely difficult for the agent to effectively adapt the behavior from the environment's feedback.

One key fundamental problem in deep RL is **representation learning**. In general, representation learning refers to the process of discovering and extracting useful information from raw data (such as states, observations, or agent's own experience including actions) to form a new ontology of learned representations that can be beneficial for solving downstream tasks or improving the learning of agents. In many RL problems, the state space of the environment can be highly complex, and involve high-dimensional data such as *pixels* in image observations or a robot's proprioceptive and sensory information without knowing which dimensions are important. Representation learning helps to compress or transform such complex data into a more compact space with succinct, low-dimensional representations that are easier to handle. Ideally, learned representations should

capture the most essential features of the environment or the high-level information in the agent's state that is directly relevant to decision-making, so as to facilitate tractable and efficient solutions for downstream RL problems such as prediction, exploration, planning, and learning hierarchical agents.

1.1 Representation Learning as Abstraction

Learning representations in RL is essentially the process of **abstraction**. Abstraction refers to the process of simplifying complex systems by focusing only on relevant aspects while hiding unnecessary details (Giunchiglia et al., 1997; Abel, 2022). Abstraction is a fundamental and foundational concept that permeates various disciplines, including human cognitive process, developmental psychology, programming, computer science, and artificial intelligence. One can hypothesize that an intelligent agent, by filtering out irrelevant details and focusing solely on essential parts only, can concentrate on the core elements that are crucial for solving a problem. Indeed, cognitive and intelligent agents (such as humans) are capable of doing abstractions at multiple levels for problem solving. In the context of deep RL and decision-making AI systems, abstraction via representation learning can be categorized into two broad categories: *state abstraction* and *temporal abstraction*.

State Abstraction. On the one hand, **state abstraction** involves simplifying or distilling complex states or observations into more compact state representations, or identifying which aspect of the environment will be relevant and important, allowing the agent to ignore task-irrelevant details (Singh et al., 1994; Li et al., 2006; Abel et al., 2016). This process often leads to learning a new level of ontology or hierarchy of features and information about the world, such as recognizing entities and objects (Modayil & Kuipers, 2007, 2008), and understanding their properties and relationships, going beyond describing the world using already known concepts (e.g., pixel-level ontology like pixel-wise classification or semantic segmentation, or creating a cognitive map or simple re-description of the world where the constituting elements and concepts are already known).

For example, consider an ATARI game called FREEWAY (Figure 2.1), where the player controls a chicken sprite attempting to cross multiple lanes of traffic without being hit by vehicles. When the agent needs to decide its next action, the location of the chicken is likely the most important and crucial information, while nearby objects (moving vehicles) are somewhat important, and other background objects are barely important. Also, consider a robotics agent navigating a complex labyrinth as another example. The agent's intelligence spans multiple hierarchies: at a low-level, the agent needs to handle low-level controls, such as dodging nearby pitfalls and precisely managing motor torques for locomotion and stable posture; at a high-level, the agent will need to focus more on longer-term planning or behavioral decisions, such as determining velocity and direction, or

navigation goals towards finding the optimal path so as to maximize the given rewards. However, during high-level planning, low-level control of motors would be less important details that the agent would not need to be concerned with immediately, as they mostly are a computational burden. Given a raw observation, if such a compact high-level state representation can be learned, agents can effectively utilize this information for guiding their decisions or even directly solving the problem within the new representation space. These approaches would reduce computational complexity and make the learning process more efficient.

Temporal Abstraction. On the other hand, **temporal abstraction** (action abstraction) concerns building and composing higher-level actions that consist of multiple low-level actions spanning multiple time steps (Precup, 2000). This enables building *hierarchical* agents capable of strategic and semantic decision-making against long-horizon tasks, in contrast to *flat* agents making decisions on which actions to take at every time step, which can be myopic and inefficient for learning long-term behaviors (Parr & Russell, 1997; Vezhnevets et al., 2017; Nachum et al., 2019).

In the literature, learning of higher-level actions abstracted over time is formalized with the framework of Options (Precup & Sutton, 2000) or Skills (Konidaris & Barto, 2007), usually implemented in the form of goal-conditioned RL (Kaelbling, 1993a). These frameworks typically involves learning of a goal-conditioned policy $\pi(a|s, g)$, i.e. a policy conditioned on a goal g rather than a flat, single-task policy $\pi(a|s)$. However, there are major challenges and difficult problems involved in goal-conditioned RL: (1) *goal representation learning* — how to choose a proper representation space $\mathcal{G}$ for subgoals (called the *goal space*); and (2) *goal policy learning* — how to efficiently learn the goal-conditioned policies through proper and effective learning signals. A common approach is to implement a goal achievement reward function $r(s, g)$ that a goal-conditioned policy $\pi(a|s, g)$ can maximize. However, reward functions may not be readily available as it depends on the choice of goal space $\mathcal{G}$.

Without goal representation learning, one might have to choose a goal space $\mathcal{G}$ with some domain knowledge, such as choosing the location of the robot (x, y) coordinates. However, this reliance on pre-defined, domain-specific knowledge would limit the general applicability of hierarchical RL. At the other extreme, one might simply choose the goal space $\mathcal{G}$ to be the same as the state representation, but then it still remains unclear how to specify a *good* reward function that can tell whether a goal is achieved or not, as state/goal representations can be high-dimensional (e.g., comparing two images in the pixel space). Even if the goal representation is assumed to be pre-defined and designed by humans, designing proper reward functions would be highly expensive as they require significant effort and extensive tuning by practitioners and domain experts to provide the agent with a learning signal that is rich enough rather than too sparse.

As such, we will need better principles to learn the abstract goal representation space, as well

as practical algorithms and techniques to learn temporally-abstracted behaviors. It is crucial to find a good balance between the ease of learning goal representation and the ease of learning goal-conditioned policies under the representation, which can contribute towards building hierarchical agents that can be genuinely autonomous and intelligent. In this dissertation, I focus on skill learning which aims at learning useful macro-actions (Chapters 3 and 4) while simultaneously learning (intrinsic) reward functions to learn skills, and connect them from the goal representation learning perspective (Chapter 3).

Self-supervised Learning. Ideally, learning representations should be done without much supervision or reliance on (external) task rewards, because task rewards can be sparse, and they solely might not provide enough learning signals. In RL, since it is possible to obtain data from agent's experiences through interaction with the environment (state transitions and actions), an agent can make use of its experience to learn a useful internal knowledge about the environment and/or the task. Dominant approaches for this is to optimize auxiliary learning objectives in addition to the standard RL objective — such as reconstruction of the observation (Jaderberg et al., 2017; Nair et al., 2018; Pong et al., 2019), learning forward or inverse dynamics models (Oh et al., 2015; Pathak et al., 2017; Burda et al., 2019a; Choi et al., 2019), maximizing mutual information (Mohamed & Rezende, 2015; Eysenbach et al., 2019; Hjelm et al., 2019; Choi et al., 2021; Zhao et al., 2021), etc. This learning paradigm is called as *self-supervised learning* (Gui et al., 2023), since a RL agent can generate its own learning signals or supervisory signals from its interaction with the environment, rather than relying on external supervision, rewards, or predefined labels.

1.2 Thesis Outline and Contributions

The goal of this dissertation is to discuss, propose, and study techniques and principles that can advance the problem of learning representations in RL. In particular, I study and propose methods in the field of learning state and temporal abstractions that can make it easier to tackle sparse-reward, long-horizon RL tasks, through exploration in a setting where extrinsic supervision (i.e., task reward) is limited.

More specifically, Chapter 2 discusses a self-supervised approach to discover and identify task-relevant information out of high-dimensional input and observations (e.g., images) for efficient exploration, as an instance of state representation learning. Chapter 3 discusses present a novel unification of unsupervised skill discovery methods and goal-conditioned RL as a principled framework for learning goal representations in goal-conditioned and hierarchical RL, as an instance of state and temporal representation learning. Chapter 4 discusses an intrinsic reward approach based on counterfactual reasoning to learn object-object interaction skills, as temporal abstraction against

entity-centric environments. Chapter 5 summarizes the contribution of this dissertation and discusses future research directions.

Chapter 2: Contingency-Aware Exploration in Reinforcement Learning

The first chapter of the dissertation considers state representation learning for exploration to deal with sparse-reward, long-horizon RL problems (Choi et al., 2019). We adopt the idea of *contingency-awareness* and to learn and identify *controllable aspects of an environment* in a self-supervised setting, which can lead to efficient exploration in sparse-reward settings.Specifically, we develop a novel attentive dynamics model (ADM) that discovers controllable aspects of the environment (*contingent regions*). This forms a new object-level ontology that concisely represents the identification and localization of a controllable entity within a visual observation, without the need for discovering and extracting full object/entity information from the scene. We then use this contingency-awareness information as a part of the state representation for exploration purposes, such as in combination with count-based exploration. We empirically show that learning such information about controllable dynamics can help achieve strong exploration performance in sparse-reward, long-horizon settings *without* external annotations or supervision. This approach allowed to achieve state-of-the-art performance on notoriously difficult hard-exploration ATARI environments including Montezuma's Revenge (Choi et al., 2019; Guo et al., 2020).

Chapter 3: Variational Empowerment as Representation Learning for Goal-Conditioned Reinforcement Learning

The next chapter of the dissertation considers representation learning for goal-conditioned RL, as an instance of temporal abstraction and skill discovery (Choi et al., 2021). As introduced in Section 1.1, one important problem for goal-conditioned RL is how to learn a good representation for *goals* (as macro-actions) and a goal-conditioned policy to fulfill the goal without external supervision or domain-specific knowledge (e.g., specifying the x, y locations in robotics control).

In this work, we present a novel foundation and perspective that *unifies* variational empowerment RL methods for unsupervised skill discovery methods (Mohamed & Rezende, 2015; Eysenbach et al., 2019), and classical goal-conditioned RL (GCRL) (Kaelbling, 1993a) into a single family of methods sharing the unified optimization objective. Variational empowerment methods are based on the MI-max (maximizing mutual information) principle (Appendix B.1), or more specifically, aim at maximizing the mutual information between the latent representation or the goal space and the state the agent reaches by executing the goal. Starting from a simple observation that the objective of the standard GCRL can be seen as a special case of variational MI with a fixed hard-coded variational posterior distribution, We can view empowerment-based RL methods as a principled

representation learning framework for goal-conditioned RL. The framework provides a principle way to *represent latent goals* and to *learn a reward function* for measuring goal fulfillment, in a challenging unsupervised RL setting where external rewards are lacking.

Chapter 4: Unsupervised Object Interaction Learning with Counterfactual Dynamics Models

The next chapter of the dissertation studies entity-centric unsupervised skill discovery focusing on object interaction, as an instance of temporal abstraction and improving exploration via intrinsic motivation (Choi et al., 2024).

Real-world tasks often consist of multiple *entities*, or objects: for example, household robots should use household items, and factory robots should manipulate multiple objects and tools to accomplish complex tasks. In the literature, there are many advances in learning object-centric representations from visual inputs and leveraging them to facilitate more efficient reinforcement learning, but still many of the existing approaches often rely on learning monolithic functions or representations that would produce a single high-dimensional vector for representing sub-goals. For example, most of the prior skill discovery and option learning methods (Eysenbach et al., 2019; Choi et al., 2021; Park et al., 2022) rely on monolithic goal representation, which makes it challenging to specify skills centered around specific entities and their interactions because all the information entities would have to be fused in a single vector representation.

With such a motivation, we propose to use a *structured* goal representation that can query and scope which objects to interact with, which can serve as a basis for solving more complex downstream tasks. Our particular choice is a pair of objects (A, B), namely A and B (among N objects), whose semantic meaning is that two objects A and B should have an interaction (or some mutual effects) as a consequence of the agent's skill. We design a novel intrinsic reward that determines and induces object interactions based on *counterfactual reasoning and inference*, called COIL (Counterfactual Object Interaction Learning). We show that an agent can learn object interaction behaviors (e.g., attaching or stacking one block to another) in entity-centric control environments (Magnetic Block and Stacking Box) without any external rewards or additional domain-specific knowledge.

CHAPTER 2

Contingency-Aware Exploration in Reinforcement Learning

This paper investigates whether learning contingency-awareness and controllable aspects of an environment can lead to better exploration in reinforcement learning. To investigate this question, we consider an instantiation of this hypothesis evaluated on the Arcade Learning Element (ALE). In this study, we develop an attentive dynamics model (ADM) that discovers controllable elements of the observations, which are often associated with the location of the character in Atari games. The ADM is trained in a self-supervised fashion to predict the actions taken by the agent. The learned contingency information is used as a part of the state representation for exploration purposes. We demonstrate that combining actor-critic algorithm with count-based exploration using our representation achieves impressive results on a set of notoriously challenging Atari games due to sparse rewards. For example, we report a state-of-the-art score of >11,000 points on MONTEZUMA'S REVENGE without using expert demonstrations, explicit high-level information (*e.g.*, RAM states), or supervisory data. Our experiments confirm that contingency-awareness is indeed an extremely powerful concept for tackling exploration problems in reinforcement learning and opens up interesting research questions for further investigations.

2.1 Introduction

The success of reinforcement learning (RL) algorithms in complex environments hinges on the way they balance *exploration* and *exploitation*. There has been a surge of recent interest in developing effective exploration strategies for problems with high-dimensional state spaces and sparse rewards (Schmidhuber, 1991b; Oudeyer & Kaplan, 2009; Houthooft et al., 2016; Bellemare et al., 2016; Osband et al., 2016; Pathak et al., 2017; Plappert et al., 2018b; Zheng et al., 2018). Deep neural networks have seen great success as expressive function approximators within RL and as powerful representation learning methods for many domains. In addition, there have been recent studies on using neural network representations for exploration (Tang et al., 2017; Martin et al., 2017; Pathak

et al., 2017). For example, count-based exploration with neural density estimation (Bellemare et al., 2016; Tang et al., 2017; Ostrovski et al., 2017) presents one of the state-of-the-art techniques on the most challenging Atari games with sparse rewards.

Despite the success of recent exploration methods, it is still an open question on how to construct an optimal representation for exploration. For example, the concept of visual similarity is used for learning density models as a basis for calculating *pseudo-counts* (Bellemare et al., 2016; Ostrovski et al., 2017). However, as Tang et al. (2017) noted, the ideal way to represent states should be based on what is relevant to solving the MDP, rather than only relying on visual similarity. In addition, there remains another question on whether the representations used for recent exploration works are easily interpretable. To address these questions, we investigate whether we can learn a complementary, more intuitive, and interpretable high-level abstraction that can be very effective in exploration by using the ideas of contingency awareness and controllable dynamics.

The key idea that we focus on in this work is the notion of *contingency awareness* (Watson, 1966; Bellemare et al., 2012) — the agent's understanding of the environmental dynamics and recognizing that some aspects of the dynamics are under the agent's control. Intuitively speaking, this can represent the segmentation mask of the agent operating in the 2D or 3D environments (yet one can think of more abstract and general state spaces). In this study, we investigate the concept of contingency awareness based on *self-localization*, *i.e.*, the awareness of where the agent is located in the abstract state space. We are interested in discovering parts of the world that are directly dependent on the agent's immediate action, which often reveal the agent's approximate location.

For further motivation on the problem, we note that contingency awareness is a very important concept in neuroscience and psychology. In other words, being self-aware of one's location is an important property within many observed intelligent organisms and systems (Moser et al., 2015; Kuipers, 2000; Durrant-Whyte & Bailey, 2006). For example, recent breakthroughs in neuroscience, such as the Nobel Prize winning work on the grid cells (Moser et al., 2015; Banino et al., 2018), show that organisms that perform very well in spatially-challenging tasks are self-aware of their location. This allows rats to navigate, remember paths to previously visited places and important sub-goals, and find shortcuts. In addition, the notion of contingency awareness has been shown as an important factor in developmental psychology (Watson, 1966; Baeyens et al., 1990). We can think of self-localization (and more broadly self-awareness) as a principled and fundamental direction towards intelligent agents.

Based on these discussions, we hypothesize that contingency awareness can be a powerful mechanism for tackling exploration problems in reinforcement learning. We consider an instantiation of this hypothesis evaluated on the Arcade Learning Element (ALE). For example, in the context of 2D Atari games, contingency-awareness roughly corresponds to understanding the notion of controllable entities (*e.g.*, the player's avatar), which Bellemare et al. (2012) refer to as *contingent regions*.

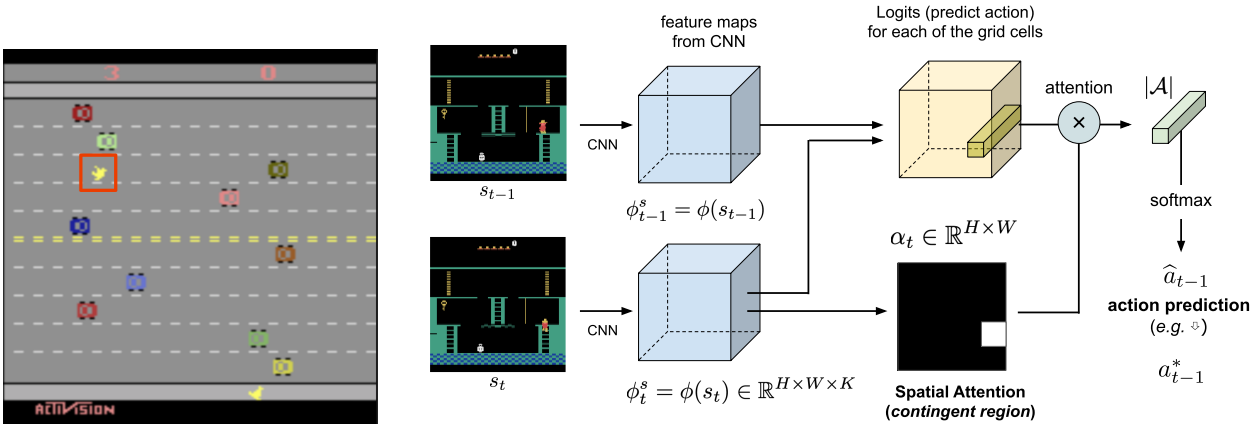


Figure 2.1: Left: Contingent region in FREEWAY; an object in a red box denotes what is under the agent's control, whereas the rest is not. **Right:** A diagram for the proposed ADM architecture.

More concretely, as shown in Figure 2.1, in the game FREEWAY, only the chicken sprite is under the agent's control and not the multiple moving cars; therefore the chicken's location should be an informative element for exploration (Bellemare et al., 2012; Pathak et al., 2017).

In this study, we also investigate whether contingency awareness can be learned without any external annotations or supervision. For this, we provide an instantiation of an algorithm for automatically learning such information and using it for improving exploration on a 2D ALE environment (Bellemare et al., 2013). Concretely, we employ an *attentive dynamics model* (ADM) to predict the agent's action chosen between consecutive states. It allows us to approximate the agent's position in 2D environments, but unlike other approaches such as (Bellemare et al., 2012), it does not require any additional supervision to do so. The ADM learns in an online and self-supervised fashion with pure observations as the agent's policy is updated and does not require hand-crafted features, an environment simulator, or supervision labels for training.

In experimental evaluation, our methods significantly improve the performance of A2C on hard-exploration Atari games in comparison with competitive methods such as density-based exploration (Bellemare et al., 2016; Ostrovski et al., 2017) and SimHash (Tang et al., 2017). We report very strong results on sparse-reward Atari games, including the state-of-the-art performance on the notoriously difficult MONTEZUMA'S REVENGE, when combining our proposed exploration strategy with PPO (Schulman et al., 2017), without using expert demonstrations, explicit high-level information (*e.g.*, RAM states), or resetting the environment to an arbitrary state.

We summarize our contributions as follows:

- We demonstrate the importance of learning contingency awareness for efficient exploration in challenging sparse-reward RL problems.

- We develop a novel instance of attentive dynamics model using contingency and controllable dynamics to provide robust localization abilities across the most challenging Atari environments.
- We achieve a strong performance on difficult sparse-reward Atari games, including the state-of-the-art score on the notoriously challenging MONTEZUMA'S REVENGE.

Overall, we believe that our experiments confirm the hypothesis that contingency awareness is an extremely powerful concept for tackling exploration problems in reinforcement learning, which opens up interesting research questions for further investigations.

2.2 Related Work

Self-Localization. The discovery of grid cells (Moser et al., 2015) motivates working on agents that are self-aware of their location. Banino et al. (2018) emphasize the importance of self-localization and train a neural network which learns a similar mechanism to grid cells to perform tasks related to spatial navigation. The presence of grid cells is correlated with high performance. Although grid cells seem tailored to 2D or 3D problems that animals encounter in their life, it is speculated that their use can be extended to more abstract spaces. A set of potential approaches to self-localization ranges from ideas specific to a given environment, *e.g.*, SLAM (Durrant-Whyte & Bailey, 2006), to learning high-level spatial representation for cognitive maps (Kuipers, 2000), and to other types of methods with potential generalizability (Mirowski et al., 2017; Jaderberg et al., 2017; Mirowski et al., 2018).

Self-supervised Dynamics Model and Controllable Dynamics. Several works have used forward and/or inverse dynamics models of the environment (Oh et al., 2015; Agrawal et al., 2016; Shelhamer et al., 2017). Pathak et al. (2017) employ a similar dynamics model to learn feature representations of states that captures controllable aspects of the environment. This dense representation is used to design a curiosity-driven intrinsic reward. The idea of learning representations on relevant aspects of the environment by learning auxiliary tasks is also explored in (Jaderberg et al., 2017; Bengio et al., 2017; Sawada, 2018). Our presented approach is different as we focus on explicitly discovering controllable aspects using an attention mechanism, resulting in better interpretability.

Exploration and Intrinsic Motivation. The idea of providing an exploration bonus reward depending on the state-action visit-count was proposed by Strehl & Littman (2008) (MBIE-EB), originally under a tabular setting. Later it has been combined with different techniques to deal with

high-dimensional state spaces. Bellemare et al. (2016) use a Context-Tree Switching (CTS) density model to derive a state *pseudo-count*, whereas Ostrovski et al. (2017) use PixelCNN as a state density estimator. Martin et al. (2017) also construct a visitation density model over a compressed feature space rather than the raw observation space. Alternatively, Tang et al. (2017) propose a locality-sensitive hashing (LSH) method to cluster states and maintain a state-visitation counter based on a form of similarity between frames. We train an agent with a similar count-based exploration bonus, but the way of maintaining state counter seems relatively simpler in that key feature information (*i.e.*, controllable region) is explicitly extracted from the observation and directly used for counting states.

Another popular family of exploration strategies in RL uses intrinsic motivation (Schmidhuber, 1991b; Singh et al., 2004; Oudeyer & Kaplan, 2009; Barto, 2013). These methods encourage the agent to look for something surprising in the environment which motivates its search for novel states, such as surprise (Achiam & Sastry, 2017), curiosity (Pathak et al., 2017; Burda et al., 2019a), and diversity (Eysenbach et al., 2019), or via feature control (Jaderberg et al., 2017; Dilokthanakul et al., 2017).

2.3 Approach

2.3.1 Discovering Contingency via Attentive Dynamics Model

To discover the region of the observation that is controllable by the agent, we develop an instance of *attentive dynamics model* (ADM) based on inverse dynamics f_{inv}. The model takes two consecutive input frames (observations) $s_{t-1}, s_t \in \mathcal{S}$ as input and aims to predict the action ($a_{t-1} \in \mathcal{A}$) taken by the agent to transition from s_{t-1} to s_t:

$$\widehat{a}_{t-1} = f_{\text{inv}}(s_{t-1}, s_t). \tag{2.1}$$

Our key intuition is that the inverse dynamics model should attend to the most relevant part of the observation, which is controllable by the agent, to be able to classify the actions. We determine whether each region in a $H \times W$ grid is controllable, or in other words, useful for predicting the agent's action, by using a spatial *attention mechanism* (Bahdanau et al., 2015; Xu et al., 2015). An overview of the model is shown in Figure 2.1.

Model. To perform action classification, we first compute a convolutional feature map $\phi_t^s = \phi(s_t) \in \mathbb{R}^{H \times W \times K}$ based on the observation s_t using a convolutional neural network ϕ. We estimate a set of *logit* (score) vectors, denoted $e_t(i, j) \in \mathbb{R}^{|\mathcal{A}|}$, for action classification from each grid

cell (i, j) of the convolutional feature map. The local convolution features and feature differences for consecutive frames are fed into a shared multi-layer perceptron (MLP) to derive the logits as:

$$e_t(i,j) = \mathrm{MLP}\Big(\big[\phi_t^s(i,j) - \phi_{t-1}^s(i,j);\ \phi_t^s(i,j)\big]\Big) \in \mathbb{R}^{|\mathcal{A}|}. \tag{2.2}$$

We then compute an attention mask $\alpha_t \in \mathbb{R}^{H \times W}$ corresponding to frame t, which indicates the controllable parts of the observation s_t. Such attention masks are computed via a separate MLP from the features of each region (i, j), and then converted into a probability distribution using softmax or sparsemax operators (Martins & Astudillo, 2016):

$$\alpha_t = \mathrm{sparsemax}(\widetilde{\alpha}_t) \quad \text{where} \quad \widetilde{\alpha}_t(i,j) = \mathrm{MLP}\big(\phi_t^s(i,j)\big), \tag{2.3}$$

so that $\sum_{i,j} \alpha_t(i,j) = 1$. The sparsemax operator is similar to softmax but yields a sparse attention, leading to more stable performance. Finally, the logits $e_t(i,j)$ from all regions are linearly combined using the attention probabilities α_t:

$$p(\widehat{a}_{t-1} \mid s_{t-1}, s_t) = \mathrm{softmax}\Big(\textstyle\sum_{i,j} \alpha_t(i,j) \cdot e_t(i,j)\Big) \in \mathbb{R}^{|\mathcal{A}|}. \tag{2.4}$$

Training. The model can be optimized with the standard cross-entropy loss $\mathcal{L}_{\text{action}}(a_{t-1}^*, \widehat{a}_{t-1})$ with respect to the ground-truth action $a_{t-1}^* \in \mathcal{A}$ that the agent actually has taken. Based on this formulation, the attention probability $\alpha_t(i,j)$ should be high only on regions (i,j) that are predictive of the agent's actions. Our formulation enables learning to localize controllable entities in a self-supervised way without any additional supervisory signal, unlike some prior work (*e.g.*, Bellemare et al. (2012)) that adopts simulators to collect extra supervisory labels.

Optimizing the parameters of ADM on on-policy data is challenging for several reasons. First, the ground-truth action may be unpredictable for given pairs of frames, leading to noisy labels. For example, actions taken in uncontrollable situations do not have any effect (*e.g.*, when the agent is in the middle of jumping in MONTEZUMA'S REVENGE). Second, since we train the ADM online along with the policy, the training examples are not independently and identically distributed, and the data distribution can shift dramatically over time. Third, the action distribution from the agent's policy can run into a low entropy[1], being biased towards certain actions. These issues may prevent the ADM from generalization to novel observations, which hurts exploration. Generally, we prefer models that quickly adapt to the policy and learn to localize the controllable regions in a robust manner.

To mitigate the aforementioned issues, we adopt a few additional objective functions. We encourage the attention distribution to attain a high entropy by including an *attention entropy regular-*

[1]We note that an entropy regularization term (*e.g.*, Eq.(A.4)) is used when learning the policy.

ization loss, *i.e.*, $\mathcal{L}_{\text{ent}} = -\mathcal{H}(\alpha_t)$. This term penalizes over-confident attention masks, making the attention closer to uniform whenever action prediction is not possible. We also train the logits corresponding to each grid cell independently using a separate cross-entropy loss: $p(\widehat{a}_{t-1}^{i,j} \mid e_t(i,j)) = \text{softmax}(e_t(i,j))$. These additional cross-entropy losses, denoted $\mathcal{L}_{\text{cell}}^{i,j}$, allow the model to learn from unseen observations even when attention fails to perform well at first. The entire training objective becomes:

$$\mathcal{L}^{\text{ADM}} \;=\; \mathcal{L}_{\text{action}} + \textstyle\sum_{i,j} \mathcal{L}_{\text{cell}}^{i,j} + \lambda_{\text{ent}} \mathcal{L}_{\text{ent}} \tag{2.5}$$

where λ_{ent} is a mixing hyperparameter.

2.3.2 Count-based Exploration with Contingent Regions

One natural way to take advantage of discovered contingent regions for exploration is count-based exploration. The ADM can be used to localize the controllable entity (*e.g.*, the agent's avatar) from an observation s_t experienced by the agent. In 2D environments, a natural discretization $(x, y) = \arg\max_{(j,i)} \alpha_t(i,j)$ provides a good approximation of the agent's location within the current observation[2]. This provides a key piece of information about the current state of the agent.

Inspired by previous work (Bellemare et al., 2016; Tang et al., 2017), we add an exploration bonus of r^+ to the environment reward, where $r^+(s) = 1/\sqrt{\#(\psi(s))}$ and $\#(\psi(s))$ denotes the visitation count of the (discrete) mapped state $\psi(s)$, which consists of the contingent region (x, y). We want to find a policy π that maximizes the expected discounted sum of environment rewards r^{ext} plus count-based exploration rewards r^+, denoted $\mathcal{R} = \mathbb{E}_\pi\big[\sum_t \gamma^t \left(\beta_1 r^{\text{ext}}(s_t, a_t) + \beta_2 r^+(s_t)\right)\big]$, where $\beta_1, \beta_2 \geq 0$ are hyperparameters that balance the weight of environment reward and exploration bonus. For every state s_t encountered at time step t, we increase the counter value $\#(\psi(s_t))$ by 1 during training. The full procedure is summarized in Algorithm A.1 in Appendix A.1.

2.4 Experiments

In the experiments below we investigate the following key questions:

- Does the contingency awareness in terms of self-localization provide a useful state abstraction for exploration?
- How well can the self-supervised model discover the ground-truth abstract states?
- How well does the proposed exploration strategy perform against other exploration methods?

[2]To obtain more accurate localization by taking temporal correlation into account, we can use exponential smoothing as $\overline{\alpha}_t(i,j) = (1-\omega_t)\overline{\alpha}_{t-1}(i,j) + \omega_t \alpha_t(i,j)$, where $\omega_t = \max_{(i,j)}\{\alpha_t(i,j)\}$.

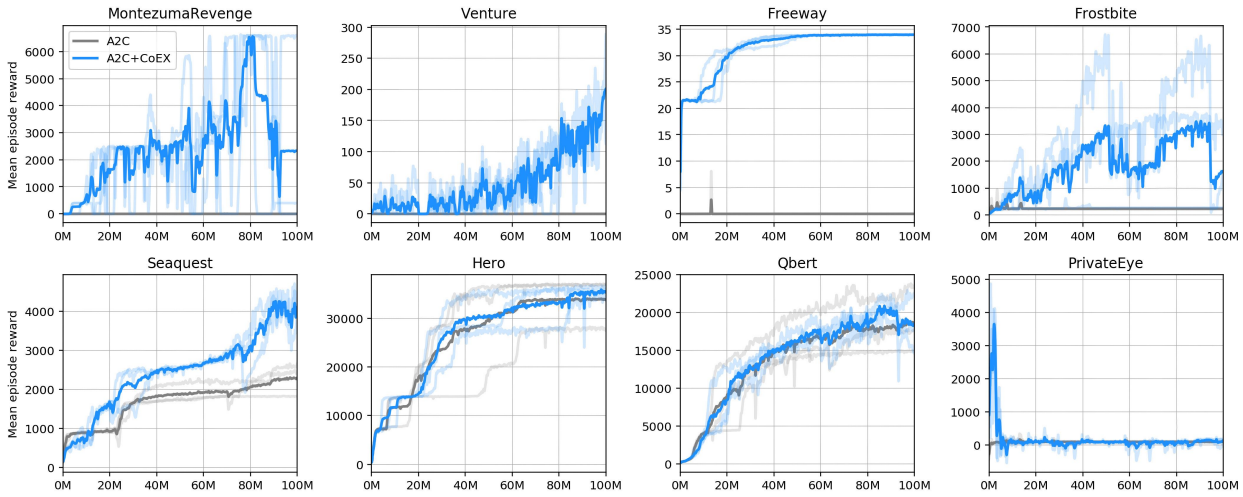


Figure 2.2: Learning curves on several Atari games: A2C+CoEX and A2C. The x-axis represents total environment steps and the y-axis the mean episode reward averaged over 40 recent episodes. The mean curve is obtained by averaging over 3 random seeds, each shown in a light color.

2.4.1 Experiments with A2C

We evaluate the proposed exploration strategy on several difficult exploration Atari 2600 games from the Arcade Learning Environment (ALE) (Bellemare et al., 2013). We focus on 8 Atari games including FREEWAY, FROSTBITE, HERO, PRIVATEEYE, MONTEZUMA'S REVENGE, QBERT, SEAQUEST, and VENTURE. In these games, an agent without an effective exploration strategy can often converge to a suboptimal policy. For example, as depicted in Figure 2.2, the Advantage Actor-Critic (A2C) baseline (Mnih et al., 2016) achieves a reward close to 0 on MONTEZUMA'S REVENGE, VENTURE, FREEWAY, FROSTBITE, and PRIVATEEYE, even after 100M steps of training. By contrast, our proposed technique, which augments A2C with count-based exploration with the location information learned by the attentive dynamics model, denoted **A2C+CoEX** (CoEX stands for "Contingency-aware Exploration"), significantly outperforms the A2C baseline on six out of the 8 games.

We compare our proposed A2C+CoEX technique against the following baselines:[3]

- **A2C**: an implementation adopted from OpenAI baselines (Dhariwal et al., 2017) using the default hyperparameters, which serves as the building block of our more complicated baselines.

- **A2C+Pixel-SimHash**: Following (Tang et al., 2017), we map 52×52 gray-scale observations

[3]In Section 2.4.6, we also report experiments using Proximal Policy Optimization (PPO) (Schulman et al., 2017) as a baseline, where our **PPO+CoEX** achieves the average score of >11,000 on MONTEZUMA'S REVENGE.

Method	Freeway	Frostbite	Hero	Montezuma	PrivateEye	Qbert	Seaquest	Venture
A2C	7.2	1099	34352	13	574	19620	2401	0
A2C+Pixel-SimHash	0.0	829	28181	412	276	18180	2177	31
A2C+CoEX	**34.0**	**4260**	**36827**	**6635**	**5316**	**23962**	**5169**	**204**
A2C+CoEX+RAM*	34.0	4418	36765	6600	24296	24422	6113	1100

Table 2.1: Performance of our method and its baselines on Atari games: maximum mean scores (averaged over 40 recent episodes) achieved over total 100M environment timesteps (400M frames) of training, averaged over 3 seeds. The best entry in the group of experiments without supervision is shown in bold. * denotes that A2C+CoEX+RAM acts as a control experiment, which includes some supervision. More experimental results on A2C+CoEX+RAM are shown in Appendix A.3.

to 128-bit binary codes using random projection followed by quantization (Charikar, 2002). Then, we add a count-based exploration bonus based on quantized observations.

As a control experiment, we evaluate **A2C+CoEX+RAM***, our contingency-aware exploration method together with the ground-truth location information obtained from game's RAM. It is roughly an upper-bound of the performance of our approach.

2.4.2 Implementation Details

For the A2C (Mnih et al., 2016) algorithm, we use 16 parallel actors to collect the agent's experience, with 5-step rollout, which yields a minibatch of size 80 for on-policy transitions. We use the last 4 observation frames stacked as input, each of which is resized to 84×84 and converted to grayscale as in (Mnih et al., 2015, 2016). We set the end of an episode to when the game ends, rather than when the agent loses a life. Each episode is initialized with a random number of no-ops (Mnih et al., 2015). More implementation details can be found in Appendix A.1 and A.2.

For the ADM, we take observation frames of size 160×160 as input (resized from the raw observation of size 210×160).[4] We employ a 4-layer convolutional neural network that produces a feature map $\phi(s_t)$ with a spatial grid size of $H \times W = 9 \times 9$. As a result, the prediction of location coordinates lies in the 9×9 grid.

In some environments, the contingent regions within the visual observation alone are not sufficient to determine the exact location of the agent within the game; for example, the coordinate cannot solely distinguish between different rooms in HERO, MONTEZUMA'S REVENGE, and PRIVATEEYE, etc. Therefore, we introduce a discrete context representation $c \in \mathbb{Z}$ that summarizes the high-level visual context in which the agent currently lies. We use a simple clustering method

[4] In some games such as Venture, the agent is depicted in very small pixels, which might be hardly recognizable in rescaled 84×84 images.

Method	#Steps	Freeway	Frostbite	Hero	Montezuma	PrivateEye	Qbert	Seaquest	Venture
A2C+CoEX (Ours)	50M	33.9	3900	31367	4100	5316	17724	2620	128
A2C+CoEX (Ours)	100M	**34.0**	4260	**36827**	**6635**	5316	**23962**	**5169**	204
DDQN+	25M	29.2	-	20300	3439	1880	-	-	369
A3C+	50M	27.3	507	15210	142	100	15805	2274	0
TRPO-AE-SimHash	50M	33.5	**5214**	-	75	-	-	-	445
Sarsa-ϕ-EB	25M	0.0	2770	-	2745	-	4112	-	1169
DQN-PixelCNN	37.5M	31.7	-	-	2514	**15806**	5501	-	**1356**
Curiosity-Driven	25M	32.8	-	-	2505	3037	-	-	416

Table 2.2: Performance of our method and state-of-the-art exploration methods on Atari games. For fair comparison, we report the maximum mean score achieved over the specific number of timesteps during training, averaged over 3 seeds. The best entry is shown in bold. Baselines (for reference) are: DDQN+ and A3C+ (Bellemare et al., 2016), TRPO-AE-SimHash (Tang et al., 2017), Sarsa-ϕ-EB (Martin et al., 2017), DQN-PixelCNN (Ostrovski et al., 2017), and Curiosity-Driven (Burda et al., 2019a). The numbers for DDQN+ were taken from (Tang et al., 2017) or were read from a plot.

similar to (Kulis & Jordan, 2012), which we refer to as *observation embedding clustering* that clusters the random projection vectors of the input frames as in (Tang et al., 2017), so that different contexts are assigned to different clusters. We further explain this heuristic approach more in detail in Appendix A.4.

In sparse-reward problems, the act of collecting a reward is rare but frequently instrumental for the future states of the environment. The cumulative reward $R_t = \sum_{t'=0}^{t-1} r^{\text{ext}}(s_{t'}, a_{t'})$ from the beginning of the episode up to the current step t, can provide a useful high-level *behavioral context* because collecting rewards can trigger significant changes to the agent's state and as a result the optimal behavior can change as well. In this sense, the agent should revisit the previously visited location for exploration when the context changes. For example, in MONTEZUMA'S REVENGE, if the agent is in the first room and the cumulative reward is 0, we know the agent has not picked up the key and the optimal policy is to reach the key. However, if the cumulative reward in the first room is 100, it means the agent has picked up the key and the next optimal goal is to open a door and move on to the next room. Therefore, we could include the cumulative reward as a part of state abstraction for exploration, which leads to empirically better performance.

To sum up, for the purpose of count-based exploration, we utilize the location (x, y) of the controllable entity (*i.e.*, the agent) in the current observation discovered by ADM (Section 2.3.1), a context representation $c \in \mathbb{Z}$ that denotes the high level visual context, and a cumulative environ-

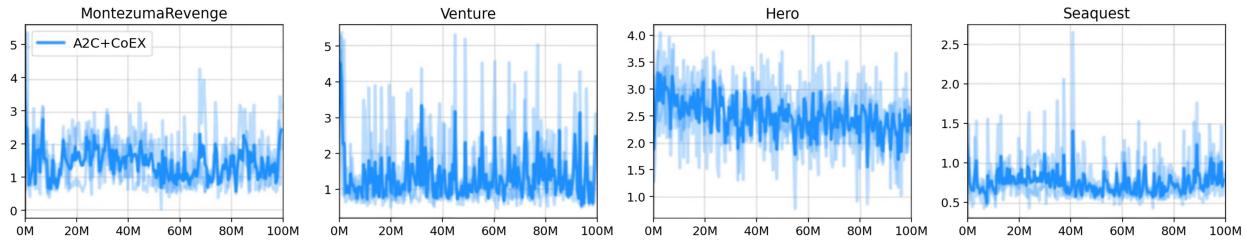


Figure 2.3: Performance plot of ADM trained using on-policy samples from the A2C+CoEX agent.

ment reward $R \in \mathbb{Z}$ that represents the exploration behavioral state. In such setting, we may denote $\psi(s) = (x, y, c, R)$.

2.4.3 Performance of Count-Based Exploration

Figure 2.2 shows the learning curves of the proposed methods on 8 Atari games. The performance of our method A2C+CoEX and A2C+CoEX+RAM as well as the baselines A2C and A2C+Pixel-SimHash are summarized in Table 2.1. In order to find a balance between the environment reward and the exploration bonus reward, we perform a hyper-parameter search for the proper weight of the environment reward β_1 and the exploration reward β_2 for A2C+CoEX+RAM, as well as for A2C+CoEX. The hyper-parameters for the two ended up being the same, which is consistent with our results. For fair comparison, we also search for the proper weight of environment reward for A2C baseline. The best hyper-parameters for each game are shown in Table A.2 in Appendix A.2.

Compared to the vanilla A2C, the proposed exploration strategy improves the score on all the hard-exploration games. As shown in Table 2.1, provided the representation (x, y, c, R) is perfect, A2C+CoEX+RAM achieves a significant improvement over A2C by encouraging the agent to visit novel locations, and could nearly solve these hard exploration games as training goes on.

Furthermore, A2C+CoEX using representations learned with our proposed attentive dynamics model and observation embedding clustering also outperforms the A2C baseline. Especially on Freeway, Frostbite, Hero, Montezuma's Revenge, Qbert and Seaquest, the performance is comparable with A2C+CoEX+RAM, demonstrating the usefulness of the contigency-awareness information discovered by ADM.

Comparison to other count-based exploration methods. Table 2.2 compares the proposed method with previous state-of-the-art results, where our proposed method outperforms the other methods on 5 out of 8 games. DQN-PixelCNN is the strongest alternative achieving a state-of-the-art performance on some of the most difficult sparse-reward games. We argue that using Q-learning as the base learner with DQN-PixelCNN makes the direct comparison with A2C+CoEX not com-

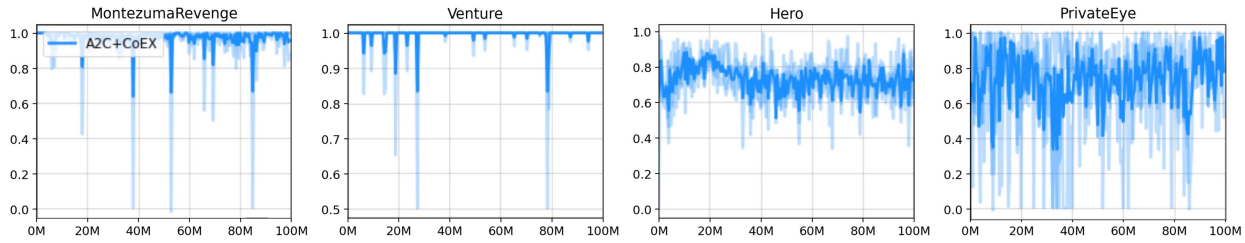


Figure 2.4: Curves of ARI score during training of A2C+CoEX, averaged over 100 recent observations.

pletely adequate. Note that the closest alternative count-based exploration method to A2C+CoEX would be A3C+ (Bellemare et al., 2016), which augments A3C (Mnih et al., 2016) with exploration bonus derived from pseudo-count, because A2C and A3C share a similar policy learning method. With that in mind, one can observe a clear improvement of A2C+CoEX over A3C+ on all of the 8 Atari games.

2.4.4 Analysis of Attentive Dynamics Model

We also analyze the performance of the ADM that learns the controllable dynamics of the environment. As a performance metric, we report the average distance between the ground-truth agent location (x^*, y^*) and the predicted location (x, y) within the 9×9 grid: $\|(x, y) - (x^*, y^*)\|_2$. The ground-truth location of the agent is extracted from RAM[5], then rescaled so that the observation image frame fits into the 9×9 grid.

Figure 2.3 shows the results on 4 Atari games (MONTEZUMA'S REVENGE, SEAQUEST, HERO, and VENTURE). The ADM is able to quickly capture the location of the agent without any supervision of localization, despite the agent constantly visiting new places. Typically the predicted location is on average 1 or 2 grid cells away from the ground-truth location. Whenever a novel scene is encountered (*e.g.*, the second room in MONTEZUMA'S REVENGE at around 10M steps), the average distance temporarily increases but quickly drops again as the model learns the new room. We provide videos of the agents playing and localization information as the supplementary material.

2.4.5 Analysis of Observation Embedding Clustering

To make the agent aware of a change in high-level visual context (*i.e.*, rooms in Atari games) in some games such as MONTEZUMA'S REVENGE, VENTURE, HERO, and PRIVATEEYE, we obtain

[5]Please note that the location from RAM is used only for analysis and evaluation purposes.

a representation of the high-level context and use it for exploration. The high-level visual contexts are different from each other (different layouts, objects, colors, etc.), so the embedding generated by a random projection is quite distinguishable and the clustering is accurate and robust.

For evaluation, given an observation in Atari games, we compare the discrete representation (*i.e.*, which cluster it is assigned to) based on the embedding from random projection to the ground-truth room number extracted from RAM. The Adjusted Rand Index (ARI) (Rand, 1971) measures the similarity between these two data clusterings. The ARI may only yield a value between 0 and 1, and is exactly 1 when the clusterings are identical.

The curves of the Adjusted Rand Index are shown in Figure 2.4. For MONTEZUMA'S REVENGE and VENTURE, the discrete representation as room number is roughly as good as the ground-truth. For HERO and PRIVATEEYE, since there are many rooms quite similar to one another, it is more challenging to accurately cluster the embeddings. The samples shown in Figure A.2 in Appendix A.4 show reasonable performances of the clustering method on all these games.

2.4.6 Additional Experiments with PPO

We also evaluate the proposed exploration algorithm on MONTEZUMA'S REVENGE using the sticky actions environment setup (Machado et al., 2017) identical to the setup found in (Burda et al., 2019b). In the sticky action setup, the agent randomly repeats the previous action with probability of 0.25, preventing the algorithm from simply memorizing the correct sequence of actions and relying on determinism. The agent is trained with Proximal Policy Optimization (PPO) (Schulman et al., 2017) in conjunction with the proposed exploration method using 128 parallel actors to collect the experience. We used reward normalization and advantage normalization as in (Burda et al., 2019a).

The method, denoted **PPO+CoEX**, achieves the score of 11,618 at 500M environment steps (2 billion frames) on MONTEZUMA'S REVENGE, when averaged over 3 runs. The learning curve is illustrated in Figure 2.5. Since the vanilla PPO baseline achieves a score near 0 (our runs) or 1,797 (Burda et al., 2019b), this result is not solely due to the benefits of PPO. There is another approach "Exploration by Random Network Distillation" (Burda et al., 2019b) concurrent with our work which achieves similar performance by following a slightly different philosophy.

2.4.7 Discussions and Future Work

This paper investigates whether discovering controllable dynamics via an attentive dynamics model (ADM) can help exploration in challenging sparse-reward environments. We demonstrate the effectiveness of this approach by achieving significant improvements on notoriously difficult video games. That being said, we acknowledge that our approach has certain limitations. Our currently presented instance of state abstraction method mainly focuses on controllable dynamics and em-

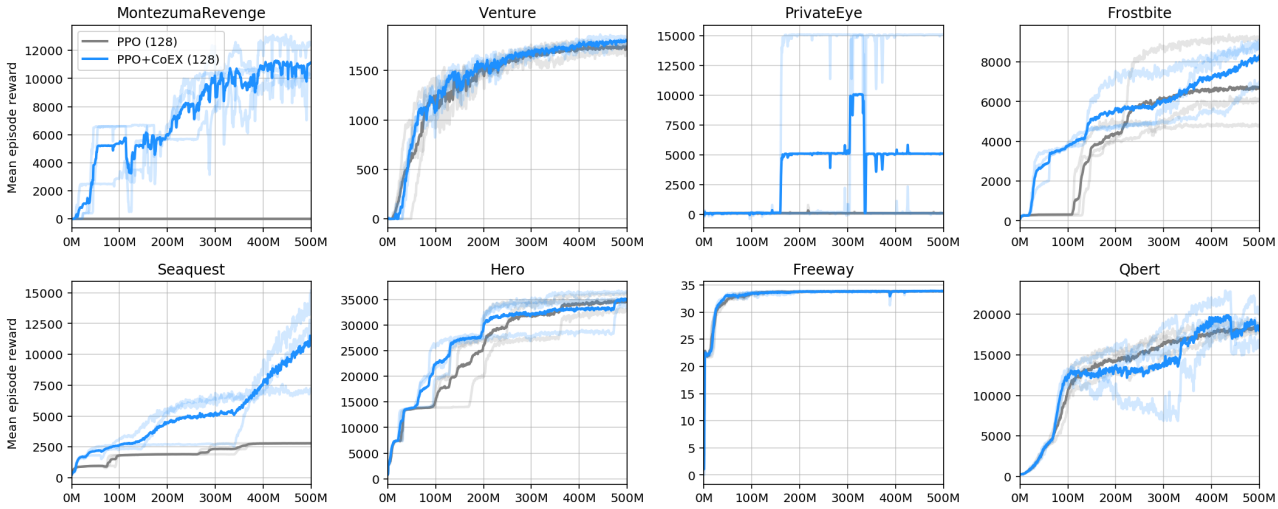


Figure 2.5: The learning curve of PPO+CoEX on several Atari games with sticky actions setup. The x-axis represents the total number of environment steps and the y-axis the mean episode reward averaged over 40 recent episodes. The mean curve is obtained by averaging over 3 random seeds, each shown in a light color.

Method	#Steps	Freeway	Frostbite	Hero	Montezuma	PrivateEye	Qbert	Seaquest	Venture
PPO	500M	**34.0**	7340	36263	29	942	19980	2806	1875
PPO+CoEX	500M	**34.0**	**9076**	**36664**	**11618**	**11000**	**22647**	**11794**	**1916**

Table 2.3: Performance of PPO and PPO+CoEX: maximum mean scores (average over 40 recent episodes) achieved over total 500M environment steps (2B frames) of training, averaged over 3 seeds.

ploys a simple clustering scheme to abstract away uncontrollable elements of the scene. In more general setting, one can imagine using attentive (forward or inverse) dynamics models to learn an effective and compact abstraction of the controllable and uncontrollable dynamics as well, but we leave this to future work.

Key elements of the current ADM method include the use of spatial attention and modelling of the dynamics. These ideas can be generalized by a set of attention-based dynamics models (ADM) operating in forward, inverse, or combined mode. Such models could use attention over a lower-dimensional embedding that corresponds to an intrinsic manifold structure from the environment (*i.e.*, intuitively speaking, this also corresponds to *being self-aware of (e.g., locating) where the agent is in the abstract state space*). Our experiments with the inverse dynamics model suggest that the mechanism does not have to be perfectly precise, allowing for some error in practice. We speculate that mapping to such subspace could be obtained by techniques of embedding learning.

We note that RL environments with different visual characteristics may require different forms

of attention-based techniques and properties of the model (*e.g.*, partial observability). Even though this paper focuses on 2D video games, we believe that the presented high-level ideas of learning contingency-awareness (with attention and dynamics models) are more general and could be applicable to more complex 3D environments with some extension. We leave this as future work.

2.5 Conclusion

We proposed a method of providing contingency-awareness through an attentive dynamics model (ADM). It enables approximate self-localization for an RL agent in 2D environments (as a specific perspective). The agent is able to estimate its position in the space and therefore benefits from a compact and informative representation of the world. This idea combined with a variant of count-based exploration achieves strong results in various sparse-reward Atari games. Furthermore, we report state-of-the-art results of >11,000 points on the infamously challenging MONTEZUMA'S REVENGE without using expert demonstrations or supervision. Though in this work we focus mostly on 2D environments in the form of sparse-reward Atari games, we view our presented high-level concept and approach as a stepping stone towards more universal algorithms capable of similar abilities in various RL environments.

CHAPTER 3

Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning

Learning to reach goal states and learning diverse skills through mutual information (MI) maximization have been proposed as principled frameworks for self-supervised reinforcement learning, allowing agents to acquire broadly applicable multi-task policies with minimal reward engineering. Starting from a simple observation that the standard goal-conditioned RL (GCRL) is encapsulated by the optimization objective of variational empowerment, we discuss how GCRL and MI-based RL can be generalized into a single family of methods, which we name *variational GCRL (VGCRL)*, interpreting variational MI maximization, or variational empowerment, as representation learning methods that acquire functionally-aware state representations for goal reaching. This novel perspective allows us to: (1) derive simple but unexplored variants of GCRL to study how adding small representation capacity can already expand its capabilities; (2) investigate how discriminator function capacity and smoothness determine the quality of discovered skills, or latent goals, through modifying latent dimensionality and applying spectral normalization; (3) adapt techniques such as hindsight experience replay (HER) from GCRL to MI-based RL; and lastly, (4) propose a novel evaluation metric, named latent goal reaching (LGR), for comparing empowerment algorithms with different choices of latent dimensionality and discriminator parameterization. Through principled mathematical derivations and careful experimental studies, our work lays a novel foundation from which to evaluate, analyze, and develop representation learning techniques in goal-based RL.

3.1 Introduction

Reinforcement learning (RL) provides a general framework for discovering optimal behaviors for sequential decision-making. Combined with powerful function approximators like neural networks, RL can be used to learn to play computer games from raw pixels (Mnih et al., 2013) and acquire complex sensorimotor skills with real-world robots (Gu et al., 2017a; Kalashnikov et al., 2018;

Haarnoja et al., 2018). Neural networks show best performance, generalization, and reusability when they are trained on large and diverse datasets (Krizhevsky et al., 2012; Devlin et al., 2018). However, a critical limitation in RL is that human experts often need to spend considerable efforts designing and fine-tuning reward functions per task, making it hard to scale and define a huge set of tasks in advance. If we have agents that can interact with the world without rewards, build up a body of knowledge autonomously, and utilize this knowledge to accomplish new tasks efficiently, then we can greatly scale up task and skill learning to achieve similar level of generalization and performance for RL as what neural networks have enabled for other domains.

Several works have tried to find a single generalizable task-agnostic reward function which can potentially be used across several environments. The use of such intrinsic reward functions has been motivated as exploration heuristics such as curiosity and novelty (Schmidhuber, 1991a; Oudeyer & Kaplan, 2009; Bellemare et al., 2016; Pathak et al., 2017), as optimizing mutual information (MI) (Gregor et al., 2017; Eysenbach et al., 2019; Sharma et al., 2020b,a) or as empowerment (Klyubin et al., 2005; Jung et al., 2011; Mohamed & Rezende, 2015). Classically, goal-conditioned RL (GCRL) has shown success in learning diverse and useful skills in concurrence to MI-based methods. GCRL optimizes a stationary and interpretable reward for goal-reaching, but when the goal space is high-dimensional, how does the agent know which part of the space is relevant and which part can be ignored? In such cases, prior GCRL works frequently rely on manual definition (Andrychowicz et al., 2017) or off-the-shelf representation learning (Nachum et al., 2018; Nair et al., 2018; Wu et al., 2018) optimized prior to or separately from reinforcement learning. Meanwhile, MI or empowerment-based RL offers a clear objective for representation learning *through* reinforcement learning, but the properties of the learned behaviors are often unclear due to lack of a proper evaluation metric. Prior works use qualitative inspections of learned behaviors, variational bound estimates, or downstream task performances of a skill-utilizing high-level policy (Eysenbach et al., 2019; Sharma et al., 2020b), but these heuristics are costly or indirect measures and make objective comparisons and analyses of various mathematically-similar MI-based algorithms difficult (Florensa et al., 2017; Eysenbach et al., 2019; Achiam et al., 2018; Warde-Farley et al., 2019; Hansen et al., 2020; Sharma et al., 2020b). To recover a more direct metric, an important question is: what do these MI-based objectives learn representations for?

In this work, we interpret MI and empowerment-based RL as a principled framework for representation learning in goal-conditioned RL. Starting from a simple observation that the objective of the standard GCRL can be seen as a special case of variational MI with a fixed hard-coded variational posterior, our analysis provides a unification of these ideas and explicitly reframes skill discovery via mutual information maximization (Gregor et al., 2017; Eysenbach et al., 2019) as a combination of representation learning and goal-conditioned reinforcement learning, where both the space of goals and the skills to reach those goals are learned jointly via a MI-based objective.

While the connections between representation learning, mutual information estimation, and goal-conditioned RL have been explored in a number of previous works (Gregor et al., 2017; Warde-Farley et al., 2019; Gupta et al., 2018), our exact mathematical formulation and granular analyses enable new perspectives and synergies between GCRL and MI-based RL:

1. **[MI to GCRL]** We propose simple but novel variants of GCRL – adaptive-variance and linear-mapping GCRL – to study how adding small representation capacity can already expand the capabilities of GCRL.
2. **[MI to GCRL]** We show that a proper representation regularization from generative modeling, such as spectral normalization (Miyato et al., 2018), can improve the quality of latent goals discovered (and the stability of MI-based algorithms).
3. **[GCRL to MI]** We adapt hindsight experience replay (HER) (Andrychowicz et al., 2017) from GCRL to more general MI-based objectives and show posterior HER (P-HER) consistently provides substantial performance gains in MI-based RL algorithms.
4. **[GCRL to MI]** We propose the latent goal reaching (LGR) metric as an intuitive, task-oriented, and discriminator-agnostic metric for objectively evaluating empowerment algorithms.

3.2 Related Work

Reward engineering has been a bottleneck to broad application of RL. Some of the prior attempts to alleviate this problem have sought introduce human supervision in alternative, easier forms, such as demonstrations (Ng et al., 2000; Abbeel & Ng, 2004; Ziebart et al., 2008; Ho & Ermon, 2016; Fu et al., 2017; Ghasemipour et al., 2019) or preferences (Hadfield-Menell et al., 2017; Christiano et al., 2017). However, since these methods still rely on non-negligible amounts of human interventions, they cannot automatically scale to solving thousands of new environments and tasks.

Empowerment and reward-free RL. Task-agnostic reward functions have been proposed to encourage exploration in environments using notions of curiosity or novelty (Schmidhuber, 1991a; Oudeyer & Kaplan, 2009; Schmidhuber, 2010; Bellemare et al., 2016; Pathak et al., 2017; Colas et al., 2018). In a similar vein, some methods maximize the state-visitation entropy (Hazan et al., 2018; Pong et al., 2019; Lee et al., 2019a; Ghasemipour et al., 2019). These approaches can enable solutions to otherwise hard exploration sparse-reward problems. Some of the recent work has emphasized on empowerment or option/skill discovery through optimization of mutual information based intrinsic reward functions. Classically, empowerment measures the ability of an agent to control the environment (Salge et al., 2014; Klyubin et al., 2005; Jung et al., 2011), which was scaled up by Mohamed & Rezende (2015); Karl et al. (2017). The concept of mutual information,

which is also at the heart of empowerment based methods, has been further used to motivate several objectives for skill discovery (Florensa et al., 2017; Eysenbach et al., 2019; Achiam et al., 2018; Warde-Farley et al., 2019; Hansen et al., 2020; Sharma et al., 2020b; Campos et al., 2020). Recent works have shown that skills learned through mutual information can be meaningfully combined to solve downstream tasks (Eysenbach et al., 2019; Sharma et al., 2020b), even on real robots (Sharma et al., 2020a).

Goal-conditioned RL. Goal-conditioned RL (Kaelbling, 1993b; Pong et al., 2018; Andrychowicz et al., 2017; Schaul et al., 2015) provides a framework for enabling agents to reach user-specified goal states. The behaviors learned via GCRL can be interpretable and easy to analyze in terms of the goal-reaching function. However, GCRL assumes that a goal-reaching function has been specified in addition to goal states, which precludes broader application for the same reasons as those for reward engineering. To overcome such limitations, some prior works have used variational inference (Rudner et al., 2021) or mutual information in the GCRL framework (Pong et al., 2019; Warde-Farley et al., 2019) to learn. However, these works use the MI optimization as an unsupervised scheme to generate and achieve goals. On the other hand, our work studies skill-discovery/empowerment methods and provides an explicit reinterpretation within the GCRL framework, combining the representation learning perspective with the goal-reaching behavior of GCRL.

3.3 Background

In this section, we briefly review mutual information (MI)-based objectives for skill discovery, focusing on variational approaches introduced in (Mohamed & Rezende, 2015; Gregor et al., 2017; Eysenbach et al., 2019; Sharma et al., 2020b), and goal-conditioned RL (GCRL).

We denote a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$ denotes the underlying (possibly stochastic) transition dynamics of the environment with the initial state distribution $p_0 : \mathcal{S} \to [0, \infty)$, and a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The goal of the RL optimization problem is to learn a policy $\pi(a \mid s)$ which maximizes the return $\mathbb{E}_{p,\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] = \mathbb{E}_{s\sim\rho^\pi, a\sim\pi}\left[r(s, a)\right]$, for a discount factor $\gamma \in [0, 1)$ where ρ^π is an unnormalized γ-discounted state visitation density. Importantly, once we write an objective in the form of $\mathbb{E}_{\rho^\pi,\pi}\left[r(s, a)\right]$, we can apply the policy gradient theorem (Sutton et al., 2000) to derive a practical RL solver, as done in (Kakade, 2002; Silver et al., 2014; Schulman et al., 2015; Gu et al., 2017b; Ciosek & Whiteson, 2018), or learn it with Q-learning (Watkins & Dayan, 1992). For simplicity of our notations, we omit the discount factor γ in the following sections and derivations.

3.3.1 Mutual Information Maximization and Empowerment

MI maximization in RL such as empowerment generally means maximizing the mutual information between (some representations of) actions and (some representations of) future states following those actions (Klyubin et al., 2005; Mohamed & Rezende, 2015). The goal is to learn a set of actions that can influence future states to be diverse, but also be predictable if we know what action is taken. In this work we focus on learning abstract representation $z \in \mathcal{Z}$, which is an additional input to the policy $\pi(a|s, z)$ and defines a set of empowered actions. The latent code z (either discrete or continuous) can be interpreted as a macro-action, *skill* or *goal* (Eysenbach et al., 2019; Sharma et al., 2020b).

We discuss two variants of MI objectives in RL: *state-predictive MI* (Sharma et al., 2020b), which maximizes $\mathcal{I}(s'; z \mid s)$, and *state-marginal MI* (Eysenbach et al., 2019), which maximizes $\mathcal{I}(s; z)$. Due to page limit, we discuss these variants more in detail in Appendix B.1. In this work, we focus on state-marginal MI, whose optimization objective is:

$$\begin{aligned} \mathcal{I}(s; z) &= \mathbb{E}_{z\sim p(z), s\sim\rho^{\pi}(s|z)}[\log p(z \mid s) - \log p(z)] \\ &\geq \mathbb{E}_{z\sim p(z), s\sim\rho^{\pi}(s|z)}[\log q_\lambda(z \mid s) - \log p(z)] \end{aligned} \tag{3.1}$$

where $q_\lambda(z|s)$ is a variational approximation to the intractable posterior $p(z|s)$, often called a (skill) discriminator (Eysenbach et al., 2019).

Given a parameterized policy $\pi_\theta(a|s, z)$, Eq. 3.1 gives a joint maximization objective (a variational lower bound) with respect to π_θ and q_λ:

$$\mathcal{F}(\theta, \lambda) = \mathbb{E}_{z,s\sim\pi_\theta}\left[\log q_\lambda(z|s) - \log p(z)\right]. \tag{3.2}$$

A simple iterative RL procedure can be derived to optimize this lower bound, assuming a parameterized policy $\pi_\theta(a|s, z)$, where at iteration i,

$$\lambda^{(i)} \leftarrow \arg\max_\lambda \mathbb{E}_{z,s\sim\pi^{(i-1)}}\left[\log q_\lambda(z|s) - \log p(z)\right] \tag{3.3}$$

$$\theta^{(i)} \leftarrow \arg\max_\theta \mathbb{E}_{z,s\sim\pi_\theta}\left[\log q_{\lambda^{(i)}}(z|s) - \log p(z)\right]. \tag{3.4}$$

Eq. 3.3 is a simple supervised regression (*e.g.*, maximum likelihood) on on-policy samples. Eq. 3.4 has the same form of standard RL, and therefore can be optimized using any RL algorithm (Gregor et al., 2017; Eysenbach et al., 2019).

3.3.2 Goal-Conditioned RL

Goal-conditioned RL (Kaelbling, 1993b; Schaul et al., 2015) (GCRL) is a standard, stationary-reward problem where we aim to find a policy $\pi(a|s, g)$ conditional on a goal $g \in \mathcal{G}$ by maximizing

$$F(\pi) = \mathbb{E}_{g\sim p(g), s\sim\pi_\theta}\left[-d(s, g)\right], \tag{3.5}$$

where $p(g, s) = p(g)\rho^\pi(s|g)$, $p(g)$ defines the task distribution over goals, and $d(s, g)$ is a distance metric between state s and goal g, such as an Euclidean distance. The main challenges for goal-conditioned RL lie in defining the goal space and the goal-reaching reward function $-d(s, g)$, which often requires task-specific knowledge or careful choices of goal space (Plappert et al., 2018a).

In off-policy learning, hindsight experience replay (HER) (Kaelbling, 1993b; Andrychowicz et al., 2017) has shown to improve learning of goal-conditioned policy significantly. The key insight is that for a given exploration episode $\{g, s_{0:T}\}$, one can relabel the goal with an *actually* achieved goal $\mathbb{S}(s_{0:T})$, derived by a strategy function $\mathbb{S}(\cdot)$. A typical choice is to relabel the goal as $\tilde{g} = \mathbb{S}(s_{0:T}) = s_T$, which can be seen as self-supervised curriculum learning (Andrychowicz et al., 2017; Lynch et al., 2019).

3.4 Expressivity Tradeoffs in Variational Empowerment

Interestingly, the simple objective in Eq. 3.2, which we term *Variational Goal-Conditioned RL* (VGCRL), encapsulates most of the prior MI-based algorithms (Eysenbach et al., 2019; Warde-Farley et al., 2019; Hansen et al., 2020) with the only differences being goal space $\mathcal{Z}$, prior $p(z)$, and discriminator $q_\lambda(z|s)$, as detailed in Table 3.1. For example, when z is a discrete variable, this reduces to DIAYN (Eysenbach et al., 2019) or VALOR (Achiam et al., 2018).

If z is continuous, a natural choice for q_λ is a Gaussian, i.e. $\mathcal{N}(\mu(s), \Sigma(s))$, where both μ and Σ may be parameterized using any function approximators with a range of expressivities, from identity functions to deep neural networks. Throughout the rest of the paper, we show how various simple choices for q_λ lead to algorithms with different properties.

Goal-Conditioned RL as a Coarse Variational Approximation. A simple observation is that if we choose a fixed variational distribution, such as $\mathcal{N}(s, \sigma^2 I)$ with σ as a fixed hyperparameter and the goal space identical to the observation space ($\mathcal{Z} = \mathcal{S}$), the RL objective in Eq. 3.4 becomes (see Appendix B.2 for mathematical details):

$$\mathcal{F}(\pi) = \mathbb{E}_{z,s\sim\pi_\theta}\left[-\tfrac{1}{\sigma^2}\left\|z - s\right\|^2\right] + \text{constant}. \tag{3.6}$$

Method	Goal space	$q_\lambda(z\|s)$	Learnable λ	Learning π_z
GCRL (Kaelbling, 1993b)	Continuous ($\mathbb{R}^d$)	$\mathcal{N}(s, \sigma^2 I)$	-	(HER)
aGCRL **(ours)**	Continuous ($\mathbb{R}^d$)	$\mathcal{N}(s, \Sigma)$	Σ	HER
linGCRL **(ours)**	Continuous ($\mathbb{R}^d$)	$\mathcal{N}(As, \sigma^2 I)$	A	P-HER
InfoGAIL* (Li et al., 2017)	Discrete	Categorical	q_λ	-
DIAYN (Eysenbach et al., 2019)	Discrete	Categorical	q_λ	-
DIAYN (continuous)	Continuous ($\mathbb{R}^d$)	$\mathcal{N}(\mu(s), \Sigma(s))$	$\mu(\cdot)$	-
DISCERN (Warde-Farley et al., 2019)	$= \mathcal{S}$ (e.g. image)	Non-parametric	Embedding$(\cdot)$	HER
VISR (Hansen et al., 2020)	Continuous ($\mathbb{R}^d$)	vMF$(\mu(s), \kappa)$	$\mu(\cdot)$	SF
VGCRL	Any	Any	Any	P-HER or SF

Table 3.1: A summary of algorithms, all are optimized with the single objective in Eq. 3.2. vMF stands for von Mise-Fisher distribution. DIAYN (Eysenbach et al., 2019), DISCERN (Warde-Farley et al., 2019), VISR (Hansen et al., 2020) can be seen as special cases. For InfoGAIL (Li et al., 2017)*, we focus on the MI regularization objective $L_I(\pi, Q)$ only. Since they are under the same objective, learning techniques for goal-conditioned policy π_z such as successor features (SF) (Barreto et al., 2017) and hindsight experience replay (HER) (Andrychowicz et al., 2017) can be adapted for more general settings within the VGCRL objective.

It is straightforward to see that this recovers the objective of GCRL in Eq. 3.5 exactly (up to a constant), where the distance function uses a squared loss. This provides a novel interpretation for GCRL algorithms as a *variational empowerment algorithm with a hard-coded and fixed variational distribution*. Given that no q_λ parameters are adapted, this generally provides a very loose bound on MI; however, prior work on GCRL shows that this RL objective, unlike empowerment-based, learns useful goal-reaching skills stably (Kaelbling, 1993b; Andrychowicz et al., 2017; Pong et al., 2018) thanks to a stationary reward function. This suggests that GCRL and prior variational empowerment methods represent two ends of a spectrum, corresponding to the expressivity of the variational distribution used to approximately maximize mutual information, and neither of the two is perfect, with their own pros and cons. Varying expressivity — through the choices of $\mathcal{Z}$ and q_λ — and evaluating the qualities of learned goal spaces is a central theme of the next section.

3.5 Goal-Conditioned RL as Variational Empowerment

In this section, we discuss GCRL with representation learning, through the lens of variational empowerment: how the representation capacity leads to algorithms with different properties. We first derive two "lost relatives" of GCRL that only add minimal representation capacities but still result

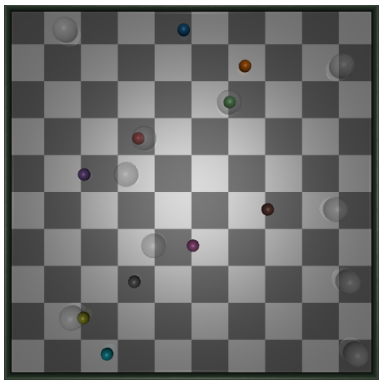
in interesting learning behaviors while keeping the stability of GCRL, and then discuss how we can study representation capacity in more general settings through varying smoothness constraints.

3.5.1 Adaptive Variances for Relevance Determination

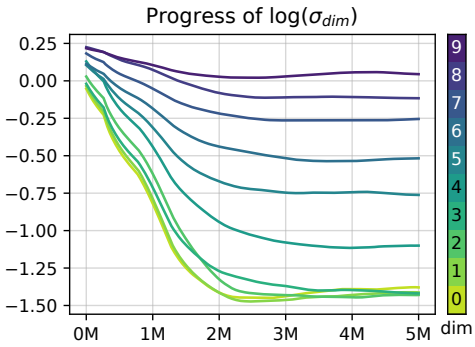
Given the observation in Section 3.4, a straightforward modification to GCRL is to allow the variances to be learned, while keeping $\mu(s) = s$. If we assume a global learned covariance, i.e., $q_\lambda(z|s) = \mathcal{N}(s, \Sigma)$, $\lambda = \{\Sigma\}$, Eq. 3.2 gives us a novel variant of GCRL, which we call *adaptive GCRL* (aGCRL). The intuition behind this algorithm is the following: let us assume a simple diagonal covariance matrix; during learning, this algorithm will quickly shrink σ for the goal dimensions that the agent can reliably reach, and will expand variances for the dimensions that the agent has a hard time to; it therefore can identify and prioritize goal-reaching in feasible directions, discounting unfeasible ones, resembling properties of automatic relevance determination (ARD) (Wipf & Nagarajan, 2008).

Experiment: Automatic Controllability Determination on Windy PointMass. We design a simple *Windy PointMass* environment to study adaptive behaviors, which is simulated in Mujoco (Todorov et al., 2012). We assume a point mass in N-dimensional space (Figure 3.1a), where some dimensions have random force perturbations and therefore are difficult to control. Such perturbations are often studied in the risk-sensitive RL literature (Fox et al., 2015; Maddison et al., 2017); however, in our experiments, they serve to create different levels of controllability. Our goal is to have GCRL automatically ignore dimensions that are not controllable and prioritize dimensions that are easy to control. More details can be found in Appendix B.3.

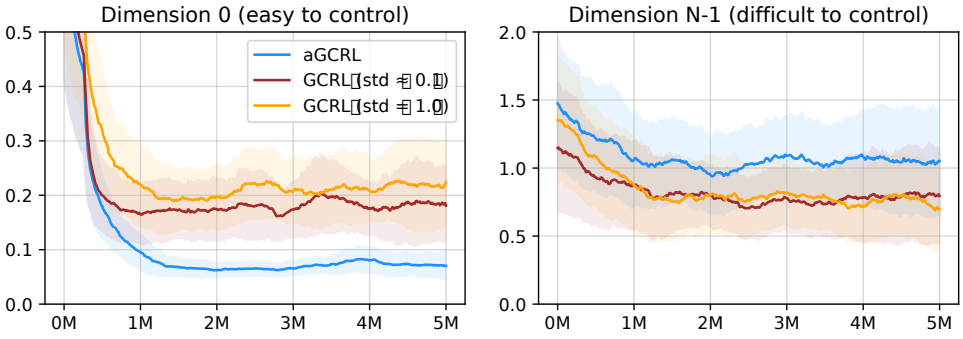
We evaluated goal-conditioned RL with an adaptive global diagonal variance term in Figure 3.1. Our results show that this simple modification to goal-based RL can accurately identify controllable dimensions in the state space. For example, in a 2-dimensional windy pointmass environment, aGCRL recovered a smaller variance for the first dimension, $\sigma_x = 0.368$, and a larger variance for the second dimension, $\sigma_y = 1.648$, which corresponds to having a reward function $r(s, z) = \|x - g_x\|/0.368 + \|y - g_y\|/1.648$ where (x, y) is the position of the point mass and $z = (g_x, g_y)$ is the goal location. A benefit of such adpative variance is that we can prioritize goal reaching in controllable dimensions; in Figure 3.1c, we can observe that aGCRL can reach goals in the controllable dimension (*e.g.*, dim 0) more quickly than the standard constant-variance GCRL baseline on the 10-dimensional Windy PointMass environment, showing the effectiveness of such *automatically learned reward functions that can ignore nuisance dimensions*.



(a) Windy PointMass.



(b) Learned variances (10D).



(c) Goal reaching performance in the controllable dimension (dim 0) and the uncontrollable dimension (dim 9). The y-axis denotes the mean squared error between the goal location and achieve location.

Figure 3.1: Adaptive-variance GCRL. The learned variance is clearly smaller for the easier (noiseless) dimension, which makes goal reaching in controllable dimensions more focused than in uncontrollable ones.

3.5.2 Adaptive Mean with Varying Expressivity

The aGCRL variant adapts variances but fixes the mean $\mu(s)$ to be s. By using more expressive parameterizations, such as neural networks (Eysenbach et al., 2019), the algorithm can theoretically optimize a tighter lower-bound to MI. However, as it gains more expressivity, interpretability and learning stability might be reduced. We study a linear case, i.e. $q_\lambda(z|s) = \mathcal{N}(As, \Sigma)$ where $\lambda = \{A\}$ along with identity and NN cases, and carefully evaluate this design choice.

Experiment: Recovering Intrinsic Dimensions of Variations with Linear GCRL. In this study, we design a simple 2D point mass with a random projection applied to the observation. We use an affine transformation W to generate the agent's observation $o = Ws$ from a physics simulator's state s. For example, when a raw state in 2D point mass (Figure 3.2) includes the (x, y)

location of the point mass, the agent will instead receive an entangled, obfuscated observation: $o = (w_{11}x + w_{12}y, w_{21}x + w_{22}y, \ldots)$ which does not align with the action space. We see whether a variant of VGCRL where $q(z|o) = \mathcal{N}(Ao, \Sigma)$, called *linGCRL*, can recover the inverse of an underlying projection $A = W^{-1}$ when we use a rectangle-shaped 2D uniform prior $p(z)$. This resembles PCA discovering principal components in and underlying true dimensionalities.

Figure 3.2 shows an example where a unknown 2×2 random projection W is applied. The arena of the 2D point mass environment and two example trajectories are visualized: the space of true state s (left), observation $o = Ws$ (middle), and goal $z = (A \cdot W)s$. We can see two intrinsic, orthogonal dimensions are recovered by the learned matrix A such that the posteriors z from marginal states match the prior distribution, but up to rotation and reflection. This was also possible with more complex (e.g., $W \in \mathbb{R}^{10\times 2}$) random projections. We show that linGCRL can *recover intrinsic dimensionalities of the state* from random projections.

3.5.3 Spectral Normalization

If the variational posterior ("discriminator") $q_\lambda(z|s)$ has high expressive power, for example when it is represented by a neural network, it can easily achieve maximum discriminability. However, this can lead to a very suboptimal solution for the policy. Because the marginal states as a result of latent goal and state pairs produced by the latent-conditioned policy are almost random and of poor quality in the early stage of training, the discriminator might easily *overfit* to such a near-random distribution of (s, z), where $s \sim \pi_\theta(\cdot|\cdot, z)$. This usually happens when in general $q_\lambda(z|s)$ is much easier to fit (especially given a high capacity of neural networks) than the policy π_θ. Figure 3.3(a) shows a motivating example of $q_\lambda(z|s)$ learned on a toy 2D point mass environment: the landscape of $q(z|s)$ is highly non-smooth, which can hinder learning latent-conditioned skills due to a ill-

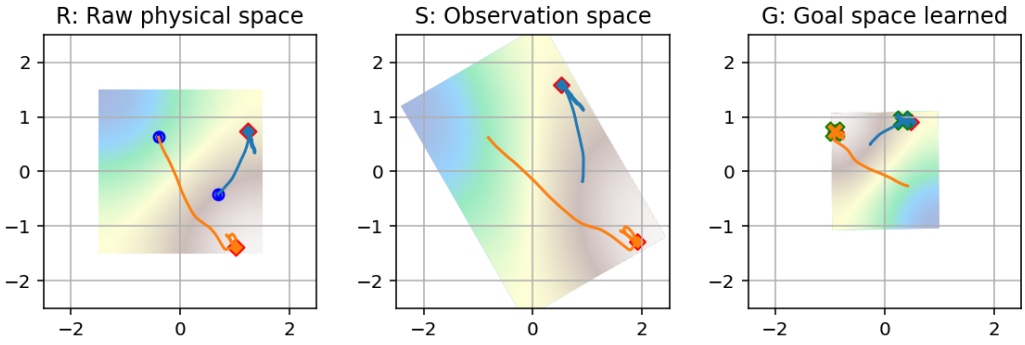
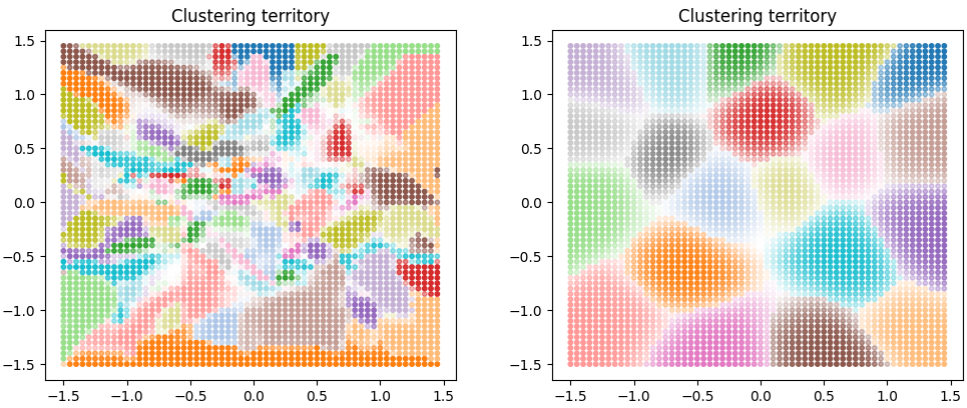


Figure 3.2: linGCRL on 2D point mass. Left: the underlying physics space. Middle: An agent's observation after a random linear projection. Right: A goal space recovered by $\mu = Ao$. The orange/green cross marks denote sampled goals $z \sim p(z)$, and blue dots are initial locations.



(a) Without SN. **(b)** With SN.

Figure 3.3: Visualization of the decision boundary of $q(z|s)$ on 2D point mass environment with $|\mathcal{G}| = 20$ (See §3.5.3).

posed reward structure.

One way to alleviate this issue is to regularize the discriminator using Spectral Normalization (Miyato et al., 2018), which has been very effective in stablizing Generative Adversarial Networks (GAN). Intuitively speaking, spectral normalization enforces the discriminator to be a *smooth* function by satisfying the Lipschitz continuity.

Experiment: Spectral Normalization. We study the behavior of variational empowerment algorithms on a toy 2D point mass environment, for a purpose of simple analysis. To simplify the analysis, we used $\mu(s) = (x, y)$, and a simple 2-layer MLP (with 128 hidden units). Figure 3.3 shows the decision boundary of the discriminator $q_\lambda(z|s)$, without and with Spectral Normalization (SN), where the dimension of z is 20 (or the number of skills). We can observe that, even though in both cases the empowerment objective $\mathcal{F}$ has converged to near-optimum value with a reasonably good discriminability (Table 3.2: top-1 accuracy of $q_\lambda(z|s)$ is $> 90\%$), the decision boundary is much more smooth with Spectral Normalization and closer to interpretations of empowerment (Eysenbach et al., 2019; Mohamed & Rezende, 2015). Without Spectral Normalization, the discriminator had to learn a highly non-smooth decision boundary that is *over-fit* to near-random data generated the premature policy π_z, which would make joint optimization of π_z and $q_\lambda(z|s)$ and discovery of meaningful behaviors difficult.

Furthermore, spectral normalization can improve the performance of variational empowerment algorithms in more challenging control tasks, as will be discussed in Section 3.6.3. These results confirm that *high expressivity does not necessarily mean better performance*, and therefore that inductive biases or *proper regularizations on the posterior* $q_\lambda(z|s)$ *are important* for the performance

2D Pointmass	SN?	$\mathcal{F}$	LGR(z)
$\lvert G\rvert = 10$	-	-0.38	0.94
	✔	**-0.14**	**0.96**
$\lvert G\rvert = 20$	-	-0.86	0.91
	✔	**-0.24**	**0.96**
$\lvert G\rvert = 50$	-	**-1.02**	**0.83**
	✔	-1.15	0.78

Table 3.2: DIAYN v.s. DIAYN + Spectral Normalization (SN) on 2D PointMass (See Figure 3.3). Please refer to Section 3.6.3 for details of the metrics and discussions.

of the algorithm and the quality of the goal representation learned.

3.6 Variational Empowerment as Goal-Conditioned RL

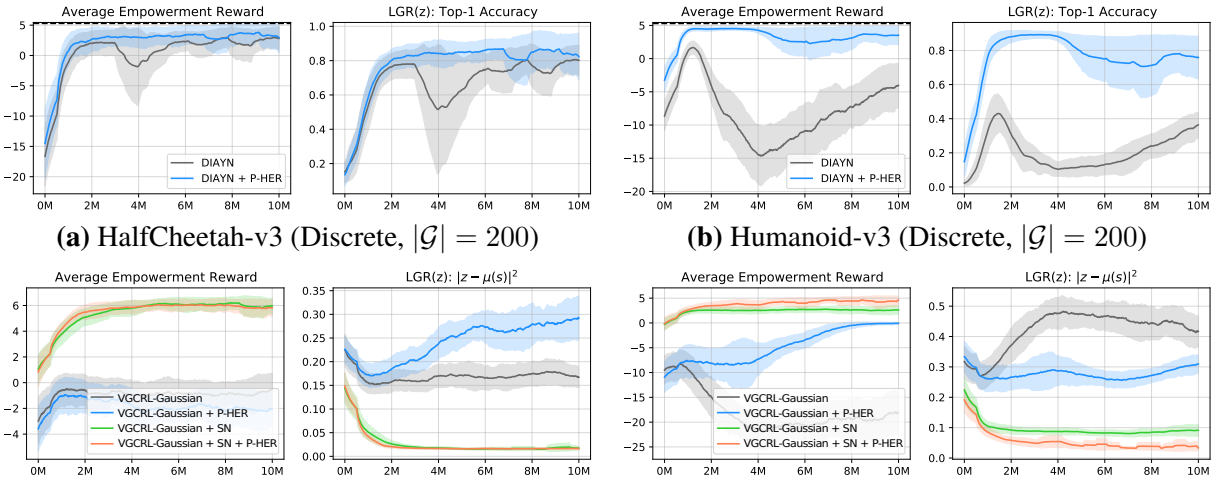
We have seen that goal-conditioned RL can be viewed as a special case of the unified optimization objective in Equation 3.2. A natural question that follows is whether training methods that work well for GCRL can also be used within the more general VGCRL framework. We answer in the affirmative, deriving an extension to a goal-relabeling technique (HER) and an evaluation metric for variational empowerment algorithms. We present some techniques that are helpful for learning VGCRL.

3.6.1 P-HER: Posterior Hindsight Experience Replay

As discussed in Section 3.3.2, Hindsight Experience Replay (HER) (Andrychowicz et al., 2017) substantially improves off-policy learning of goal-conditioned policies and value functions. Inspired by the mathematical connection to GCRL, we can derive an equivalent of HER or goal relabeling in the context of VGCRL. Specifically, we relabel z for fitting the policy $\pi_\theta(a|s,z)$ with the relabeled goal $\mathbb{S}(s_{0:T})$ derived from the final state s_T:

$$\mathbb{S}(s_{0:T}) \sim q_\lambda(z \mid s_T). \tag{3.7}$$

We call this relabeling technique *Posterior HER* (**P-HER**), as it can be seen as an application of posterior sampling (Hausman et al., 2018; Rakelly et al., 2019) to HER. Since q_λ is non-stationary, we always use the *up-to-date* estimate of $q_\lambda(z|s_T)$ with the latest parameter λ of posterior q_λ. This



(a) HalfCheetah-v3 (Discrete, $|\mathcal{G}| = 200$)

(b) Humanoid-v3 (Discrete, $|\mathcal{G}| = 200$)

(c) HalfCheetah-v3 (Gaussian, $|\mathcal{G}| = 5$)

(d) Ant-v3 (Gaussian, $|\mathcal{G}| = 5$)

Figure 3.4: Learning curves (selected) of variational empowerment methods with *discrete* ((a)-(b), see Table 3.3) and *continuous* ((c)-(d), see Table 3.4) goal spaces. We can see that Posterior HER (P-HER) makes optimization of the MI objective faster and more stable, especially when the goal space is large and/or goal-conditioned policy is difficult to learn. The use of spectral normalization (SN, Section 3.5.3) provides performance gains for VGCRL with Gaussian discriminators. The higher LGR(z) is, the better in discrete cases (accuracy; (a)-(b)); the lower LGR(z) is, the better in continuous cases (distance; (c)-(d)). Learning curves are averaged over 3 random seeds. More plots can be found in Appendix B.5.

is the biggest difference to the vanilla HER where the goal mapping function is fixed. We note that other state sampling distributions of HER, e.g., uniform over $s_{0:T}$ or $s_{t>k}$ can also be trivially supported. For fitting the discriminator $q_\lambda(z|s)$, we do not relabel the goal. Similarly to HER in GCRL, this technique can be viewed as a curriculum for enabling parameterized policy optimization to focus on high-reward regions. This technique can *accelerate optimization steps* for π_θ (Equation 3.4) in the same spirit of HER when latent goal reaching is non-trivial, especially in high-dimensional state spaces and with immaturely-shaped reward functions.

3.6.2 Latent Goal Reaching: A Metric for MI-Based Empowerment Algorithms

One limitation in mutual information-based RL (Eysenbach et al., 2019; Sharma et al., 2020b) is lack of objective evaluation metrics. Previous works directly evaluate qualitative results of learned behaviors, but quantitative metrics are limited to discriminator rewards (*i.e.*, $\mathbb{E}[\log q(z|s) - p(z)]$) which can saturate and do not always correspond to the quality of learned behaviors, or downstream task evaluations such as exploration bonus and pre-trained primitives for hierarchical RL (Florensa et al., 2017; Eysenbach et al., 2019; Sharma et al., 2020b; Hansen et al., 2020).

Algorithm 3.1: Latent Goal Reaching Metric: LGR(s)

Input: Target states $s^{1:N}$, trained $\pi_\theta(a|s,z), q_\lambda(z|s)$
Output: Average distance $\bar{d}$ to goal states
for $i \leftarrow 1$ **to** N **do**
  Embed target state into a goal: $z^i \leftarrow \mathbb{E}\left[q_\lambda(\cdot|s^i)\right]$
  Run $\pi(\cdot|\cdot, z^i)$ for T time steps, observe final state s^i_T
  Compute $d(s^i, s^i_T)$ (*e.g.*, squared distance)
end
Report the average over N episodes: $\bar{d} = \sum_i d(s^i, s^i_T)/N$

By contrast, goal-conditioned RL has a clear metric: define a sample of goals and evaluate average goal-reaching performance. Since we demonstrated how MI-based RL is closely related to GCRL, we propose Latent Goal Reaching (LGR) as a new metric for evaluating MI-based algorithms such as DIAYN or DADS. The procedure is described in Algorithm 3.1. *It allows us to evaluate MI-based RL as just another goal-reaching problem*: we measure how accurately the goal-conditioned policy can reach the given goal states of interest. This metric measures the quality of both the discriminator (*i.e.*, goal representation) and the goal-conditioned policy. We note the range of this metric is not dependent on the choice of distribution family of $q(\cdot)$, allowing comparison across different types of $q(z|s)$.

The value of LGR(s) metric depends on the choice of target states. In locomotion control tasks (Brockman et al., 2016), we would often want to discover and learn behaviors where the robot can walk or move (Eysenbach et al., 2019; Sharma et al., 2020b). To evaluate this, one can generate diverse target states of moving in different directions and at different velocities from expert policies. In such cases we can compute the distance $d(s^i, s^i_T)$ between target and achieved states with respect to velocity dimensions (*e.g.*, velocity in x and y axis). We call this variant of the LGR metric as LGR$_v(s)$. Details of target state generation used in our experiments are given in Appendix B.3.

We also consider the LGR(z) metric which measures the goal reaching performance in the latent goal space. For discrete z, this simply can be the top-1 accuracy of the discriminator $q(z|s)$ with respect to the true goal $z \sim p(z)$. For continuous z, LGR(z) is the squared distance $\|z - \arg\max q(z|s)\|^2$ between the goal and the mode of the discriminator $\arg\max q(z|s)$. We note that this metric agrees the state distance metric used in Laplacian embedding (Wu et al., 2018), i.e., $\|\phi(s) - \phi(h^{-1}(z))\|$, with a state embedding $\phi(s) := \arg\max_z q(z|s)$, where $h^{-1}(z)$ is defined to be an arbitrary state s that is associated with latent z, which is in our case the marginal state from the latent-conditioned policy π_θ. A difference is that Wu et al. (2018) use contrastive learning whereas VGCRL maximizes likelihood to learn the representation q.

$\lvert\mathcal{G}\rvert$	Method	HalfCheetah			Ant			Humanoid		
		$\mathcal{F}$	LGR$_v(s)$	LGR(z)	$\mathcal{F}$	LGR$_v(s)$	LGR(z)	$\mathcal{F}$	LGR$_v(s)$	LGR(z)
10	DIAYN	**1.608**	**0.800**	**0.963**	-0.835	0.529	0.806	1.261	0.523	0.922
	DIAYN + P-HER	1.372	1.424	0.934	**-0.049**	**0.486**	**0.889**	**1.856**	**0.312**	**0.953**
20	DIAYN	1.732	**1.125**	**0.920**	-1.308	0.610	0.763	0.713	0.315	0.768
	DIAYN + P-HER	**1.852**	1.214	0.891	**-1.288**	**0.515**	**0.823**	**2.251**	**0.183**	**0.922**
50	DIAYN	1.475	**0.673**	0.827	-3.812	**0.402**	0.523	-1.158	**0.268**	0.549
	DIAYN + P-HER	**1.699**	0.704	**0.834**	**-2.171**	0.637	**0.750**	**2.848**	0.545	**0.891**
200	DIAYN	2.854	**0.698**	0.801	-8.450	**0.396**	0.113	-4.396	**0.208**	0.337
	DIAYN + P-HER	**3.357**	0.814	**0.844**	**-7.047**	0.555	**0.263**	**3.448**	0.866	**0.766**
1000	DIAYN	-8.286	1.156	0.176	-16.795	0.434	0.005	-8.914	**0.325**	0.101
	DIAYN + P-HER	**-4.424**	**0.510**	**0.361**	**-10.941**	**0.322**	**0.028**	**3.395**	1.569	**0.762**

Table 3.3: Evaluation of Latent Goal-Reaching Metric on MuJoCo control suites, after a total of 10M environment steps of training. $\mathcal{F}$ is the (average) empowerment reward, $\mathcal{F} = \mathbb{E}_z[\log q_\lambda(z|s) - p(z)]$. LGR$_v(s)$ is the squared error in observation space (the lower, the better) with respect to velocity dimensions between the marginal state and the target state, and LGR(z) denotes the accuracy of top-1 classification of the discriminator $q_\lambda(z|s)$ (the higher, the better, max 1.0).

3.6.3 Experiments: Posterior HER and Latent Goal Reaching

In this section, we evaluate the performance of several variants of VGCRL on standard locomotion tasks (Brockman et al., 2016). We consider both a discrete latent space, analogous to that used by DIAYN (Eysenbach et al., 2019), and a continuous latent space, where the variational posterior $q_\lambda(z|s)$ chosen to be a Gaussian posterior (VGCRL-Gaussian), with either learnable or fixed variances, or Gaussian Mixtures (VGCRL-GMM). To evaluate the performance, we report the following metrics: (1) the empowerment objective $\mathcal{F} = \mathbb{E}_z[\log q(z|s) - p(z)]$, (2) LGR($z$): the latent goal reaching metric, and (3) LGR$_v(s)$: the latent goal reaching metric with respect to velocity dimensions (Section 3.6.2).

We first observe that P-HER can accelerate and improve learning (Figure 3.4, Tables 3.3 and 3.4). Such improvements are significant in high-dimensional goal spaces (*e.g.*, $|\mathcal{G}| = 200$) and more difficult control tasks such as Ant or Humanoid with high dimensionalities, in which both the discriminator and the latent-conditioned policy are difficult to learn. Because an optimization of goal-conditioned policy (Eq. 3.4) is more difficult than discriminators (Eq. 3.3), relabeling of goal can greatly accelerate RL, which also results in better discriminability. As shown in Tables 3.4 and B.1, **P-HER** can improve not only the optimization objective but also other metrics such as discriminator's accuracy and goal reaching performance across different design choices and environments.

	$q_\lambda(z\|s)$	P-HER?	SN?	HalfCheetah			Ant			Humanoid		
				$\mathcal{F}$	LGR$_v(s)$	LGR(z)	$\mathcal{F}$	LGR$_v(s)$	LGR(z)	$\mathcal{F}$	LGR$_v(s)$	LGR(z)
$\|\mathcal{G}\| = 5$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	0.932	1.005	0.159	-0.590	1.005	0.382	0.239	1.461	0.202
		✔	-	-0.142	1.273	0.360	0.140	2.449	0.300	0.020	1.452	0.244
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-0.731	1.251	0.172	-18.490	**0.306**	0.427	-3.597	0.538	0.147
		✔	-	-2.161	1.132	0.289	-0.108	2.423	0.303	1.207	0.206	0.074
		-	✔	**5.856**	**0.604**	0.019	2.548	0.925	0.091	4.509	0.460	0.040
		✔	✔	5.803	1.352	**0.017**	**4.349**	0.463	**0.039**	**5.203**	**0.203**	**0.026**
	GMM ($K = 8$)	-	-	-2.646	0.766	0.325	-16.196	0.367	0.486	-3.576	0.231	0.198
		✔	-	-3.091	1.065	0.404	-2.874	2.794	0.325	3.526	0.581	0.043

Table 3.4: Comparison of continuous variants of VGCRL, where the dimension of the goal space is $|\mathcal{G}| = 5$. SN denotes Spectral Normliazation (Section 3.5.3). LGR(z) is the goal reaching performance in the latent space (the lower, the better). A full table containing more comprehensive comparison and corresponding plots can be found in Appendix B.5 (Table B.1).

Moreover, when Spectral Normalization (Section 3.5.3) is applied, we observe a significant improvement in terms of the learning progress and evaluation metrics (Figure 3.4, Table 3.4). As shown in Figure 3.4, while there are some progress with vanilla VGCRL-Gaussian (or VGCRL-GMM), the optimization objective $\mathcal{F}$ as well as evaluation metrics do not improve as much as the one with SN, despite the *bigger expressivity* due to no constraint on $q_\lambda(z|s)$. We can also see that with SN (and P-HER as well), the distance between achieved and desired goal is much lower. More experimental results and discussions can be found in Appendix B.5.

3.7 Conclusion

Our variational GCRL (VGCRL) framework unifies unsupervised skill learning methods based on variational empowerment (Eysenbach et al., 2019; Gregor et al., 2017; Sharma et al., 2020b) with goal-conditioned RL (GCRL) methods, allowing us to transfer techniques and insights across both types of approaches. Viewing GCRL as variational empowerment, we derive simple extensions of goal-based methods that exhibit some representation learning capability of variational methods, e.g., disentangle underlying factors of variations and automatically determine controllable dimensions, while keeping the learning stability of GCRL. Viewing variational empowerment as GCRL, we can transfer popular optimization techniques such as relabeling from GCRL to variational empowerment algorithms, and propose latent goal-reaching (LGR) as a more objective, performance-based metric for evaluating the quality of skill (latent goal) discovery. We hope that these insights can lay the ground for further developments of more capable and performant algorithms for unsu-

pervised reinforcement learning in future work.

CHAPTER 4

Unsupervised Object Interaction Learning with Counterfactual Dynamics Models

We present COIL (Counterfactual Object Interaction Learning), a novel way of learning skills of object interactions on entity-centric environments. The goal is to learn primitive behaviors that can induce interactions without external reward or any supervision. Existing skill discovery methods are limited to locomotion, simple navigation tasks, or single-object manipulation tasks, mostly not inducing interaction between objects. Unlike a monolithic representation usually used in prior skill learning methods, we propose to use a structured goal representation that can query and scope which objects to interact with, which can serve as a basis for solving more complex downstream tasks. We design a novel counterfactual intrinsic reward through the use of either a forward model or successor features that can learn an interaction skill between a pair of objects given as a goal. Through experiments on continuous control environments such as Magnetic Block and 2.5-D Stacking Box, we demonstrate that an agent can learn object interaction behaviors (e.g., attaching or stacking one block to another) without any external rewards or domain-specific knowledge.

4.1 Introduction

Reinforcement learning (RL) has achieved remarkable progress at many application domains such as playing games (Mnih et al., 2013; Vinyals et al., 2019), and robotics control (Andrychowicz et al., 2020), etc. Very often RL agents are trained to specific tasks, with access to task-specific *extrinsic* rewards. A major drawback of task-specific training is that a proper reward function needs to be given, designed, and tuned so as to achieve desired behaviors, which can be often time-consuming and limits scalability in practice. It is important to be able to solve the task with a very sparse reward signal upon completion/failure of the task, or even without any external task rewards. Unsupervised RL such as task-agnostic exploration or pre-training of skills, aiming at learning interesting or useful behaviors without the use of task rewards or offline data, can provide better initialization or useful

macro-actions (skills or options) for building a hierarchical agent to solve more complex and difficult tasks (Eysenbach et al., 2019; Zhang et al., 2021). Unsupervised learning often enables faster learning and achieves better generalization performance when multiple tasks are given after the skill acquiral or pre-training phase.

Despite a number of successes in unsupervised skill discovery (Eysenbach et al., 2019; Sharma et al., 2020b; Park et al., 2022) or task-agnostic exploration based on state-entropy maximization or diversity (Pathak et al., 2017; Burda et al., 2019b), relatively only a few attempts have been made on environments and tasks with *multiple entities* (e.g. objects in robotics manipulation). In the context of robotics manipulation or (discrete) entity-centric environments other than locomotion or maze navigation environments, exploration can be quite challenging because of this nature of multiple entities. One limitation of novelty-seeking exploration methods in the reward-free context is that exploration would easily converge to a low-hanging fruit behavior where exploration mostly focuses on one particular entity. For instance, in robotics manipulation environments, diversification or novelty seeking of the entire state can be easily dominated by that of the embodied agent itself (i.e., proprioceptive states) or some easy-to-control objects only, as observed and reported in (Zhao et al., 2021; Gu et al., 2021; Park et al., 2022). More interesting primitive behaviors would be interactions between many objects, for more realistic and challenging multi-object tasks such as block stacking (Lee et al., 2021; Sancaktar et al., 2022) or furniture assembly (Lee et al., 2019b; Ghasemipour et al., 2022). Notably, some recent works including (Sancaktar et al., 2022; Cho et al., 2022) present reward-free exploration and skill learning in multi-object manipulation tasks.

In this work, we focus on learning a set of primitive skills that enable interaction between different objects in a task-agnostic, unsupervised fashion. Roughly speaking, interaction between two objects can be described as an action or event that occurs when two objects have a (mutual) effect on each other. Our work leverages an inductive bias that an interaction between objects learned in a task-agnostic manner can be a useful event and hence a useful primitive behavior for solving downstream tasks. Such object-object interactions (as well as agent-object interactions) are usually sparse and difficult to reach with naive exploration, but at the same time can be useful bottleneck states an agent would want to explore and visit often to achieve bigger tasks. In the kitchen, for instance, an interaction between a knife and various ingredients by slicing them with a knife can be one of the basic steps necessary for cooking; when assembling smaller building blocks to build a complex object like furniture, car, or electronic device, connecting matching pieces to form a composite body would be another type of indispensable interaction. As such, it will be important to learn *skills* or primitive behaviors that would induce object-object interactions, in the promise that a hierarchical control that acts upon the interaction skills (Zhang et al., 2021) or chaining of skills in sequence (Bagaria & Konidaris, 2020) should solve complex tasks much faster than flat RL agents.

We study how to learn object interaction skills in a very challenging, online, reward-free setting while minimizing the use of domain and task-specific knowledge or task-agnostic offline data, which can be difficult to obtain. More specifically, we learn a goal-conditioned policy where a goal denotes an interaction of which objects is to be made. To enable this by *learning a reward function* (Zheng et al., 2020), we design a novel intrinsic reward that is computed by counterfactual reasoning on the dynamics model (forward model and successor features)as will be explained in Section 4.2.2, which we call Counterfactual Object Interaction Learning (COIL).

The concept of counterfactual reasoning, i.e., "*what if...?*" — predicting or inferring the outcome if something had happened differently (Mesnard et al., 2020; Gajcin & Dusparic, 2022) — naturally aligns with an intuitive interpretation of interaction: interaction is when an object's future state would have been different if it were not for the presence of the other object. In the experiments, we show that the intrinsic reward derived by counterfactual reasoning on object states can efficiently induce the interaction of objects and enable an RL agent to learn such interaction behaviors without extrinsic rewards.

Our contribution can be summarized as follows:

- We study a setting of representing goals in terms of entities and objects to interact with, in the context of unsupervised skill-based and goal-conditioned RL.
- We present a novel intrinsic reward algorithm COIL (Counterfactual Object Interaction Learning) in a reward-free unsupervised exploration setting, which uses counterfactual reasoning on forward model or successor features. We show COIL can learn skills that make the goal objects interact with each other.
- We show that such an entity-centric interaction skill is generalizable to unseen, more object settings.

4.2 Approach

4.2.1 Preliminaries and Notations

Throughout the paper, we consider the task as an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, $\mathcal{P}$ is a transition probability, $\mathcal{R}$ is a (extrinsic) task reward function, and $0 \leq \gamma < 1$ is a discount factor. Our goal is task-agnostic, unsupervised skill learning with no extrinsic rewards. We assume that the state space $\mathcal{S}$ can be explicitly factorized as the Cartesian product $(\mathcal{S}_{\text{object}})^N \times \mathcal{S}_{\text{agent}}$ where N is the number of objects, $\mathcal{S}_{\text{object}}$ is the object state space, and $\mathcal{S}_{\text{agent}}$ is the agent state space. We also assume the joint object space is permutation-invariant, i.e., $\{o_1, \cdots, o_N\}$ is a *set* (where $o_i \in \mathcal{S}_{\text{object}}$). Such a structural representation is common in robotics

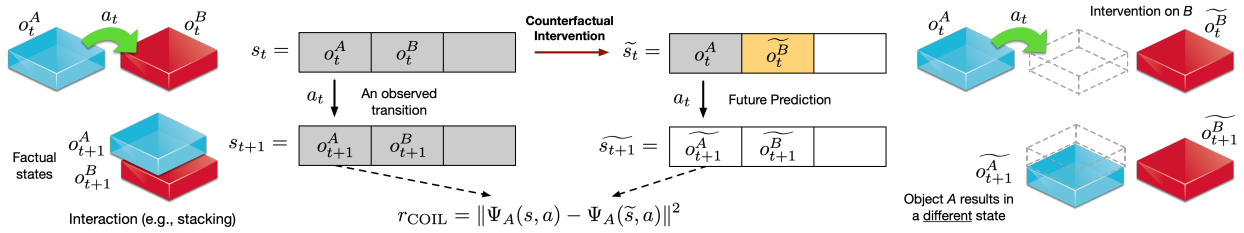


Figure 4.1: (i) Suppose an interaction was made, then a counterfactual intervention on object B (e.g., putting it aside or change the object state randomly) would have made the future state of object A different. (ii) If no interaction was made, object A would remain in the same state regardless of the counterfactual intervention. (iii) We measure the discrepancy of object A with and without the counterfactual intervention, which becomes the intrinsic reward for interaction.

control (Keramati et al., 2018; Zhao et al., 2021; Sancaktar et al., 2022) and is a mild assumption. However, our method is not necessarily limited to state-based control only, as one could combine with existing entity-centric representation learning methods from pixel observations (Watters et al., 2017; Greff et al., 2019; Xu et al., 2019; Veerapaneni et al., 2020; Locatello et al., 2020).

Goal representation. Skills are usually modeled in the form of goal-conditioned policies, $\pi(a|s, g)$, where $g \in \mathcal{G}$ represents a *goal*. Common choices for goal g include full state observation, a handcrafted goal with domain knowledge, or latent variables. Our particular choice is a pair of objects, namely A and B (among the N objects). A semantic meaning for this goal representation would be that two objects A and B should have an interaction (or some mutual effects) as a consequence of agent's actions. In our settings, for the sake of simplicity, we assume the reference to objects are simply categorical indices (or pointers), i.e., $A, B \in [N] = \{0, 1, \cdots, N-1\}$, respectively. However, more in general (e.g., for image observations), the goal representation for target objects can be replaced with a continuous vector to represent a reference to an *arbitrary* object in the current state, e.g., $g = (o^A, o^B)$ where $o^A, o^B \in \mathcal{S}_{\text{object}}$, which we leave as a future work.

4.2.2 Learning Interaction Skills with Counterfactual Forward Model (COIL-Forward)

How can we learn interaction skills for two given objects, and how can we learn a reward function that would incentivize interactions between two objects? Our goal is to simultaneously learn such a reward function and object-object interaction skills in a *reward-free* setting.

Our main idea is to use a *counterfactual reasoning*; i.e., predict what would have happened instead if other objects involved in an interaction were not there or were in a different state. We argue that this form of inductive bias can provide us with a useful learning signal for interaction

learning without relying on an external task reward.

Given a trajectory of observations as object states, we want to identify whether an interaction between two objects happened or not. Roughly speaking, we can say a (physical) interaction occurs between two objects A and B if and only if there exists a counterfactual state for B that would change the future state of A (and vice versa). (*Case* I) When an interaction between A and B happened, these two objects would have affected each other's state. In other words, the future state of an object would have been different without a specific configuration of the other object, provided that an interaction happened. (*Case* II) On the contrary, when there was no interaction between them, the future state of an object would remain almost the same or not dramatically different regardless of the counterpart object. A motivating example is depicted in Figure 4.1.

We can formalize this idea as follows. Consider a MDP transition observed by an agent, (s_t, a_t, s_{t+1}) where $s_t = \{o_t^A, o_t^B, \cdots\}$ and $s_{t+1} = \{o_{t+1}^A, o_{t+1}^B, \cdots\}$ (without the loss of generality) for a pair of objects A and B given as a goal g. We would want to tell whether an interaction was made in this transition.

(*Case* I) Suppose an interaction between object A and B happened, where A got affected by B in the interaction (without the loss of generality). Then, if we made an *counterfactual intervention* on the object B, i.e., changing the object state o_t^B randomly with $\widetilde{o_t^B}$ to obtain an *intervened state* $\widetilde{s_t} = \{o_t^A, \widetilde{o_t^B}, \cdots\}$, the same action a_t applied on $\widetilde{s_t}$ would have resulted in a different (counterfactual) next state $\widetilde{o_{t+1}^A}$ of object A than its (factual) next state o_{t+1}^A. In other words, the discrepancy between the factual next state o_{t+1}^A and the counterfactual next state $\widetilde{o_{t+1}^A}$ will be high.

(*Case* II) On the other hand, when there was no interaction happened between the two in this transition, we can expect that o_{t+1}^A would remain the same regardless of the intervention $\widetilde{o^B}$ on B, i.e., it would be that $\widetilde{o_{t+1}^A} = o_{t+1}^A$. To put together, the difference between o_{t+1}^A and $\widetilde{o_{t+1}^A}$ (e.g., $\|o_{t+1}^A - \widetilde{o_{t+1}^A}\|^2$) can quantize the degree of an interaction between objects A and B.

However, the counterfactual next state $\widetilde{s}_{t+1}$ is not observable by an agent. So we can instead predict the object A's next state by learning a forward dynamics model:

$$\widehat{o_{t+1}^A} = f_{\text{forward}}(o_t^A, \widetilde{o_t^B}, a_t, s^t \setminus \{o_t^A, o_t^B\}) \tag{4.1}$$

This gives us a counterfactual interaction reward function: computationally, we first make a random intervention $\widetilde{o_t^B}$ on object B, and plug it to the forward model to predict the next state $\widetilde{o_{t+1}^A}$ of object A. Intervention on the object B can be implemented in many ways, such as random perturbation of the state vector by adding Gaussian noises, but an easy yet effective way to yield in-distribution randomization is to randomly sample an object state from the replay buffer.

Finally, we define the counterfactual interaction reward $r_{\text{COIL-Forward}}(s_t, a_t, s_{t+1}) = \|o_{t+1}^A - \widehat{o_{t+1}^A}\|^2$, which can be maximized by any underlying RL method (e.g., SAC or DQN) with a si-

multaneous learning of the forward model and the goal-conditioned policy π. We call this resulting algorithm **COIL** (Counterfactual Object Interaction Learning) and specifically this variant of using forward model **COIL-Forward**.

4.2.3 Learning Interaction Skills with Counterfactual Successor Features (COIL-SF)

In this section, we will present an improvement to COIL-Forward, called **COIL-SF**. One downside of the COIL-Forward (Section 4.2.2) is that it assumes the counterfactual intervention would have an immediate, easily distinguishable change within a single-step transition. In many realistic environments, the effect and consequence of interaction is *delayed* to be discernible enough; the change actually exists in the true world state but an observer would not be able to recognize the subtle difference until a few time step has elapsed. Therefore, it is practically important to take long-term futures into consideration so as to correctly evaluate the consequence of counterfactual interventions.

One natural way to deal with this problem would be to learn a multi-step, recurrent forward dynamics model (Oh et al., 2015). However, learning such a forward model can be challenging due to high uncertainty and the quick accumulation of prediction errors over a long horizon (Moerland et al., 2020; Lutter et al., 2021). Instead of learning a multi-step forward model, we propose to use the successor features (SF) (Dayan, 1993; Barreto et al., 2016) to incorporate long-term futures that can still derive a reward signal for interaction learning.

A successor feature $\Psi^\pi(s, a)$ of a state s with respect to a policy π is an expected discounted sum of the feature of future states to be visited when starting from the state s and the action a, and following the policy π thereafter:

$$\Psi^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \Phi(s_t) \;\middle|\; s_0 = s, a_0 = a \right]. \tag{4.2}$$

where $\Phi(s_t)$ is called the cumulant, which is the feature of future states to accumulate. Successor features can be seen as an instance of generalized value functions (GVF) (Sutton et al., 2011) that *predicts the future* and *summarize* what will happen in the future for a state s in some specific form, which can be easier than directly predicting the next states accurately. Successor features can be learned using simple TD learning methods like Q-learning (Dayan, 1993).

To derive a reward function that tells whether an interaction is made or not, let's again consider two objects A and B given as a goal g, and focus on the future of object A when a counterfactual intervention is made on the object B. For this, we consider an entity-centric successor feature with

an object cumulant function $\phi : \mathcal{S}_{\text{object}} \to \mathbb{R}^d$ for the target object o^A in the state observation s :

$$\Psi^\pi_A(s,a) = \Psi^\pi_A(\{o^A, o^B, \cdots\}, a) \quad = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \phi(o^A) \;\middle|\; s_0 = s, a_0 = a \right] \tag{4.3}$$

for a query state $s = \{o^A, o^B, \cdots\}$. The SF $\Psi^\pi_A(s,a) \in \mathbb{R}^d$ summarizes the future state of object A with respect to the policy π. In general, one could learn the object representation $\phi(\cdot)$ with some auxiliary objectives (e.g., in pixel-based control), use some prior domain knowledge, or use a fixed random function as in (Zhang et al., 2019), but in a state-based control one can simply set $\phi(o^A) = o^A$ without an extra need to learn the cumulant feature function.

The reward function can be derived as in COIL-Forward: let's suppose we make a counterfactual intervention on object B at timestep t to get the intervened object state $\widetilde{o^B}$ from o_B. Denoting $\widetilde{s} = \{o^A, \widetilde{o^B}, \cdots\}$, the reward function for interaction can be written as

$$r_{\text{COIL-SF}}(s,a,s') = \|\Psi^\pi_A(s,a) - \Psi^\pi_A(\widetilde{s},a)\|^2. \tag{4.4}$$

We call this variant of using successor features for learning interactions **COIL-SF**. This reward also can be explained as follows: (*Case* II) When there was no interaction happened between objects A and B, the entity-centric successor features Ψ^π_A will be the same regardless of the intervention, in which case $r_{\text{COIL-SF}}$ would be 0. Note that, in practice, rewards for non-interaction transitions might be slightly bigger than 0 due to the epistemic uncertainty of the model. (*Case* I) On the other hand, if the future state of the object A would have changed much due to the intervention on object B, the SF values $\Psi^\pi_A(s,a)$ and $\Psi^\pi_A(\widetilde{s},a)$ will be different, in which $r_{\text{COIL-SF}}$ will evaluate to a higher scalar value. Section 4.4.5 presents an analysis of the learned reward function for different types of transitions (e.g., a high reward is indeed given when interaction happens). Learning of COIL-SF also involves a simultaneous optimization of SF and policy;

4.3 Related Work

Object-Oriented RL. Object-oriented RL (Diuk et al., 2008) aims at improving data efficiency and generalization by leveraging representation of multiple objects and their relations. C-SWM (Kipf et al., 2019) proposes a GNN-based network to learn the world model of the object-based task using contrastive learning. Compared to models based on pixel reconstruction, C-SWM provides a rich representation of objects. CEE-US (Sancaktar et al., 2022) utilizes the epistemic uncertainty of structured world model (Kipf et al., 2019) as an intrinsic reward and uses it to gather data for the world model training. The world model is then used for planning to solve downstream tasks. The behavior that emerges during world model training is mostly object manipulation rather than

interactions between objects and their algorithm can learn object-object interaction only when an extrinsic reward is provided.

Exploration. Cho et al. (2022) proposed a mutual-information (MI) based exploration algorithm to induce interactions between the *agent* and an object, which combines the MUSIC objective (Zhao et al., 2021), i.e., MI between agent and object, and the diversity term similar to DADS (Sharma et al., 2020b) for the object's future state. Seitzer et al. (2021) used object-centric causal action-influence as an intrinsic reward. However, interactions between different objects are not considered, and the skills are limited to simple control of a single target object specified by the task. Very recently, Sancaktar et al. (2022) proposed curiosity-based exploration algorithm that learns a GNN-based world model, with the intrinsic reward being the epistemic uncertainty through ensemble disagreement (Pathak et al., 2019). This work is the closest to our work, but despite GNN's ability to generalize to multiple objects during planning, their monolithic skill representation is limited to be useful for hierarchical learning or planning.

Several papers have proposed exploration methods using successor features (SF). Zhang et al. (2019) use the difference of SF between consecutive states as an intrinsic reward to efficiently explore bottleneck states. Machado et al. (2020) propose an inverse of the L1-norm of the SF as a variant of count-based exploration. Hoang et al. (2021) utilize SF to define the distance function between states and learn a goal-conditioned policy to drive exploration. However, to the best of our knowledge, SF has not been used in object-centric environments and has not been combined with counterfactual reasoning.

Counterfactual Reasoning in RL. Buesing et al. (2018) use a structural causal model in POMDP, which generates counterfactual trajectories for background planning, leading to a better sample efficiency and smaller bias of the prediction in guided policy search. Sharma & Kroemer (2020) utilize an inductive bias that, in similar scenes, if similar action has been taken it would give similar results. They utilize contrastive learning in object-centric tasks to acquire an object relation model, which is subsequently utilized in real-world precondition learning tasks. Counterfactual Credit Assignment (Mesnard et al., 2020) utilizes counterfactual reasoning on action to achieve unbiased, low variance credit assignment. Most approaches do counterfactual inference on the agent's action, i.e., concerns what would have happened if the agent made a *different decision* (i.e., action or goal); our approach differs in the sense that our counterfactual intervention is made on the object states instead of the agent's action.

4.4 Experiments

4.4.1 Environments

In the experiments, we test our proposed approach on multi-object continuous control environments: a toy environment (**StackingBox**) and more challenging environment (**Magnetic Blocks**).

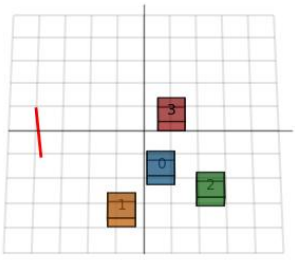
Stacking Box. Stacking Box is a 2.5-D continuous control environment in which a cursor agent and multiple box-shaped objects of the same size are randomly spread throughout a fixed arena. The agent can move in any direction within the xy plane and can grab an object that overlaps with the agent. If the agent moves towards an object while holding another object, the object being held and moved will be placed on top of any other existing object. We assume that the height of each object is quantized to integer values (such as $0, 1, 2, \ldots$). The process of stacking one object onto another occurs instantly in a single MDP transition.

Magnetic Blocks. Magnetic Blocks is a continuous control environment in which an embodied cursor agent can interact with square-shaped block objects. The agent has a continuous action space that includes movement (translation), rotation, and grabbing through control of the joint's torque. The agent can move freely within the arena and can grab an adjacent object by slightly lifting up and moving around the object, or rotating it along with the agent. When the agent moves a held object close enough to another object such that the two objects become parallel, they will be connected by magnetic force. If the edges are not parallel, one object will push the other. A distinctive interaction in this environment is observed when two objects become connected through magnetic forces and then move together.

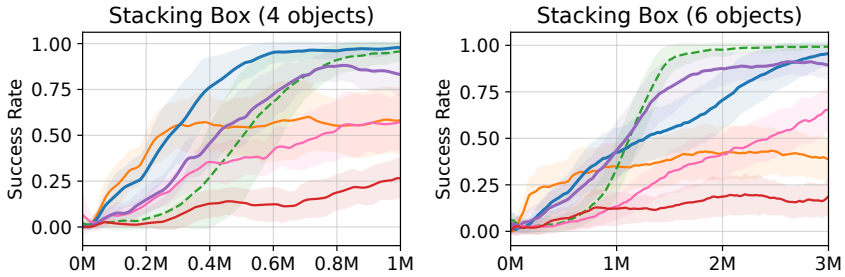
4.4.2 Implementation Details

The full network architecture for the policy and the model is shown in Figure **??**. Taking the factorized state representation into consideration, we use a network with scaled dot-product attention architecture (Vaswani et al., 2017) to transform object states into desired outputs (actor, critic, and forward/successor models). We note that the shared parameters for key and value matrices on the $N - 2$ objects other than the goal objects allows the network to be permutation-invariant over their ordering, and that such an architecture allows generalization to a different number of objects(see Section 4.4.6).

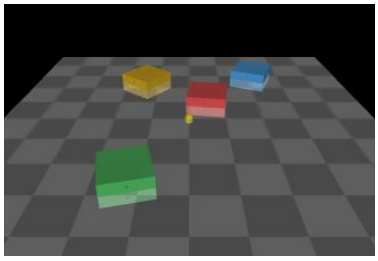
COIL alternatingly updates the policy (actor and critic) and the model (forward model or successor features); for RL algorithm, we use SAC (Haarnoja et al., 2018) although COIL can be combined with any RL algorithms.



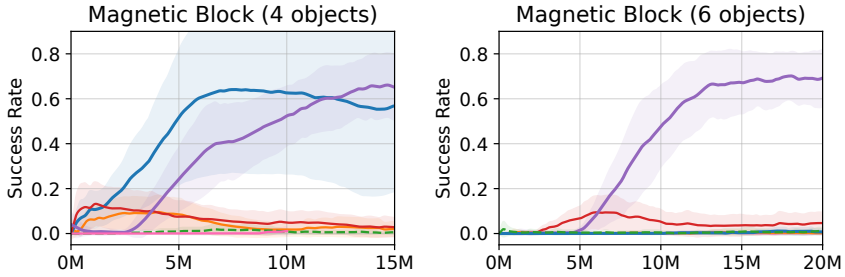
(a) Stacking Box



(b) Performance on Stacking Box



(c) Magnetic Blocks



(d) Performance on Magnetic Blocks

Figure 4.2: Progress of the success rate on the Stacking Box and the Magnetic Blocks environment. Runs are averaged over 5 random seeds. See Section 4.4.3 for analyses and interpretation of the result.

4.4.3 Performance of Learning Object Interaction Skills: Quantitative Results

We first study how well the proposed approach (COIL) can learn object interaction skills in a reward-free setting, with a comparison to strong exploration methods. At the beginning of every episode, a goal $g = (A, B)$ is chosen randomly to specify which objects should interact.

Baselines. (1) Sparse-GT: A SAC agent trained to maximize the *sparse* ground-truth interaction reward, where the per-step reward is 1 if a correct interaction between the target objects is made (e.g., stacking or magnetic connection) or 0 otherwise, which is the same as the success metric. (2) Forward-Curiosity: this maximizes the prediction error of the forward model for object A as an intrinsic reward: $\|o^A_{t+1} - f_{\text{forward}}(s_t, a_t, g_t)\|^2$. (3) SID (Zhang et al., 2019): this maximizes the "successor feature control" reward: $\|\Psi(o^A_{t+1}) - \Psi(o^A_t)\|^2$. (4) RND (Burda et al., 2019b): this maximizes the prediction error of a randomly initialized network's feature representation of the

target object's state as an intrinsic reward: $\|f_{\text{random}}(o^A_{t+1}) - f(o^A_{t+1})\|^2$.

For object-centric tasks, interactions can lead to significant changes in the object's state, making it desirable to employ curiosity-based exploration methods as baselines. RND is a state-of-the-art exploration method that seeks novel states, and Forward-Curiosity and SID are curiosity-based exploration techniques that use the Forward Model and Successor Feature, respectively.

Quantitative Results. The success rate of the algorithms is displayed in Figure 4.2, based on the evaluation episodes. Successful outcome is defined as the stacking of one object on the other in Stacking Box and the connection of the two selected goal blocks in Magnetic Blocks.

Stacking Box. COIL agents converge to a success rate of approximately 1.0, while curiosity-based exploration methods show limitations with upper bounds in their performance. One thing to note is that COIL-Forward outperforms COIL-SF in Stacking Box with 4 objects. In the Stacking Box environment, interactions occur instantaneously, enabling the 1-step forward model of COIL-Forward capture the occurrence of the interaction. This is supported by an analysis of the error of the dynamics model. (see Figure C.4). Transitions involving interactions exhibit a significantly higher ratio of the counterfactual prediction error (i.e., the prediction error when counterfactual intervention is made) to epistemic uncertainty, compared to transitions without events. On the other hand, Forward-Curiosity, SID, and RND are limited to manipulating individual objects without learning interaction stably (see Figure C.2).

Magnetic Blocks. COIL-SF is the only algorithm that successfully learns interaction skills between objects. Despite leveraging domain knowledge regarding the occurrence of interactions, Sparse-GT fails to learn even the basic task of grabbing an object (see Figure C.3). Forward-Curiosity, SID, and RND can learn how to grab an object but interaction between the objects barely happen. This suggests that learning to induce interactions between objects in Magnetic Blocks is a challenging exploration problem, unlike the Stacking Box environment.

We find COIL-Forward is not effective enough to learn interactions in Magnetic Blocks, which accords with the motivation discussed in Section 4.2.3. In this environment, interactions make only a subtle difference in the object's state during a single-step transition and can be better discerned only in longer-term future; we verify this by analyzing the dynamics model errors (see Figure C.5). When interactions occur, the counterfacutal prediction error is not significantly higher than the epistemic uncertainty in the forward model (in COIL-Forward). However, the counterfactual prediction error of the successor feature (in COIL-SF), is significantly higher than the epistemic uncertainty despite the counterfactual intervention, so the interaction reward could lead to learning interactions.

4.4.4 Qualitative Results

In Stacking Box, a typical interaction behavior for $g = (A, B)$ that COIL learns is to stack object A on object B. Note that A should be on top of B (i.e., bigger z coordinate) to say interaction happened. If the B were on top of A, changing or perturbing the state of B would not affect the A's state. An interesting finding was that the agent repeatedly stacked and unstacked the boxes, resulting in multiple interactions within a single episode.

In Magnetic Blocks, the interaction behavior is to grab object A and connect it to object B by making some movements and rotations as needed. Note that the agent needs to rotate objects accurately to connect the blocks, which makes the environment require some good exploration strategies to successfully learn object-object interactions. We present snapshots of COIL-SF's typical behaviors in Magnetic Blocks in Appendix (Figure C.1). Typically, the agent grabs the object A and approaches the object B to make these two objects connected to each other, and pushing them further to move the compound around.

4.4.5 Analysis of COIL-SF Reward

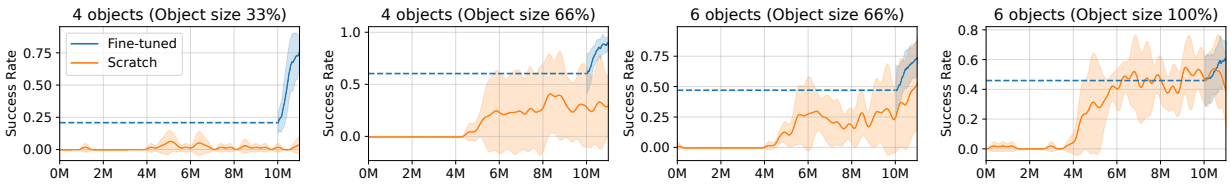
To analyze what reward function COIL-SF has learned, we labeled each state with the following 7 categories on the **Magnetic Blocks** environment with 4 objects.

- Grab-A: the agent is grabbing the object A.
- Grab-B: the agent is grabbing the object B.
- Connect-AB: the objects A and B correctly connected. Note that when A and B are connected, the object A will be highly likely to be be affected by the object B, i.e., interaction occurs.
- Connect-AB-Only: a subset of Connect-AB states, excluding states where objects other than A and B are connected as well.
- Connect-AX: the object A is connected to a wrong object (X), i.e., anything but to B. This is a falsy interaction that does not conform to the goal given to the agent.
- Connect-BX: the object B is connected to a wrong object (X), i.e., anything but to A. This is also a falsy interaction that does not conform to the goal given to the agent.
- No-Event: all other states not included in the above 6 categories (e.g., the agent wanders around and does nothing)

Table 4.1 shows an average reward given to states with each label, for a successful instance of COIL-SF. Among the 7 labels, Connect-AB-Only receives the highest rewards. Connect-AB receives a slightly lower reward than Connect-AB-Only. Considering that Connect-AX or Connect-BX receive small rewards, we assume that a small portion of Connect-AB states are the states where

State Labels	Average Reward	Relative Ratio
No-Event	0.7	0.040
Grab-A	14.21	0.803
Grab-B	10.7	0.604
Connect-AB	17.3	0.977
Connect-AB-Only	**17.7**	**1.0**
Connect-AX	0.51	0.029
Connect-BX	1.8	0.101

Table 4.1: Average COIL-SF reward given to the 7 types of states on the **Magnetic Blocks** environment. COIL-SF gives the highest reward to Connect-AB-Only.



(a) smaller size (b) smaller size (c) more objects, smaller size (d) more objects

Figure 4.3: Progress of the success rate when fine-tuning from a COIL-SF agent *pre-trained on the* 4 objects (object size 100%) setting in Magnetic Blocks environment. Runs are averaged over 3 random seeds. See Section 4.4.6.

objects other than A and B are also connected, and those states have small rewards. Grab-A and Grab-B receive high rewards compared to Connect-AX, Connect-BX, or No-Event. This may be due to Grab-A having an intersection with Connect-AB-Only, which is a set of states where objects A and B are connected and the agent is grabbing the object A.

4.4.6 Generalization to More/Unseen objects

We evaluate the object interaction skills learned by COIL-SF, testing whether they can be applied to environments with more and unseen objects. First, the policy and successor feature networks are pre-trained on Magnetic Blocks with 4 objects for 10 million steps and perform fine-tuning for 1 million additional steps. For each setup, the performance of COIL-SF fine-tuned from pre-trained networks is compared to that of COIL-SF trained from scratch for (10+1) million steps, ensuring a fair comparison. We tested the generalization ability on 4 different setups with varying object sizes and numbers: (a) 4 objects, 33% object size, (b) 4 objects, 66% object size, (c) 6 objects, 100%

object size, and (d) 6 objects, 66% object size.

Unseen objects: (a), (b) To test the generalization ability of COIL-SF on unseen objects, we varied the scale (size) of the objects by 33% or 66%. Figure 4.3 (a-b) show the performance of COIL-SF fine-tuned on pre-trained networks. When tested on the 66% scale, COIL-SF gets a high success rate even without any training. When tested on the 33% scale, the initial performance of COIL-SF is poor, but the performance improves rapidly within 1 million steps of further training while learning COIL-SF from the scratch fails.

More and Unseen objects: (c), (d) To test the generalization ability of COIL-SF on a different number of objects, we conducted experiments with more objects and varying scales (66%, 100%). Figure 4.3 (c-d) show the performance of COIL-SF. Surprisingly COIL-SF fined-tuned on pre-trained networks performs better even in more and unseen objects settings indicating that skills learned from COIL-SF can be used as task-agnostic skills.

Overall, the successful learning of task-agnostic skills with COIL-SF has implications for future research, as these skills could potentially be incorporated into hierarchical reinforcement learning for more complex tasks.

4.5 Conclusion

In this paper, we introduce **COIL** (Counterfactual Object Interaction Learning), a novel approach to learning object-object interaction skills using intrinsic rewards, and the concept of counterfactual dynamics. Our results demonstrate that COIL can effectively learn to interact with objects in challenging continuous, object-centric environments outperforming all the baselines including Sparse-GT, which incorporates task-specific knowledge. We also showed a generalization ability of interaction skills learned by COIL.

Given the complexity and diversity of real-world tasks such as furniture assembly or complex robotics object manipulation, we believe that unsupervised learning of object-object interactions is important, and COIL presents a significant step towards this challenging goal. We note that COIL has some limitations that the method currently relies on a factorized state representation, and do not consider diverse modes of interaction skills. Considering that the real-world tasks contain multiple modes of interaction and complex state representation, combining diverse skill learning (Eysenbach et al., 2019; Sharma et al., 2020b; Park et al., 2022) and object-centric representation learning methods (Locatello et al., 2020) will be an interesting future work.

CHAPTER 5

Conclusion

5.1 Summary of Contributions

The main thesis of this dissertation is that learning representations can facilitate more efficient exploration and goal-conditioned reinforcement learning, including skill discovery. In this dissertation, I focused on two main aspects of representation learning: state abstraction and temporal abstraction for learning skills and goal representations.

Chapter 2 explores an instance of state representation learning to extract task-relevant information, investigating whether discovering controllable elements through the idea of contingency-awareness can lead to efficient exploration. I presented the attentive dynamics model (ADM) that can be trained using the agent's online experience under a very simple self-supervised training objective without external supervision, and can infer the controllable regions as a proxy to task-relevant state information. As empirically demonstrated, this can lead to strong exploration performance (state-of-the-art at the time) on difficult hard-exploration Atari games featuring a sparse reward.

Chapter 3 and Chapter 4 focused on learning goal-conditioned policies or skills as a way of temporal abstraction: how to learn a goal representation that can assign a meaningful behavior, and how to learn a corresponding goal-conditioned policy in an unsupervised fashion, either by learning either a reward function itself or by using some inductive bias to derive intrinsic rewards. Chapter 3 presents a unified view of variational empowerment (MI maximization) methods and classic goal-conditioned RL, enable a principled way to learn the goal representation and the reward function for goal-reaching behavior. The variational GCRL framework also has enabled discovering a missing variant of goal-conditioned RL that identifies controllable dimensions while ignoring nuisance, uncontrollable dimensions in a continuous control setting, which is also closely connected to the idea of contingency-awareness (Chapter 2). Other practical findings and algorithmic contributions, such as the idea of P-HER similar to hindsight experience replay (Andrychowicz et al., 2017) for unsupervised skill discovery, and the use of spectral normalization have also motivated and contributed

to several follow-up works (Park et al., 2022) for discovering useful skills and interesting goal representations towards building hierarchical agents. In Chapter 4, I presented COIL, another instance of learning skills and temporally-abstracted behaviors in a reward-free, unsupervised RL setting. The proposed framework makes use of an idea of counterfactual reasoning, and efficiently leverages dynamics model to learn pairwise object interaction behaviors in entity-centric continuous control environments without any external reward.

Overall, this dissertation contributes to the advancement of deep RL and self-supervised learning by addressing state representation learning, goal representation learning, and skill learning problems which can help build more autonomous systems for real-world problems with less human supervision in the absence of external rewards and human supervision.

5.2 Discussion and Future Directions

Scalability of self-supervised RL methods and applications to real-world problems. Self-supervised methods are useful and motivated by mathematical principles that are generally applicable across different domains, but they are also inherently difficult to scale due to the interdependency between exploration and exploitation as well as the agent's data being non-stationary. Initially, an agent without prior knowledge would exhibit random exploratory behaviors and cover only a small portion of the entire state space. As the agent's policy evolves to learn more non-trivial behaviors, there must be some positive feedback signal (e.g., rewards) to incentivize and reinforce these infrequently achieved behaviors. The effectiveness and accuracy of the feedback usually depend on the quality of the auxiliary components and the representations learned alongside the agent's policy, such as the dynamics model as discussed in this dissertation (e.g., the less error they have, the more accurate the reward for discriminating between good and bad behaviors). This is also often called the *chicken-and-egg* problem: effective exploration requires good representations, and learning effective representations requires sufficient exploratory data (Liu et al., 2021; Tam et al., 2022). Therefore, the model has to generalize and extrapolate very well to the *unseen* data that the agent will encounter in the future. In addition, as reinforcement learning based on trial and error would require a significant amount of data, use of a more efficient exploration algorithm or hybrid approaches combined with hierarchical learning, planning, curriculum learning, etc. will be beneficial for improving the sample efficiency of self-supervised-based representation learning methods and for successfully tackling complex real-world problems such as dexterous robotic manipulation, autonomous driving, and assistant agents for automating household and business tasks, etc.

Balancing between unsupervised and human-supervised RL. While the scope of the works in this dissertation mainly concerns algorithmic perspectives for unsupervised RL (reward-free or

sparse reward settings) towards building fully autonomous and general intelligent systems, it is admitted that such approaches alone will not scale up enough for the most difficult, practical real-world tasks. Therefore, the use of human supervision, domain knowledge, and/or offline data (e.g., expert demonstrations) could also be incorporated to build more performant agents to *some* extent at a manageable level. While it can be prohibitive to collect gathering human supervision (such as high-quality demonstration data and reward engineering), finding a good balance between fully unsupervised and human-supervised reinforcement learning (RL) will be an important problem to develop practically useful and efficient agents. Also, unsupervised skill discovery methods are expected to find a set of good solutions as specified by the optimization objective and inductive biases applied, but there is no guarantee that they will necessarily learn behaviors that are useful to humans for downstream tasks rather than simple state-diversifying behaviors the learning algorithm would easily converge to. Hence, it will be useful to have some moderate level of weak supervision (e.g., few-shot labels) or prior knowledge that can be easily integrated into the model. Many techniques and approaches will have to be discovered to make this process more efficient and more tractable. One category of recently emerging approaches that appear to be promising is human-in-the-loop (Christiano et al., 2017; Wu et al., 2022; Mosqueira-Rey et al., 2023): by allowing humans to provide feedback during training, the agent's performance can be significantly enhanced.

Integrating RL with LLM (Large Language Models). Although the proposed methods can discover some kind of good state and goal representations, those are usually represented in an *abstract* form that is difficult for human to interpret or interface with. For instance, the latent goal space learned by unsupervised skill discovery methods (Choi et al., 2021; Eysenbach et al., 2019; Park et al., 2022) are parametric, have no particular literal meaning on their own without an associated mapping. In addition, the mapping between the latent space and the skill's actual semantics can be *under-specified*: it may converge to different solutions depending on initialization. Even if the skill discovery methods and exploration techniques (with some representation learning) lead to successful learning of powerful and versatile low-level behaviors, it will still be challenging to leverage these abstractions and representations without connecting them to a universal interface that is consistent and agreed upon by humans.

A more intuitive way for humans to interact with an AI agent is through *language*. Large language models (LLM), trained over a large set of corpora of human languages, can predict the next words given some context in the form of sentences. LLMs are known to possess commonsense knowledge about the world around humans and have strong reasoning capabilities (Raffel et al., 2020; OpenAI et al., 2024). As language by itself possesses abstraction and structure that can allow humans to provide AI agents instructions or feedback on which behaviors were good or bad and why, an effective use of LLM in conjunction with reinforcement learning can provide a convenient

interface between humans and AI agents.

One promising and interesting research problem will be how to use language to specify a goal. While there has been some remarkable work to align language and the agent's behavior (Jiang et al., 2019; Prakash et al., 2022; Yu et al., 2023), I plan to explore more advances to combine with the art of representation learning and skill discovery methods to learn language-conditioned skills that would allow discovering more diverse and novel behaviors. However, this is a challenging problem because one would need to build a *paired* dataset of language descriptions and corresponding states of fulfilled goals; there remain several challenges including, but not limited to: determining what would be a good reward function (that is not too sparse) to align the language goal and 'good' states discovered by an exploratory agent, and what would be a principled learning signal (not necessarily reward maximization) to learn them effectively, while requiring less strong supervision.

APPENDIX A

Appendix: Contingency-Aware Exploration in Reinforcement Learning

A.1 Summary of Training Algorithm

The learning objective $\mathcal{L}^{\text{ADM}}$ is from Equation (2.5). The objective $\mathcal{L}^{\text{A2C}}$ of Advantage Actor-Critic (A2C) is as in (Mnih et al., 2016; Dhariwal et al., 2017):

$$\mathcal{L}^{\text{A2C}} = \mathbb{E}_{(s,a,r)\sim\mathcal{E}}\left[\mathcal{L}^{\text{A2C}}_{\text{policy}} + \frac{1}{2}\mathcal{L}^{\text{A2C}}_{\text{value}}\right] \tag{A.1}$$

$$\mathcal{L}^{\text{A2C}}_{\text{policy}} = -\log \pi_\theta(a_t|s_t)(R^n_t - V_\theta(s_t)) - \alpha\mathcal{H}_t(\pi_\theta) \tag{A.2}$$

$$\mathcal{L}^{\text{A2C}}_{\text{value}} = \frac{1}{2}\Big(V_\theta(s_t) - R^n_t\Big)^2 \tag{A.3}$$

$$\mathcal{H}_t(\pi_\theta) = -\textstyle\sum_a \pi_\theta(a|s_t)\log\pi_\theta(a|s_t) \tag{A.4}$$

where $R^n_t = \sum_{i=0}^{n-1}\gamma^i r_{t+i} + \gamma^n V_\theta(s_{t+n})$ is the n-step bootstrapped return and α is a weight for the standard entropy regularization loss term $\mathcal{H}_t(\pi_\theta)$. We omit the subscript as $\theta = \theta_{\text{A2C}}$ when it is clear.

A.2 Architecture and Hyperparameter Details

The architecture details of the attentive dynamics model (ADM), the policy network, and hyperparameters are as follows.

Algorithm A.1: A2C+CoEX

Initialize parameter θ_{ADM} for attentive dynamics model f_{ADM}
Initialize parameter θ_{A2C} for actor-critic network
Initialize parameter θ_c for context embedding projector if applicable (which is not trainable)
Initialize transition buffer $\mathcal{E} \leftarrow \emptyset$
for each iteration **do**
  $\triangleright$ *Collect on-policy transition samples, distributed over K parallel actors*
  for each step t **do**
    $s_t \leftarrow$ Observe state
    $a_t \sim \pi_\theta(a_t|s_t)$
    $s_{t+1}, r_t^{\text{ext}} \leftarrow$ Perform action a_t in the environment
    $\triangleright$ *Compute the contingent region information*
    $\overline{\alpha}_{t+1} \leftarrow$ Compute the attention map of s_{t+1} using f_{ADM}
    $c(s_{t+1}) \leftarrow$ Compute the observation embedding cluster of s_{t+1} (Algorithm A.2)
    $\triangleright$ *Increment state visitation counter based on the representation*
    $\psi(s_{t+1}) \leftarrow (\arg\max_{(i,j)} \overline{\alpha}_{t+1}(i,j),\, c(s_{t+1}),\, \lfloor \sum_{k=0}^{t} r_k^{\text{ext}} \rfloor)$
    $\#(\psi(s_{t+1})) \leftarrow \#(\psi(s_{t+1})) + 1$
    $r_t^+ \leftarrow \frac{1}{\sqrt{\#(\psi(s_{t+1}))}}$
    Store transition $\mathcal{E} \leftarrow \mathcal{E} \cup \left\{(s_t, a_t, s_{t+1}, \beta_1 \text{clip}(r_t^{\text{ext}}, -1, 1) + \beta_2 r_t^+)\right\}$
  end for
  $\triangleright$ *Perform actor-critic using on-policy samples in $\mathcal{E}$*
  $\theta_{\text{A2C}} \leftarrow \theta_{\text{A2C}} - \eta \nabla_{\theta_{\text{A2C}}} \mathcal{L}^{\text{A2C}}$
  $\triangleright$ *Train the attentive dynamics model using on-policy samples in $\mathcal{E}$*
  $\theta_{\text{ADM}} \leftarrow \theta_{\text{ADM}} - \eta \nabla_{\theta_{\text{ADM}}} \mathcal{L}^{\text{ADM}}$
  Clear transition buffer $\mathcal{E} \leftarrow \emptyset$
end for

Hyperparameters		Value	
Policy and Value Network Architecture		Input: 84x84x1	
		- Conv(32-8x8-4)	/ReLU
		- Conv(64-4x4-2)	/ReLU
		- Conv(64-3x3-1)	/ReLU
		- FC(512)	/ReLU
		- FC($\|\mathcal{A}\|$), FC(1)	
ADM Encoder Architecture		Input: 160x160x3	
		- Conv(8-4x4-2)	/LeakyReLU
		- Conv(8-3x3-2)	/LeakyReLU
		- Conv(16-3x3-2)	/LeakyReLU
		- Conv(16-3x3-2)	/LeakyReLU
MLP Architecture for $e_t(i,j)$		FC(1296,256)	/ReLU
		- FC(256,128)	/ReLU
		- FC(128,$\|\mathcal{A}\|$)	
MLP Architecture for $\widetilde{\alpha}_t(i,j)$		FC(1296,64)	/ReLU
		- FC(64,64)	/ReLU
		- FC(64,1)	
λ_{ent} for Loss		0.001	
A2C	Discount Factor γ	0.99	
	Learning Rate (RMSProp)	0.0007	
	Number of Parallel Environments	16	
	Number of Roll-out Steps per Iteration	5	
	Entropy Regularization of Policy (α)	0.01	
PPO	Discount Factor γ	0.99	
	λ for GAE	0.95	
	Learning rate (Adam)	0.00001	
	Number of Parallel Environments	128	
	Rollout Length	128	
	Number of Minibatches	4	
	Number of Optimization Epochs	4	
	Coefficient of Extrinsic and Intrinsic reward	$\beta_1 = 2, \beta_2 = 1$	
	Entropy Regularization of Policy (α)	0.01	

Table A.1: Network architecture and hyperparameters.

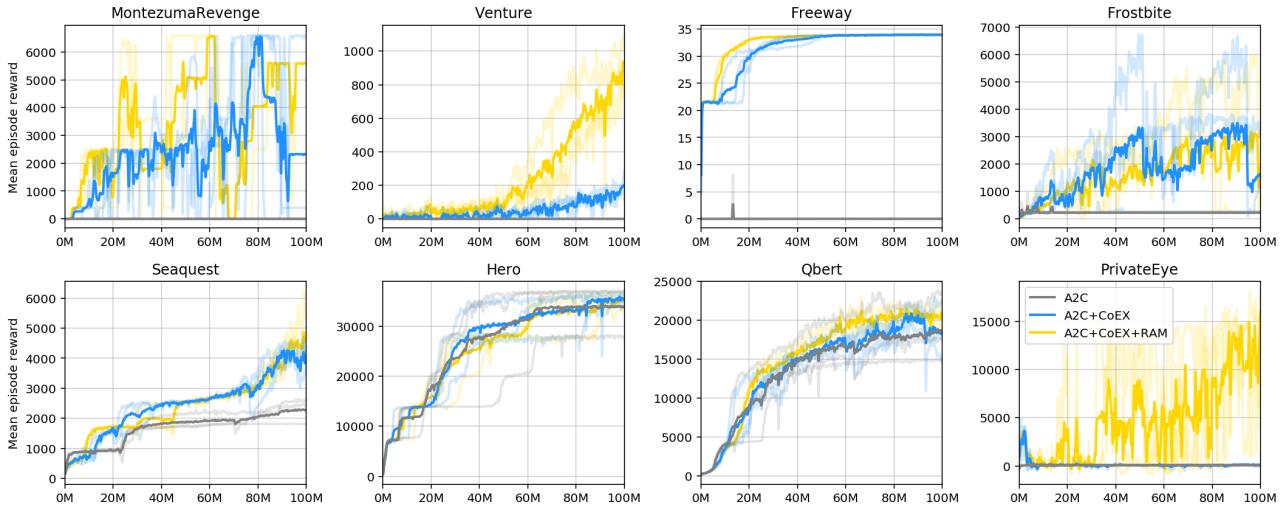


Figure A.1: Learning curves on several Atari games: A2C, A2C+CoEX, and A2C+CoEX+RAM.

Games	β_1 in A2C+CoEX	β_2 in A2C+CoEX	β_1 in A2C	τ for clustering
Freeway	10	10	10	-
Frostbite	10	10	10	-
Hero	1	0.1	1	0.7
Montezuma's Revenge	10	10	10	0.7
PrivateEye	10	10	10	0.55
Qbert	1	0.5	1	-
Seaquest	1	0.5	10	-
Venture	10	10	10	0.7

Table A.2: The list of hyperparameters used for A2C+CoEX in each game. For the four games where there is no change of high-level visual context (Freeway, Frostbite, Qbert and Seaquest), we do not include c in the state representation $\psi(s)$, hence there is no τ. The same values of τ are used in PPO+CoEX.

A.3 Experiment with RAM Information

In order to understand the performance of exploration with perfect representation, we extract the ground-truth location of the agent and the room number from RAM, and then run count-based exploration with the perfect (x, y, c, R). Figure A.1 shows the learning curves of the experiments; we could see A2C+CoEX+RAM acts as an upper bound performance of our proposed method.

A.4 Observation Embedding Clustering

We describe the detail of a method to obtain the observation embedding. Given an observation of shape $(84, 84, 3)$, we flatten the observation and project it to an embedding of dimension 64. We randomly initialize the parameter of the fully-connected layer for projection, and keep the values unchanged during the training to make the embedding stationary.

For the embedding of these observations, we cluster them based on a threshold value τ. The value of τ for each game with change of rooms is listed in Table A.2. If the distance between the current embedding and the center $\mathrm{mean}(c)$ of a cluster c is less than the threshold, we assign this embedding to the cluster with the smallest distance and update its center with the mean value of all embeddings belonging to this cluster. If the distance between the current embedding and the center of any cluster is larger than the threshold, we create a new cluster and this embedding is assigned to this new cluster.

Algorithm A.2: Observation Embedding Clustering

Initialize parameter θ_c for context embedding projector if applicable (which is not trainable)
Initialize threshold τ for clustering
Initialize clusters set $C \leftarrow \emptyset$
for each observation s **do**
  ▷ *Get embedding of the observation from the random projection*
  $v \leftarrow f_{\theta_c}(s)$
  ▷ *Find a cluster to which the current embedding fits, if any*
  Find a cluster $c \in C$ with smallest $\|\mathrm{mean}(c) - v\| \leq \tau$, or NIL if there is no such
  if $c \neq \mathrm{NIL}$ **then**
    $c \leftarrow c \cup v$
  else
    ▷ *if there's no existing cluster that v should be assigned to, create a new one*
    $C \leftarrow C \cup \{v\}$
  end if
end for

In Figure A.2, we also show the samples of observation in each cluster. We could see observations from the same room are assigned to the same cluster and different clusters correspond to different rooms.

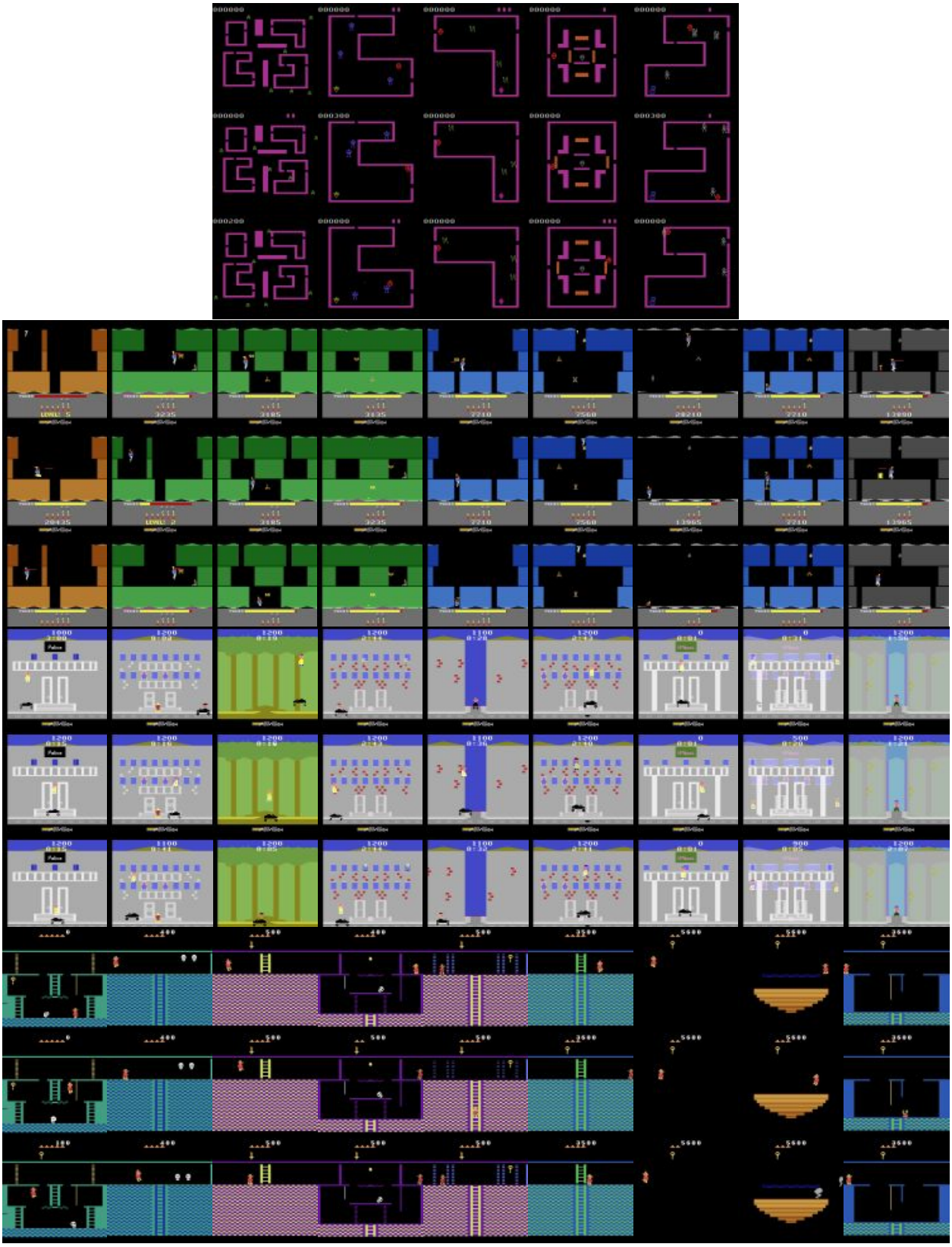


Figure A.2: Sample of clustering results for VENTURE, HERO, PRIVATEEYE, and MONTEZUMA'S REVENGE. Each column is one cluster, and we show 3 random samples assigned into this cluster.

APPENDIX B

Appendix: Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning

B.1 Background: Mutual Information Maximization

We provide a detailed discussion about mutual information objectives as promised in Section 3.3.1.

State-predictive MI: Given a generative model of the form $p^\pi(z, s, s') = p(z)\rho^\pi(s|z)p^\pi(s'|s, z)$ where $p^\pi(s'|s, z) = \int \pi(a|s, z)p(s'|s, a)da$, we define the state-predictive MI as,

$$\mathcal{I}(s'; z \mid s) = \mathcal{H}(s' \mid s) - \mathcal{H}(s' \mid z, s) \tag{B.1}$$

$$= \mathbb{E}_{(z,s,s')\sim p^\pi(z,s,s')} \left[\log p^\pi(s' \mid s, z) - \log p^\pi(s' \mid s)\right] \tag{B.2}$$

This is closer to the classic empowerment formulation as in (Klyubin et al., 2005; Jung et al., 2011). Variational bounds can be derived with respect to actions (Mohamed & Rezende, 2015; Gregor et al., 2017) or to future states (Sharma et al., 2020b). While this objective enables learning state-conditioned skills, we decide to focus on the other variant in this paper.

State-marginal MI: Similarly, given a generative model of the form $p^\pi(z, s) = p(z)\rho^\pi(s|z)$, the MI can be written as,

$$\mathcal{I}(s; z) = \mathcal{H}(z) - \mathcal{H}(z|s) \tag{B.3}$$

$$= \mathbb{E}_{z\sim p(z)}[-\log p(z)] + \mathbb{E}_{z,s\sim p^\pi(z,s)}[\log p(z \mid s)] \tag{B.4}$$

$$= \mathbb{E}_{z\sim p(z),s\sim\pi(z)}[\log p(z \mid s) - \log p(z)] \tag{B.5}$$

$$\geq \mathbb{E}_{z\sim p(z),s\sim\pi(z)}[\log q_\lambda(z \mid s) - \log p(z)], \tag{B.6}$$

where Eq. B.6 is a common variational bound for MI (Barber & Agakov, 2003) with a variational posterior $q_\lambda(z|s)$ approximating the intractable posterior $p^\pi(z|s)$. DIAYN (Eysenbach et al., 2019) optimizes for this state-marginal MI objective in a entropy-regularized RL setting, trained with the SAC algorithm (Haarnoja et al., 2018). We note that an alternative lower bound of $\mathcal{I}(s; z)$ (a "forward" form of MI, *i.e.*, $\mathcal{H}(s) - \mathcal{H}(s|z)$) is also possible (Campos et al., 2020).

B.2 Equivalence between GCRL and Gaussian VGCRL

Full Covariance Gaussian. The Gaussian discriminator (or the variational posterior) $q_\lambda(z|s)$ should take the following form:

$$q_\lambda(z|s) = \mathcal{N}(z; \phi(s), \Sigma(s)) \tag{B.7}$$

$$= \frac{1}{\sqrt{(2\pi)^{|\mathcal{G}|}|\Sigma|}} \exp\left(-\frac{1}{2}(z - \phi(s))^\top \Sigma^{-1}(z - \phi(s))\right) \tag{B.8}$$

Diagonal-Covariance Gaussian. If we assume a diagonal covariance $\Sigma(s) = \mathrm{diag}(\sigma^2(s))$, the discriminator will have the following form:

$$q_\lambda(z|s) = \frac{1}{\sqrt{(2\pi)^{|\mathcal{G}|}}\prod_i \sigma_i} \exp\left(-\sum_i \frac{1}{2\sigma_i^2}(z_i - \mu_i)^2\right), \quad \text{where } \mu_i = [\phi(s)]_i, \sigma_i = [\sigma(s)]_i \tag{B.9}$$

$$\log q_\lambda(z|s) = -|\mathcal{G}| \log(\sqrt{2\pi}) + \sum_i -\log(\sigma_i) + \sum_i \left(-\frac{1}{2\sigma_i^2}(z_i - \mu_i)^2\right) \tag{B.10}$$

As discussed in Section 3.4, the intrinsic reward function for training a goal-conditioned policy for a fixed goal z is given by $r(s) = \log q_\lambda(z|s) - \log p(z)$.

Equivalence to GCRL. It is straightforward to see that for a fixed value of σ_i (say $\sigma_i = 1.0$), Equation B.10 further reduces to

$$\log q_\lambda(z|s) = \text{Const} + \sum_i \left(-\frac{1}{2}(z_i - \mu_i)^2\right) \tag{B.11}$$

up to a constant factor. This can be interpreted as a smooth reward function for reaching a goal $z \in \mathcal{G}$, or the squared distance $\|z - \mu(s)\|_2^2$ between $\mu(s)$ and z in the goal space $\mathcal{G}$. A special case of this is when the goal space is set same as the state space ($\mathcal{G} = \mathcal{S}$) and a natural identity mapping $\mu(s) = s$ is used, where the smooth reward function in standard goal-conditioned RL (GCRL) is

recovered.

B.3 Details of Environments

B.3.1 Windy PointMass

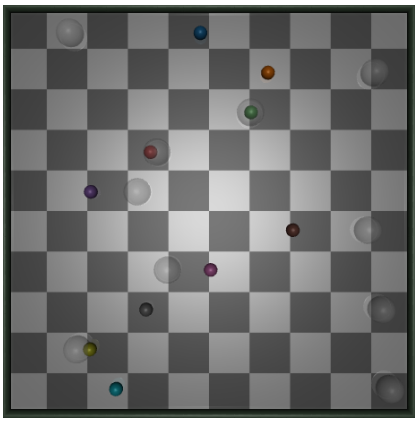


Figure B.1: Windy PointMass (10-dimensional).

The (windy) point mass environment is a N-dimensional continuous control environment. The observation space is $2N$-dimensional, each of which describes the position and the velocity per dimension. Each point mass, one per dimension, can move left and right independently within the arena of range $(-1.5, 1.5)$. The action space is N-dimensional, each of which denoting the amount of velocity acceleration on each dimension. This generalizes common 2D (planar) point mass environments (Brockman et al., 2016; Tassa et al., 2018); indeed, it is exactly equivalent to the 2D point mass environments when $N = 2$. The positions of point masses are initialized randomly at each episode. Figure B.1 shows a target goal location in overlaying transparent spheres (note that in the experiment we assumed the goal $\mathcal{G}$ to be a N-dimensional vector, same as the observation space) with $\mu(s) = s$.

For the windy point mass used in the experiment, we apply a random external force sampled from an uniform distribution $U(-R_i, R_i)$ to the point mass on dimension i, at every time step. The range of external force gets higher as the dimension index i increases; we use a profile of $R_i = 11 \times i$ for $N = 10$ (*i.e.*, $R_0 = 0$ or no force on dimension 0, and $R_9 = 99$ for the last dimension $i = 9$) and $[R_0, R_1] = [0, 40]$ for $N = 2$. With such a large external force, the point mass on dimension $i = 9$ is almost uncontrollable, mostly bouncing around the external perturbation.

B.3.2 Expert State Generation

To generate target states $s^{1:N}$ in the latent goal reaching metric Section 3.6.2, we collected states (observations) randomly sampled from an expert policy's rollout trajectory. Expert policies are SAC agents successfully trained on the task with multiple target velocities rather than the standard task (*i.e.*, only moving forward in HalfCheetah, Ant, Humanoid-v3, etc.). Similar to OpenAI gym's locomotion tasks (Brockman et al., 2016), we use a custom reward function $r_x = \mathrm{HuberLoss}(\text{target } x \text{ velocity} - \text{achieved } y \text{ velocity})$ and a similar one for r_y to let the robot move in some directions with the desired target velocities. The set of target velocities (v_x, v_y) were constructured from the choices of $(-2, -1, -0.5, 0, 0.5, 1, 2)$. We used the SAC implementation from (Guadarrama et al., 2018) with a default hyperparameter setting to train expert policies. We sample 6 random states from each expert policy, yielding a total of $7^2 \times 6 = 294$ (or $7 \times 6 = 42$ for HalfCheetah) target states for each environment. Altogether, this dataset provides a set of states where the agent is posing or moving in diverse direction.

B.4 Implementation Details

For training the goal-conditioned policy, we used Soft Actor-Critic (SAC) (Haarnoja et al., 2018) algorithm with the default hyperparameter setting. To represent the discriminator $q(z|s)$ with a neural network, we simply used a 2-layer MLP with (256, 256) hidden units and ReLU activations. The heads $\mu(s)$ and $\log \sigma(s)$ are obtained through a linear layer on top of the last hidden layer. For Gaussian VGCRLs, we employed an uniform prior $p(z) = [-1, 1]^{|\mathcal{G}|}$ and also applied $\tanh$ bijections to the variational posterior distribution $q_\lambda(z|s)$ to make the domain of z fit $[-1, 1]^{|\mathcal{G}|}$. We also clipped the output of $\log \sigma(s)$ with the clip range $[\log(0.3), \log(10.0)]$ for the sake of numerical stability, so that the magnitude of posterior evaluations (and hence the reward) does not get too large.

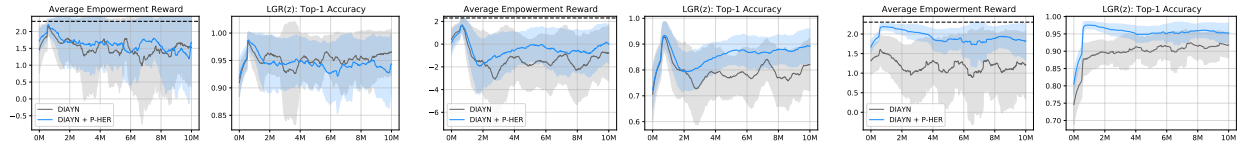
For spectral normalization, we swept hyperparameters σ that control the Lipschitz constant over a range of $[0, 0.5, 0.95, 2.0, 5.0, 7.0]$, and chose a single value $\sigma = 2.0$ that worked best in most cases. The number of mixtures used in Gaussian Mixture Models is $K = 8$. The heads $\mu(s)$, $\log \sigma(s)$, mixture weights $\alpha(s)$ are obtained through a linear layer on top of the last hidden layer.

B.5 More Experimental Results

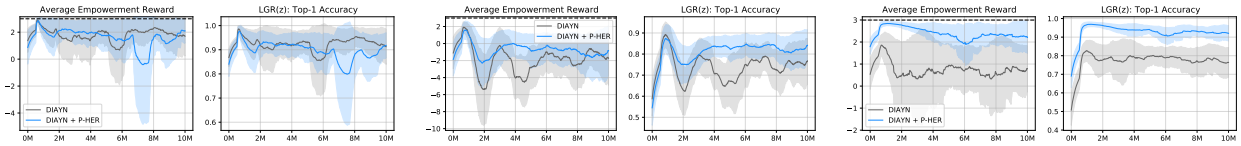
In this section, we present additional results for Section 3.6.3. Table B.1 extends Table 3.4, showing the evaluation metrics for variants of VGCRL where continuous goal spaces of various dimensions are used. Figure B.2 and Figure B.3 show learning curve plots for the VGCRL variants with categorical and Gaussian posterior, respectively.

	$q_\lambda(z\|s)$	P-HER?	SN?	HalfCheetah $\mathcal{F}$	HalfCheetah LGR$_v(s)$	HalfCheetah LGR(z)	Ant $\mathcal{F}$	Ant LGR$_v(s)$	Ant LGR(z)	Humanoid $\mathcal{F}$	Humanoid LGR$_v(s)$	Humanoid LGR(z)
$\|\mathcal{G}\| = 2$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	0.305	0.900	0.166	-0.128	0.398	0.242	-0.047	2.394	0.199
		✔	-	0.339	0.837	0.139	-0.110	0.678	0.306	-0.082	1.505	0.194
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	0.830	1.177	0.063	-4.669	0.478	0.265	0.677	0.393	0.080
		✔	-	0.403	0.720	0.079	-4.575	**0.289**	0.263	2.019	0.910	0.027
		-	✔	2.653	1.017	0.011	2.060	0.453	0.038	2.511	0.225	0.012
		✔	✔	**2.724**	1.074	**0.009**	**2.352**	0.511	**0.023**	**2.549**	**0.199**	**0.012**
	GMM ($K = 8$)	-	-	0.883	**0.707**	0.188	-4.344	0.640	0.360	1.141	1.637	0.072
		✔	-	1.183	2.032	0.181	-3.436	0.432	0.356	2.076	0.993	0.026
$\|\mathcal{G}\| = 5$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	0.932	1.005	0.159	-0.590	1.005	0.382	0.239	1.461	0.202
		✔	-	-0.142	1.273	0.360	0.140	2.449	0.300	0.020	1.452	0.244
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-0.731	1.251	0.172	-18.490	**0.306**	0.427	-3.597	0.538	0.147
		✔	-	-2.161	1.132	0.289	-0.108	2.423	0.303	1.207	0.206	0.074
		-	✔	**5.856**	**0.604**	0.019	2.548	0.925	0.091	4.509	0.460	0.040
		✔	✔	5.803	1.352	**0.017**	**4.349**	0.463	**0.039**	**5.203**	**0.203**	**0.026**
	GMM ($K = 8$)	-	-	-2.646	0.766	0.325	-16.196	0.367	0.486	-3.576	0.231	0.198
		✔	-	-3.091	1.065	0.404	-2.874	2.794	0.325	3.526	0.581	0.043
$\|\mathcal{G}\| = 10$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	-0.145	0.855	0.346	-1.719	0.674	0.508	0.246	0.704	0.237
		✔	-	-0.825	0.866	0.407	-0.276	2.309	0.330	0.125	0.708	0.239
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-3.688	1.381	0.384	-6.709	0.745	0.425	-3.399	0.313	0.221
		✔	-	-3.582	**0.640**	0.388	-0.190	3.989	0.324	-3.618	**0.244**	0.111
		-	✔	3.840	1.175	0.180	0.721	0.974	0.240	**8.134**	1.275	**0.061**
		✔	✔	**4.975**	0.874	**0.162**	**2.467**	0.674	**0.184**	6.349	0.262	0.072
	GMM ($K = 8$)	-	-	-5.137	1.250	0.404	-25.121	**0.307**	0.534	1.885	1.543	0.238
		✔	-	-6.162	0.835	0.399	-3.582	2.396	0.348	3.267	0.422	0.082
$\|\mathcal{G}\| = 20$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	-2.024	0.901	0.438	-3.206	**0.486**	0.498	-0.656	0.622	0.320
		✔	-	-1.848	0.953	0.422	-0.527	3.038	0.331	-0.295	0.461	0.295
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-3.754	**0.481**	0.377	-2.662	0.958	0.353	1.705	2.115	0.274
		✔	-	-4.704	1.648	0.385	-2.813	1.000	0.352	2.149	0.731	0.246
		-	✔	-0.176	1.054	0.318	-1.624	0.716	0.355	**7.176**	1.666	**0.191**
		✔	✔	**0.727**	1.066	**0.294**	**-0.503**	0.496	**0.320**	-0.350	**0.460**	0.340
	GMM ($K = 8$)	-	-	-6.294	1.254	0.394	-11.060	0.805	0.370	1.579	2.280	0.298
		✔	-	-10.647	1.725	0.397	-13.392	1.340	0.377	2.339	1.740	0.284

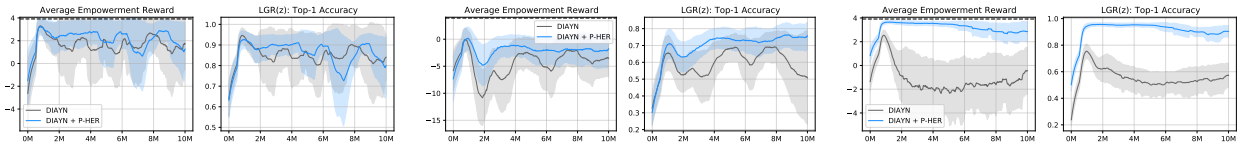
Table B.1: An extended version of Table 3.4. We present a VGCRL-Gaussian variant where the variance is not learned but kept constant (fixed, *e.g.*$\log \sigma$ = 0) and a variant where the variance is learned as a function of state s. VGCRL-GMM is when a Gaussian Mixture Model is used for the discriminator instead of a Gaussian distribution, where means, covariances, and mixture weights are learned through the neural network.



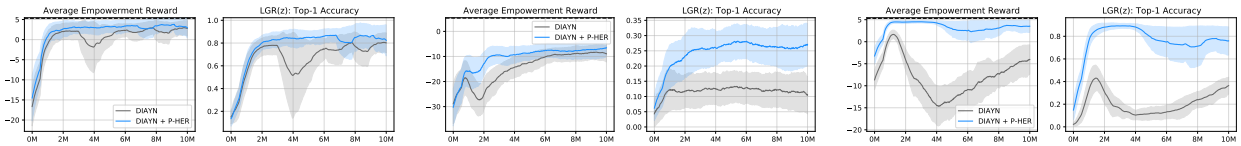
(a) HalfCheetah-v3 ($|\mathcal{G}| = 10$) **(b)** Ant-v3 ($|\mathcal{G}| = 10$) **(c)** Humanoid-v3 ($|\mathcal{G}| = 10$)



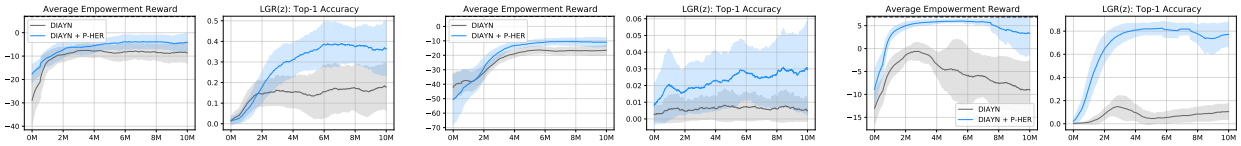
(d) HalfCheetah-v3 ($|\mathcal{G}| = 20$) **(e)** Ant-v3 ($|\mathcal{G}| = 20$) **(f)** Humanoid-v3 ($|\mathcal{G}| = 20$)



(g) HalfCheetah-v3 ($|\mathcal{G}| = 50$) **(h)** Ant-v3 ($|\mathcal{G}| = 50$) **(i)** Humanoid-v3 ($|\mathcal{G}| = 50$)

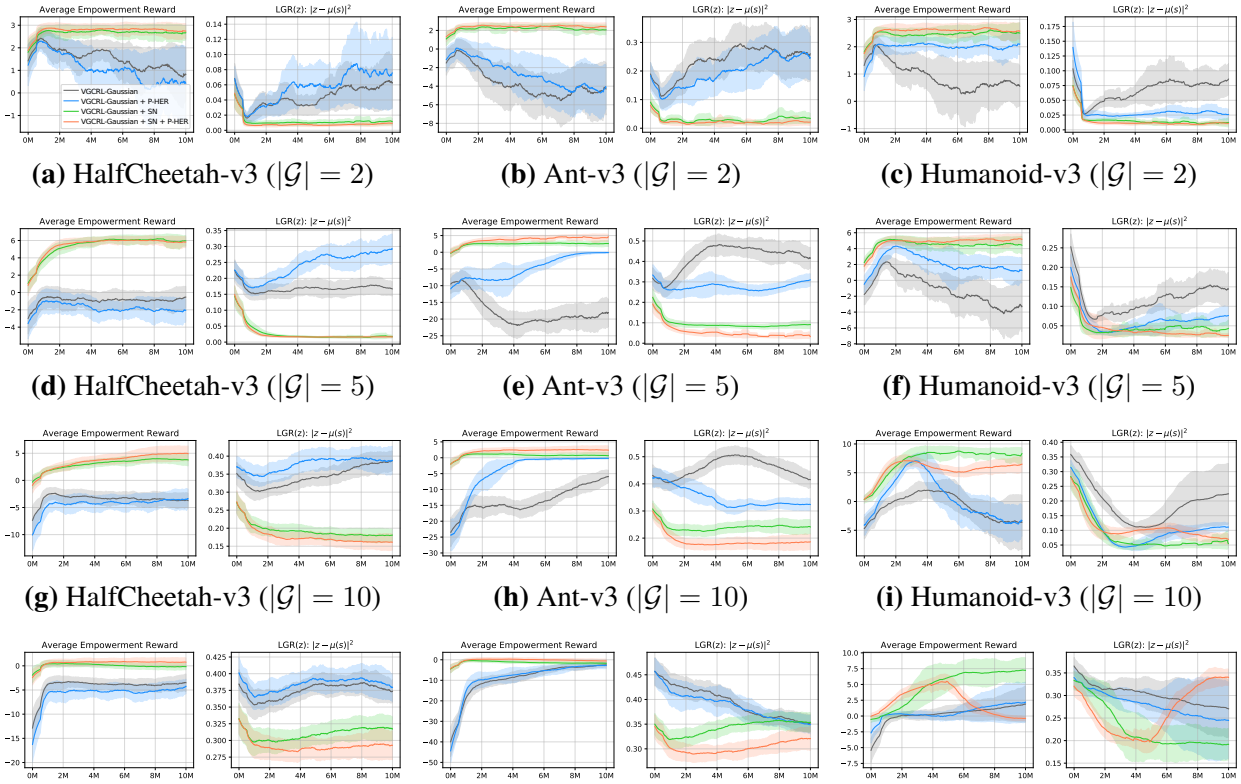


(j) HalfCheetah-v3 ($|\mathcal{G}| = 200$) **(k)** Ant-v3 ($|\mathcal{G}| = 200$) **(l)** Humanoid-v3 ($|\mathcal{G}| = 200$)



(m) HalfCheetah-v3 ($|\mathcal{G}| = 1000$) **(n)** Ant-v3 ($|\mathcal{G}| = 1000$) **(o)** Humanoid-v3 ($|\mathcal{G}| = 1000$)

Figure B.2: Extension to Figure 3.4: Learning curves for VGCRL when discrete, categorical goal spaces are used. The dashed line denotes the maximum possible reward, achieved when the discriminator $q(z|s)$ is perfect at every time step. Overall, we can see P-HER improves the learning process of variational empowerment consistently across different environments and the dimensionality of the goal space.



(a) HalfCheetah-v3 ($|\mathcal{G}| = 2$) **(b)** Ant-v3 ($|\mathcal{G}| = 2$) **(c)** Humanoid-v3 ($|\mathcal{G}| = 2$)

(d) HalfCheetah-v3 ($|\mathcal{G}| = 5$) **(e)** Ant-v3 ($|\mathcal{G}| = 5$) **(f)** Humanoid-v3 ($|\mathcal{G}| = 5$)

(g) HalfCheetah-v3 ($|\mathcal{G}| = 10$) **(h)** Ant-v3 ($|\mathcal{G}| = 10$) **(i)** Humanoid-v3 ($|\mathcal{G}| = 10$)

(j) HalfCheetah-v3 ($|\mathcal{G}| = 20$) **(k)** Ant-v3 ($|\mathcal{G}| = 20$) **(l)** Humanoid-v3 ($|\mathcal{G}| = 20$)

Figure B.3: Extension to Figure 3.4: Learning curves for VGCRL when continuous goal spaces and a family of Gaussian distribution is used for the variational posterior.

APPENDIX C

Appendix: Unsupervised Object Interaction Learning with Counterfactual Dynamics Models

C.1 Details of Implementation and Experiments

C.1.1 Stacking Box

The state of the agent is denoted by its (x, y) coordinates, while each object is represented by $(x, y, z, \textit{held})$, where the binary value of *held* indicates whether the object is in the grasp of the agent or not. The action space is three-dimensional and includes the variables Δx, Δy, and *grab*. The range of values for each variable is from -1 to 1, where Δx and Δy denote the displacement of the agent's movement and *grab* indicates whether to make a grab. If the value of *grab* is 0 or greater, the agent will grab the object; otherwise, it will release it. We use the episode length of 200 by default.

C.1.2 Magnetic Blocks

The state of the agent and each object are 9-dimensional vectors: $(x, y, z, \cos\theta, \sin\theta, v_x, v_y, v_z, \omega)$ where $\cos\theta$ and $\sin\theta$ represent a 2D Euler rotation, v is the velocity, and ω is the angular velocity with respect to the joint. The action space is four-dimensional and includes the variables F_x, F_y, τ, and *grab*. The range of values for each variable is from -1 to 1, where F_x and F_y denote the motor translation torques, τ the rotation torque, and *grab* indicates whether to make a grab. If the value of *grab* is 0 or greater, the agent will grab the object that overlaps with the agent; otherwise, it will release it. We use the episode length of 200 by default.

C.1.3 Implementation details

We discuss more noteworthy implementation details in addition to Section 4.4.2.

Optimization: RL. For the particular RL algorithm choice, we use SAC (Haarnoja et al., 2018) though COIL can be combined with any RL algorithms. The size of the replay buffer is one million (1M) and we start updating the policy after 10 000 steps to fill the replay buffer. We use soft target update with the ratio of τ which stabilizes the optimization of the policy and the model.

Optimization: Forward Model. To update the forward model f_{forward}, we minimize the standard squared L2 error $\left\| \widehat{o^A_{t+1}} - f_{\text{forward}}(o^A_t, \widetilde{o^B_t}, a_t, s^t \setminus \{o^A_t, o^B_t\}) \right\|^2$ with SGD (Adam optimizer with learning rate 3×10^{-4}), using the data uniformly and randomly sampled from the replay buffer (the same one as in SAC). Note that the use of off-policy batch samples prevents a significant overfitting to the current policy, because the model is agnostic to a policy. We also apply L2 regularization with coefficient $\alpha = 0.001$.

Optimization: Successor Features. To update the successor feature $\Psi^\pi_A(s, a)$, we minimize the TD (Temporal-Difference) error, which is a standard way of learning SF (Dayan, 1993; Barreto et al., 2016) using the off-policy data sampled from the replay buffer shared with SAC. We use the n-step TD loss ($n = 5$), which works better than the 1-step TD loss. We did not use any data augmentation techniques.

Hyperparameters. We searched over the hyperparameter range as denoted in Table C.1 for Stacking Box and Table C.2 for Magnetic Blocks. The hyper-parameters that give the highest AUC (area under the curve) in the success rate for each task are chosen as the best hyper-parameters.

C.1.4 Computing resources

To train a single instance of COIL with SAC, we used a single GPU NVIDIA Titan X (consumes around 2GB of VRAM per job), a couple of CPU cores (Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz) and a few GiBs of RAM on a typical x86_64 Linux workstation. Training for 10M environment steps (on the Magnetic Block environment) usually took around 24 hours.

COIL-Forward / Forward Curiosity			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.1	0.1
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01	0.01 / 0.001
Forward model learning rate	1e-5, 3e-5, 1e-4	3e-5	3e-5
Forward model hidden dimensions	[64,64], [256,256]	[256,256]	[256,256]
Reward scale	10, 100, 1000	10	10
COIL-SF / SID			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.1	0.1
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01 / 0.001	0.001
SF model learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
SF model hidden dimensions	[64,64], [256,256]	[256,256]	[256,256]
SF model target update period	1, 5, 10	5	5
SF model discount factor	-	0.8	0.8
τ for SF target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01 / 0.001	0.001
Reward scale	1, 10, 100	10	10
Sparse GT-SAC / RND			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.1	0.1
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01	0.001
Reward scale	10, 100, 1000	10 / 100	10 / 100

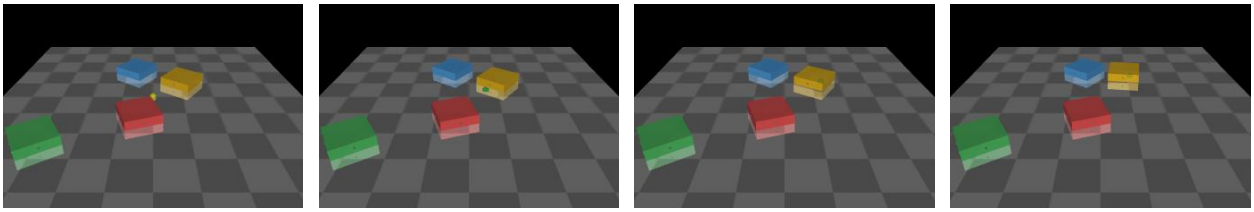
Table C.1: Hyperparameters swept over and the final values used in Stacking Box. n denotes the number of objects.

COIL-Forward / Forward Curiosity			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64,64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.01	0.01
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.3	0.1 / 0.3
Forward model learning rate	1e-5, 3e-5, 1e-4	3e-5	3e-5
Forward model hidden dimensions	[64,64,64,64], [256,256]	[256,256]	[256,256]
Reward scale	10, 100, 1000	10	10
COIL-SF / SID			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64,64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.01	0.3 / 0.01
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01 / 0.1	0.01 / 0.2
SF model learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
SF model hidden dimensions	[64,64,64,64], [256,256]	[64,64,64,64]	[64,64,64,64]
SF model target update period	1, 5, 10	5	5
SF model discount factor	-	0.8	0.8
τ for SF target update	0.001, 0.01, 0.1, 0.2, 0.3	0.01 / 0.1	0.01 / 0.2
Reward scale	1, 10, 100	10	10
Sparse GT-SAC / RND			
Hyperparameters	**Sweep range**	**n=4**	**n=6**
Actor learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Critic learning rate	3e-4, 1e-3, 3e-3	3e-4	3e-4
Actor-Critic network hidden dimensions	[64,64,64,64], [256,256]	[256,256]	[256,256]
Initial temperature	0.001, 0.01, 0.1, 0.3, 1	0.01	0.01
τ for SAC target update	0.001, 0.01, 0.1, 0.2, 0.3	0.1	0.3
Reward scale	10, 100, 1000	10 / 100	10 / 100

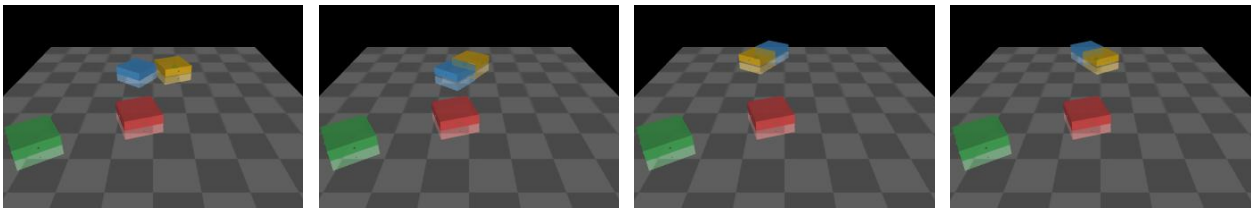
Table C.2: Hyperparameters searched over and the final values in Magnetic Blocks. n denotes the number of objects.

C.2 Qualitative Examples

Figure C.1 shows an qualitative example of the interaction behavior learned by COIL-SF on the Magnetic Blocks environment.



(a) $t = 0, r = 0.36$ **(b)** $t = 2, r = 0.24$ **(c)** $t = 4, r = 2.01$ **(d)** $t = 6, r = 15.30$



(e) $t = 17, r = 13.19$ **(f)** $t = 29, r = \mathbf{22.82}$ **(g)** $t = 134, r = 18.68$ **(h)** $t = 167, r = 17.03$

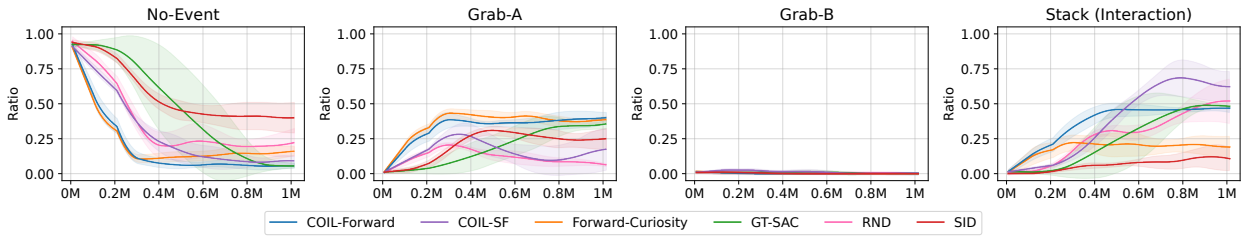
Figure C.1: Snapshots of COIL-SF in Magnetic Blocks. In this episode, A is the yellow object and B is the blue object. (a) Initial state, (b) Agent heads towards A, (c) Agent grabs A, (d) Agent heads towards B while holding A, (e) Agent rotates A to align two objects, (f) Agent connects A and B (**a successful interaction**), (g-h) Agent pushes B to somewhere near the wall. The amount of reward $r_{\text{COIL-SF}}$ the agent receives is shown in the caption of each episode; we can see that the per-step reward is highest when a correct interaction (magnetic connection between the objects in the specified goal) is made.

C.3 Additional Plots

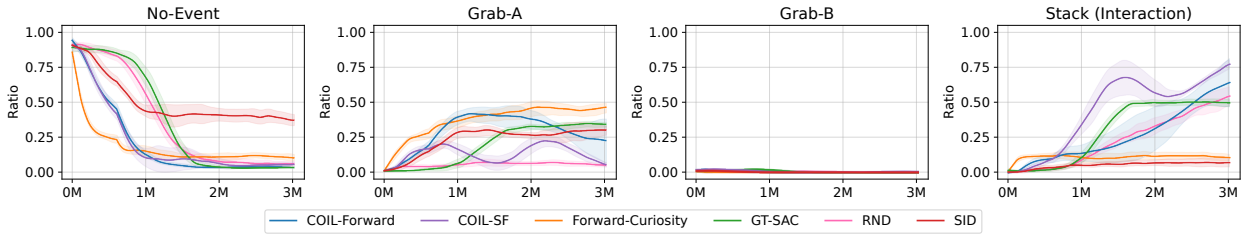
We provide additional plots for further analysis of COIL algorithms. Figure C.2 and Figure C.3 support the quantitative result that COIL learns to make interactions while curiosity-based methods are limited to grabbing the objects.

Algorithm C.1: Learning of COIL-SF (Off-policy learning based on SAC)

Initialize the replay buffer $\mathcal{D}$
while not converged or up to a training budget **do**
  Initialize an episode: Sample an uniform random goal $g \leftarrow (A, B)$, and observe s_0.
  for $t = 1$ .. until the end of episode **do**
    Determine the action $a_t \sim \pi_\theta(a \mid s_t, g)$.
    Observe $s_{t+1} \leftarrow \mathrm{Env}(s_t, a_t)$.
    Add the experience $(s_t, a_t, s_{t+1}; g)$ to the replay buffer $\mathcal{D}$.
    if needs to update (e.g., periodically) **then**
      Sample a batch $\mathcal{D}_B$ from $\mathcal{D}$
      Evaluate the intrinsic reward $r_{\text{COIL-SF}}$ for $\mathcal{D}_B$, using the current estimate of ψ_A^π.
      Run an update step of the policy π_θ with $\mathcal{D}_B$ and the reward $r_{\text{COIL-SF}}$ using SAC.
      Run an update step of the SF ψ_A^π with $\mathcal{D}_B$, minimize the TD loss.
    end if
  end for
end while
return solution

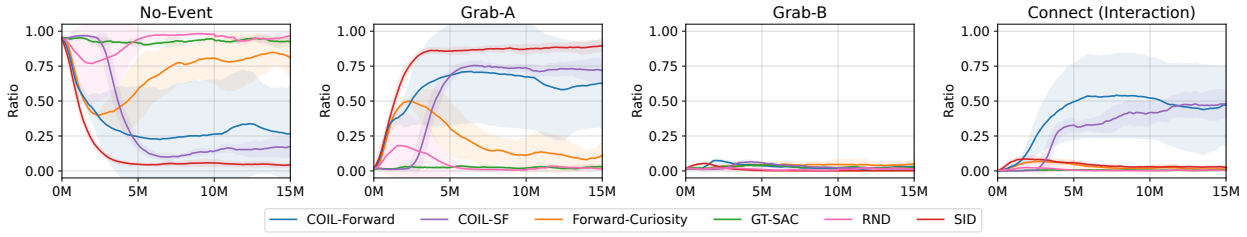


(a) Stacking Box: 4 objects.

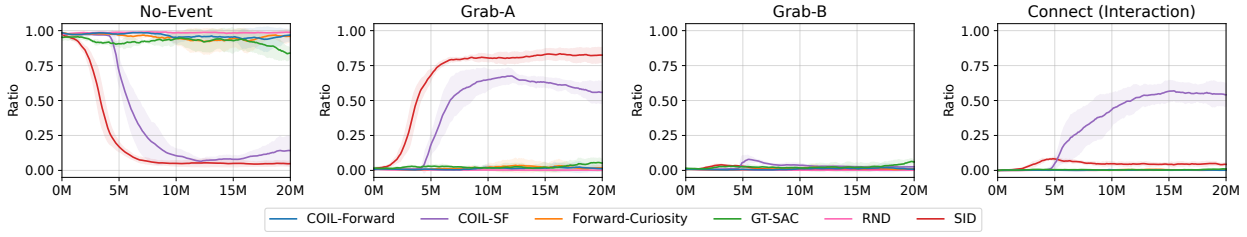


(b) Stacking Box: 6 objects.

Figure C.2: The ratios of the states visited during the episodes for each label in Stacking Box. Runs are averaged over 3 random seeds. (1) No events: States without any grabbing or stacking event, (2) Grab-A: States in which the agent grabs A, (3) Grab-B: States in which the agent grabs B, (4) Stack: States in which A is stacked on B.

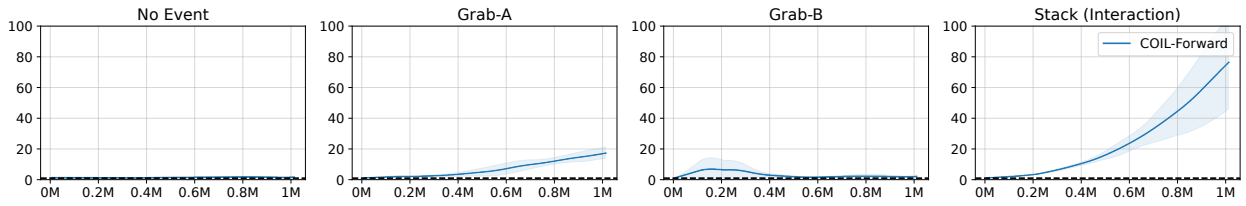


(a) Magnetic Blocks: 4 objects.

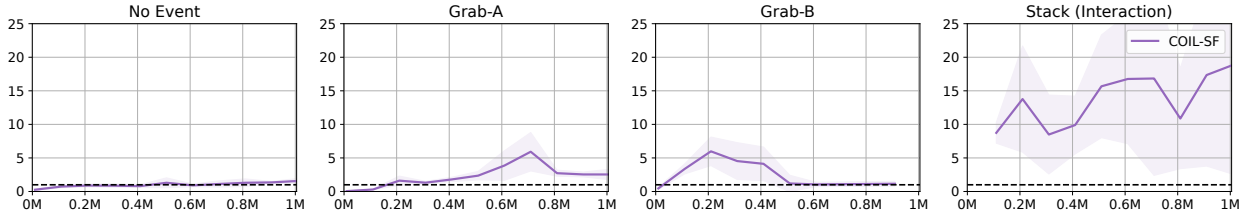


(b) Magnetic Blocks: 6 objects.

Figure C.3: The ratios of the states visited during the episodes for each label in Magnetic Blocks. Runs are averaged over 5 random seeds. (1) No events: States without any grabbing or connecting event, (2) Grab-A: States in which the agent grabs A, (3) Grab-B: States in which the agent grabs B, (4) Connect: States in which A and B are connected.

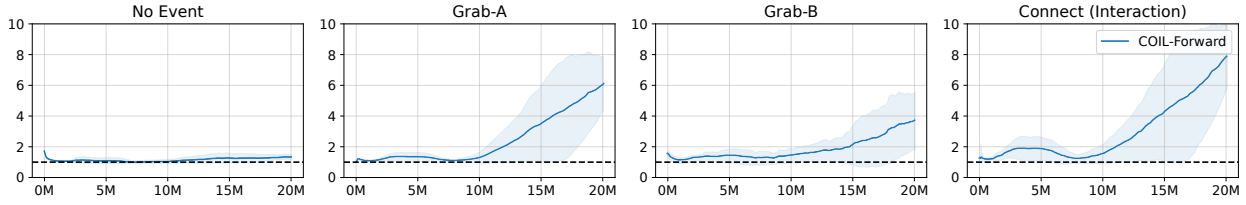


(a) COIL-Forward, Stacking Box: 4 objects.

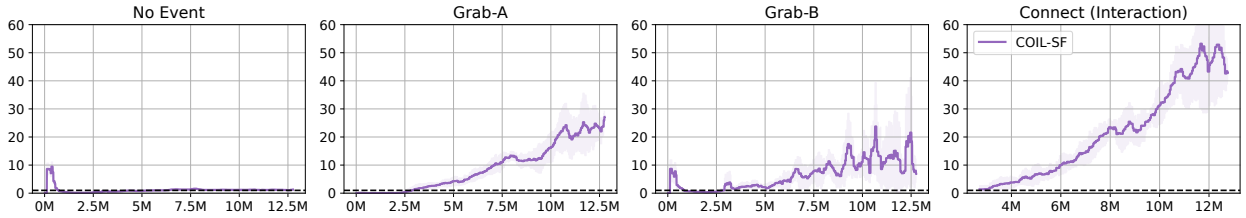


(b) COIL-SF, Stacking Box: 4 objects.

Figure C.4: The ratio of counterfactual prediction error to epistemic uncertainty of dynamics models for each label in COIL, Stacking Box. Runs are averaged over 3 random seeds. (1) No events: States without any grabbing or stacking event, (2) Grab-A: States in which the agent grabs A, (3) Grab-B: States in which the agent grabs B, (4) Stack: States in which A is stacked on B. Higher ratio means that a higher reward r_{COIL} will be given.



(a) COIL-Forward, Magnetic Blocks: 4 objects.



(b) COIL-SF, Magnetic Blocks: 4 objects.

Figure C.5: The ratio of counterfactual prediction error to epistemic uncertainty of dynamics models for each label in COIL, Magnetic Blocks. Runs are averaged over 5 random seeds. (1) No events: States without any grabbing or connecting event, (2) Grab-A: States in which the agent grabs A, (3) Grab-B: States in which the agent grabs B, (4) Connect: States in which A and B are connected. Higher ratio means that a higher reward r_{COIL} will be given.

BIBLIOGRAPHY

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

David Abel. *A Theory of Abstraction in Reinforcement Learning*. PhD thesis, 2022.

David Abel, D Ellis Hershkowitz, and Michael L Littman. Near Optimal Behavior via Approximate State Abstraction. In *ICML*, 2016.

Joshua Achiam and Shankar Sastry. Surprise-Based Intrinsic Motivation for Deep Reinforcement Learning. *arXiv preprint arXiv:1703.01732*, 2017.

Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational Option Discovery Algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In *NeurIPS*, 2016.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay. In *NeurIPS*, 2017.

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Frank Baeyens, Paul Eelen, and Omer van den Bergh. Contingency awareness in evaluative conditioning: A case for unaware affective-evaluative learning. *Cognition and emotion*, 4(1):3–18, 1990.

Akhil Bagaria and George Konidaris. Option discovery using deep skill chaining. In *ICLR*, 2020.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*, 2015.

Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, Greg Wayne, Hubert Soyer, Fabio Viola, Brian Zhang, Ross Goroshin, Neil Rabinowitz, Razvan Pascanu, Charlie Beattie, Stig Petersen, Amir Sadik, Stephen Gaffney, Helen King, Koray Kavukcuoglu, Demis

Hassabis, Raia Hadsell, and Dharshan Kumaran. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

David Barber and Felix V Agakov. The IM algorithm: a variational approach to information maximization. In *NeurIPS*, 2003.

Andre Barreto, Will Dabney, Remi Munos, Jonathan J Hunt, Tom Schaul, David Silver, and Hado van Hasselt. Successor Features for Transfer in Reinforcement Learning. In *NeurIPS*, pp. 1–24, November 2017.

André Barreto, Will Dabney, R. Munos, Jonathan J. Hunt, T. Schaul, David Silver, and H. V. Hasselt. Successor Features for Transfer in Reinforcement Learning. 2016.

Andrew G. Barto. Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*, pp. 17–47. Springer, 2013.

Marc G. Bellemare, Joel Veness, and Michael Bowling. Investigating Contingency Awareness Using Atari 2600 Games. In *AAAI*, 2012.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research 47*, 2013.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying Count-Based Exploration and Intrinsic Motivation. In *NeurIPS*, 2016.

Emmanuel Bengio, Valentin Thomas, Joelle Pineau, Doina Precup, and Yoshua Bengio. Independently Controllable Features. *arXiv preprint arXiv:1703.07718*, 2017.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

Lars Buesing, T. Weber, Yori Zwols, Sébastien Racanière, A. Guez, J. Lespiau, and N. Heess. Woulda, Coulda, Shoulda: Counterfactually-Guided Policy Search. 2018.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning. In *ICLR*, 2019a.

Yuri Burda, Harrison Edwards, Amos J. Storkey, and Oleg Klimov. Exploration by Random Network Distillation. In *ICLR*, 2019b.

Víctor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giro-i-Nieto, and Jordi Torres. Explore, Discover and Learn: Unsupervised Discovery of State-Covering Skills. *arXiv preprint arXiv:2002.03647*, 2020.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 380–388. ACM, 2002.

Daesol Cho, Jigang Kim, and H. J. Kim. Unsupervised Reinforcement Learning for Transferable Manipulation Skill Discovery. *IEEE Robotics and Automation Letters*, 2022.

Jongwook Choi, Yijie Guo, Marcin Moczulski, Junhyuk Oh, Neal Wu, Mohammad Norouzi, and Honglak Lee. Contingency-Aware Exploration in Reinforcement Learning. In *ICLR*, 2019.

Jongwook Choi, Archit Sharma, Honglak Lee, Sergey Levine, and Shixiang Shane Gu. Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning. In *ICML*, 2021.

Jongwook Choi, Sungtae Lee, Xinyu Wang, Sungryull Sohn, and Honglak Lee. Unsupervised Object Interaction Learning with Counterfactual Dynamics Models. In *AAAI*, 2024.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pp. 4299–4307, 2017.

Kamil Ciosek and Shimon Whiteson. Expected policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Cédric Colas, Pierre Fournier, Olivier Sigaud, Mohamed Chetouani, and Pierre-Yves Oudeyer. Curious: intrinsically motivated modular multi-goal reinforcement learning. *arXiv preprint arXiv:1810.06284*, 2018.

P. Dayan. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation*, 1993.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. OpenAI Baselines. https://github.com/openai/baselines, 2017.

Nat Dilokthanakul, Christos Kaplanis, Nick Pawlowski, and Murray Shanahan. Feature Control as Intrinsic Motivation for Hierarchical Reinforcement Learning. *arXiv preprint arXiv:1705.06769*, 2017.

Carlos Diuk, A. Cohen, and M. Littman. An object-oriented representation for efficient reinforcement learning. *ICML*, 2008.

Hugh Durrant-Whyte and Tim Bailey. Simultaneous Localization and Mapping: Part I. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is All You Need: Learning Skills without a Reward Function. In *ICLR*, 2019.

Carlos Florensa, Yan Duan, and Pieter Abbeel. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. *arXiv preprint arXiv:1512.08562*, 2015.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

Jasmina Gajcin and Ivana Dusparic. Counterfactual explanations for reinforcement learning. *arXiv preprint arXiv:2210.11846*, 2022.

Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. *CoRL*, 2019.

Seyed Kamyar Seyed Ghasemipour, Daniel Freeman, Byron David, Shixiang Shane Gu, Satoshi Kataoka, and Igor Mordatch. Blocks Assemble! Learning to Assemble with Large-Scale Structured Reinforcement Learning. *arXiv preprint arXiv:2203.13733*, 2022.

Fausto Giunchiglia, Adolfo Villafiorita, and Toby Walsh. Theories of abstraction. *AI Communications*, 10(3,4):167–176, December 1997.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational Intrinsic Control. In *ICLR Workshop*, 2017.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *ICRA*, pp. 3389–3396. IEEE, 2017a.

Shixiang Shane Gu, Timothy Lillicrap, Richard E Turner, Zoubin Ghahramani, Bernhard Schölkopf, and Sergey Levine. Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning. In *NeurIPS*, pp. 3846–3855, 2017b.

Shixiang Shane Gu, Manfred Diaz, Daniel Freeman, Hiroki Furuta, Seyed Kamyar Seyed Ghasemipour, Anton Raichuk, Byron David, Erik Frey, Erwin Coumans, and Olivier Bachem. Braxlines: Fast and Interactive Toolkit for RL-driven Behavior Engineering beyond Reward Maximization. *arXiv preprint arXiv:2110.04686*, 2021.

Sergio Guadarrama, Anoop Korattikara, Oscar Ramirez, Pablo Castro, Ethan Holly, Sam Fishman, Ke Wang, Ekaterina Gonina, Neal Wu, Efi Kokiopoulou, Luciano Sbaiz, Jamie Smith, Gábor Bartók, Jesse Berent, Chris Harris, Vincent Vanhoucke, and Eugene Brevdo. TF-Agents: A library for reinforcement learning in tensorflow. 2018. URL https://github.com/tensorflow/agents.

Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A Survey on Self-supervised Learning: Algorithms, Applications, and Future Trends, 2023.

Yijie Guo, Jongwook Choi, Marcin Moczulski, Samy Bengio, Mohammad Norouzi, and Honglak Lee. Efficient exploration with self-imitation learning via trajectory-conditioned policy. In *NeurIPS*, 2020.

Abhishek Gupta, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Unsupervised meta-learning for reinforcement learning. *arXiv preprint arXiv:1806.04640*, 2018.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan. Inverse reward design. In *NeurIPS*, pp. 6765–6774, 2017.

Steven Hansen, Will Dabney, Andre Barreto, Tom Van de Wiele, David Warde-Farley, and Volodymyr Mnih. Fast Task Inference with Variational Intrinsic Successor Features. In *ICLR*, 2020.

Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an embedding space for transferable robot skills. 2018.

Elad Hazan, Sham M Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. *arXiv preprint arXiv:1812.02690*, 2018.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization, 2019.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *NeurIPS*, 2016.

Christopher Hoang, Sungryull Sohn, Jongwook Choi, Wilka Carvalho, and Honglak Lee. Successor Feature Landmarks for Long-Horizon Goal-Conditioned Reinforcement Learning. In *NeurIPS*, volume 34, 2021.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: Variational Information Maximizing Exploration. In *NeurIPS*, 2016.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement Learning with Unsupervised Auxiliary Tasks. In *ICLR*, 2017.

Yiding Jiang, Shixiang Gu, Kevin Murphy, and Chelsea Finn. Language as an Abstraction for Hierarchical Deep Reinforcement Learning, 2019.

Tobias Jung, Daniel Polani, and Peter Stone. Empowerment for continuous agent—environment systems. *Adaptive Behavior*, 19(1):16–39, 2011.

Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, 1993a.

Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, pp. 1094–1099. Citeseer, 1993b.

Sham M Kakade. A natural policy gradient. In *NeurIPS*, pp. 1531–1538, 2002.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Maximilian Karl, Maximilian Soelch, Philip Becker-Ehmck, Djalel Benbouzid, Patrick van der Smagt, and Justin Bayer. Unsupervised Real-Time Control through Variational Empowerment, 2017.

Ramtin Keramati, Jay Whang, Patrick Cho, and Emma Brunskill. Fast Exploration with Simplified Models and Approximately Optimistic Planning in Model Based Reinforcement Learning. *arXiv preprint arXiv:1806.00175*, 2018.

Thomas Kipf, Elise van der Pol, and M. Welling. Contrastive Learning of Structured World Models. *ICLR*, 2019.

Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. All else being equal be empowered. In *European Conference on Artificial Life*, pp. 744–753. Springer, 2005.

George Dimitri Konidaris and Andrew G. Barto. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, 2007.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pp. 1097–1105, 2012.

Benjamin Kuipers. The Spatial Semantic Hierarchy. *AIJ*, 119:191–233, 2000.

Brian Kulis and Michael I . Jordan. Revisiting k-means: New Algorithms via Bayesian Nonparametrics. In *ICML*, 2012.

Alex X. Lee, Coline Devin, Yuxiang Zhou, T. Lampe, Konstantinos Bousmalis, J. T. Springenberg, Arunkumar Byravan, A. Abdolmaleki, Nimrod Gileadi, D. Khosid, C. Fantacci, José Enrique Chen, A. Raju, Rae Jeong, Michael Neunert, Antoine Laurens, Stefano Saliceti, Federico Casarini, Martin A. Riedmiller, R. Hadsell, and F. Nori. Beyond Pick-and-Place: Tackling Robotic Stacking of Diverse Shapes. *ArXiv*, 2021.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019a.

Youngwoon Lee, E. Hu, Zhengyu Yang, A. Yin, and Joseph J. Lim. IKEA Furniture Assembly Environment for Long-Horizon Complex Manipulation Tasks. In *ICRA*, 2019b.

Lihong Li, Thomas J. Walsh, and Michael L. Littman. Towards a unified theory of state abstraction for mdps. In *Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics*, 2006.

Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *NeurIPS*, 2017.

Evan Z. Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Decoupling exploration and exploitation for meta-reinforcement learning without sacrifices. In *ICML*, 2021.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, G. Heigold, Jakob Uszkoreit, A. Dosovitskiy, and Thomas Kipf. Object-Centric Learning with Slot Attention. In *NeurIPS*, 2020.

M. Lutter, Leonard Hasenclever, Arunkumar Byravan, Gabriel Dulac-Arnold, Piotr Trochim, N. Heess, J. Merel, and Yuval Tassa. Learning Dynamics Models for Model Predictive Agents. *arXiv preprint arXiv:2109.14311*, 2021.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. *arXiv preprint arXiv:1903.01973*, 2019.

Marlos C Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2017.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. Count-based exploration with the successor representation. In *AAAI*, 2020.

Chris J Maddison, Dieterich Lawson, George Tucker, Nicolas Heess, Arnaud Doucet, Andriy Mnih, and Yee Whye Teh. Particle value functions. *arXiv preprint arXiv:1703.05820*, 2017.

Jarryd Martin, Suraj Narayanan Sasikumar, Tom Everitt, and Marcus Hutter. Count-Based Exploration in Feature Space for Reinforcement Learning. In *IJCAI*, 2017.

André F T Martins and Ramón Fernandez Astudillo. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *ICML*, 2016.

Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. *arXiv preprint arXiv:2011.09464*, 2020.

Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to Navigate in Complex Environments. In *ICLR*, 2017.

Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. Learning to Navigate in Cities Without a Map. In *NeurIPS*, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *ICML*, 2016.

Joseph Modayil and Benjamin Kuipers. Autonomous development of a grounded object ontology by a learning robot. In *Proceedings of the national conference on Artificial intelligence*, volume 22, pp. 1095. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

Joseph Modayil and Benjamin Kuipers. The initial development of object knowledge by a learning robot. *Robotics and Autonomous Systems*, 56(11):879–890, 2008.

Thomas M. Moerland, Joost Broekens, and Catholijn M. Jonker. Model-based reinforcement learning: A survey. *Found. Trends Mach. Learn.*, 16:1–118, 2020.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *NeurIPS*, 2015.

May-Britt Moser, David Rowland, and Edvard I. Moser. Place Cells, Grid Cells, and Memory. *Cold Spring Harbor perspectives in medicine*, 5, 2015.

Eduardo Mosqueira-Rey, Elena Hernández-Pereira, David Alonso-Ríos, José Bobes-Bascarán, and Ángel Fernández-Leal. Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review*, 56(4):3005–3054, 2023.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Near-optimal representation learning for hierarchical reinforcement learning. *arXiv preprint arXiv:1810.01257*, 2018.

Ofir Nachum, Haoran Tang, Xingyu Lu, Shixiang Gu, Honglak Lee, and Sergey Levine. Why Does Hierarchy (Sometimes) Work So Well in Reinforcement Learning? *arXiv preprint arXiv:1909.10618*, 2019.

Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *NeurIPS*, 2018.

Andrew Y. Ng, Stuart J. Russell, et al. Algorithms for inverse reinforcement learning. In *ICML*, 2000.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder P Singh. Action-Conditional Video Prediction using Deep Networks in Atari Games. In *NeurIPS*, 2015.

OpenAI et al. GPT-4 Technical Report, 2024.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *NeurIPS*, 2016.

Georg Ostrovski, Marc G. Bellemare, Aaron van den Oord, and Remi Munos. Count-Based Exploration with Neural Density Models. In *ICML*, 2017.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*, 2009.

Seohong Park, Jongwook Choi, Jaekyeom Kim, Honglak Lee, and Gunhee Kim. Lipschitz-constrained Unsupervised Skill Discovery. In *ICLR*, 2022.

Ronald Parr and Stuart Russell. Reinforcement learning with hierarchies of machines. In *NeurIPS*, volume 10, 1997.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven Exploration by Self-supervised Prediction. In *ICML*, 2017.

Deepak Pathak, Dhiraj Gandhi, and A. Gupta. Self-Supervised Exploration via Disagreement. In *ICML*, 2019.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018a.

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y. Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter Space Noise for Exploration. In *ICLR*, 2018b.

Vitchyr Pong, Shixiang Gu, Murtaza Dalal, and Sergey Levine. Temporal difference models: Model-free deep RL for model-based control. In *ICLR*, 2018.

Vitchyr H. Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

Bharat Prakash, Nicholas R Waytowich, Tim Oates, and Tinoosh Mohsenin. Hierarchical agents by combining language generation and semantic goal directed RL. In *Second Workshop on Language and Reinforcement Learning*, 2022.

Doina Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2000.

Doina Precup and Richard S. Sutton. Temporal abstraction in reinforcement learning. In *ICML*, 2000.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Tim G. J. Rudner, Vitchyr H. Pong, Rowan McAllister, Yarin Gal, and Sergey Levine. Outcome-driven reinforcement learning via variational inference. 2021.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment – An Introduction. In *Guided Self-Organization: Inception*. 2014.

Cansu Sancaktar, Sebastian Blaes, and Georg Martius. Curious Exploration via Structured World Models Yields Zero-Shot Object Manipulation. *arXiv preprint arXiv:2206.11403*, 2022.

Yoshihide Sawada. Disentangling Controllable and Uncontrollable Factors of Variation by Interacting with the World. *arXiv preprint arXiv:1804.06955*, 2018.

Tom Schaul, Dan Horgan, Karol Gregor, and David Silver. Universal Value Function Approximators. In *ICML*, 2015.

Jürgen Schmidhuber. Curious model-building control systems. In *International Joint Conference on Neural Networks*, 1991a.

Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

Jürgen Schmidhuber. Adaptive Confidence And Adaptive Curiosity. *Institut fur Informatik, Technische Universitat Munchen, Arcisstr. 21, 800 Munchen 2*, 1991b.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Maximilian Seitzer, Bernhard Schölkopf, and Georg Martius. Causal influence detection for improving efficiency in reinforcement learning. In *NeurIPS*, 2021.

Archit Sharma, Michael Ahn, Sergey Levine, Vikash Kumar, Karol Hausman, and Shixiang Gu. Emergent Real-World Robotic Skills via Unsupervised Off-Policy Reinforcement Learning. In *Robotics: Science and Systems (RSS)*, 2020a.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-Aware Unsupervised Discovery of Skills. In *ICLR*, 2020b.

Mohit Sharma and Oliver Kroemer. Relational Learning for Skill Preconditions. 2020.

Evan Shelhamer, Parsa Mahmoudieh, Max Argus, and Trevor Darrell. Loss is its own Reward: Self-Supervision for Reinforcement Learning. *arXiv preprint arXiv:1612.07307*, 2017.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. 2014.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

Satinder Singh, T. Jaakkola, and Michael I. Jordan. Reinforcement learning with soft state aggregation. In *NeurIPS*, 1994.

Satinder Singh, Nuttapong Chentanez, and Andrew G. Barto. Intrinsically Motivated Reinforcement Learning. In *NeurIPS*, 2004.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8), 2008.

R. Sutton, Joseph Modayil, M. Delp, T. Degris, P. Pilarski, Adam White, and Doina Precup. Horde: a scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. *Adaptive Agents and Multi-Agent Systems*, 2011.

Richard S. Sutton, David A McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, 2000.

Allison C. Tam, Neil C. Rabinowitz, Andrew K. Lampinen, Nicholas A. Roy, Stephanie C. Y. Chan, DJ Strouse, Jane X. Wang, Andrea Banino, and Felix Hill. Semantic exploration from language abstractions and pretrained representations. In *NeurIPS*, 2022.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. #Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. In *NeurIPS*, 2017.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. DeepMind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. In *CoRL*, pp. 1439–1456. PMLR, 2020.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal Networks for Hierarchical Reinforcement Learning. In *ICML*, 2017.

Oriol Vinyals, I. Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, M. Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, J. Agapiou, Max Jaderberg, A. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, D. Budden, Yury Sulsky, James Molloy, T. Paine, Caglar Gulcehre, Ziyun Wang, T. Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 2019.

David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised Control Through Non-Parametric Discriminative Rewards. In *ICLR*, 2019.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

John S Watson. The development and generalization of "contingency awareness" in early infancy: Some hypotheses. *Merrill-Palmer Quarterly of Behavior and Development*, 12(2):123–135, 1966.

Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *NeurIPS*, volume 30, 2017.

David P. Wipf and Srikantan S. Nagarajan. A new view of automatic relevance determination. In *NeurIPS*, pp. 1625–1632, 2008.

Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. A survey of human-in-the-loop for machine learning. *Future Generation Computer Systems*, 135:364–381, 2022.

Yifan Wu, George Tucker, and Ofir Nachum. The Laplacian in RL: Learning representations with efficient approximations. *arXiv preprint arXiv:1810.04586*, 2018.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2015.

Zhenjia Xu, Zhijian Liu, Chen Sun, Kevin Murphy, William T Freeman, Joshua B Tenenbaum, and Jiajun Wu. Unsupervised discovery of parts, structure, and dynamics. *arXiv preprint arXiv:1903.05136*, 2019.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa,

and Fei Xia. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

Jesse Zhang, Haonan Yu, and W. Xu. Hierarchical Reinforcement Learning By Discovering Intrinsic Options. In *ICLR*, 2021.

Jingwei Zhang, Niklas Wetzel, Nicolai Dorka, Joschka Boedecker, and Wolfram Burgard. Scheduled intrinsic drive: A hierarchical take on intrinsically motivated exploration. *arXiv preprint arXiv:1903.07400*, 2019.

Rui Zhao, Yang Gao, P. Abbeel, Volker Tresp, and W. Xu. Mutual Information State Intrinsic Control. *ICLR*, 2021.

Zeyu Zheng, Junhyuk Oh, and Satinder Singh. On Learning Intrinsic Rewards for Policy Gradient Methods. In *NeurIPS*, 2018.

Zeyu Zheng, Junhyuk Oh, Matteo Hessel, Zhongwen Xu, Manuel Kroiss, Hado van Hasselt, David Silver, and Satinder Singh. What can learned intrinsic rewards capture? *arXiv preprint arXiv:1912.05500*, 2020.

Brian D. Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.