Université de Montréal

# On Discovering and Learning Structure under Limited Supervision

par

Sai Rajeswar Mudumba

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Informatique

August 26, 2022

**Université de Montréal**

Faculté des arts et des sciences

Cette thèse intitulée

**On Discovering and Learning Structure under Limited Supervision**

présentée par

**Sai Rajeswar Mudumba**

a été évaluée par un jury composé des personnes suivantes :

*Aishwarya Agrawal*

(président-rapporteur)

*Aaron Courville*

(directeur de recherche)

*Sarath Chandar Anbil Parthipan*

(membre du jury)

*AntonioManuel Lopez Peña*

(examinateur externe)

(représentant du doyen de la FESP)

# Résumé

Les formes, les surfaces, les événements et les objets (vivants et non vivants) constituent le monde. L'intelligence des agents naturels, tels que les humains, va au-delà de la simple reconnaissance de formes. Nous excellons à construire des représentations et à distiller des connaissances pour comprendre et déduire la structure du monde. Spécifiquement, le développement de telles capacités de raisonnement peut se produire même avec une supervision limitée. D'autre part, malgré son développement phénoménal, les succès majeurs de l'apprentissage automatique, en particulier des modèles d'apprentissage profond, se situent principalement dans les tâches qui ont accès à de grands ensembles de données annotées. Dans cette thèse, nous proposons de nouvelles solutions pour aider à combler cette lacune en permettant aux modèles d'apprentissage automatique d'apprendre la structure et de permettre un raisonnement efficace en présence de tâches faiblement supervisés.

Le thème récurrent de la thèse tente de s'articuler autour de la question « Comment un système perceptif peut-il apprendre à organiser des informations sensorielles en connaissances utiles sous une supervision limitée ? » Et il aborde les thèmes de la géométrie, de la composition et des associations dans quatre articles distincts avec des applications à la vision par ordinateur (CV) et à l'apprentissage par renforcement (RL).

Notre première contribution —Pix2Shape—présente une approche basée sur l'analyse par synthèse pour la perception. Pix2Shape exploite des modèles génératifs probabilistes pour apprendre des représentations 3D à partir d'images 2D uniques. Le formalisme qui en résulte nous offre une nouvelle façon de distiller l'information d'une scène ainsi qu'une représentation puissantes des images. Nous y parvenons en augmentant l'apprentissage profond non supervisé avec des biais inductifs basés sur la physique pour décomposer la structure causale des images en géométrie, orientation, pose, réflectance et éclairage.

Notre deuxième contribution —MILe— aborde les problèmes d'ambiguïté dans les ensembles de données à label unique tels que ImageNet. Il est souvent inapproprié de décrire une image avec un seul label lorsqu'il est composé de plus d'un objet proéminent. Nous montrons que l'intégration d'idées issues de la littérature *linguistique cognitive* et l'imposition de biais inductifs appropriés aident à distiller de multiples descriptions possibles à l'aide d'ensembles de données aussi faiblement étiquetés.

Ensuite, nous passons au paradigme d'apprentissage par renforcement, et considérons un agent interagissant avec son environnement sans signal de récompense. Notre troisième contribution —HaC— est une approche non supervisée basée sur la curiosité pour apprendre les associations entre les modalités visuelles et tactiles. Cela aide l'agent à explorer l'environnement de manière autonome et à utiliser davantage ses connaissances pour s'adapter aux tâches en aval. La supervision dense des récompenses n'est pas toujours disponible (ou n'est pas facile à concevoir), dans de tels cas, une exploration efficace est utile pour générer un comportement significatif de manière auto-supervisée.

Pour notre contribution finale, nous abordons l'information limitée contenue dans les représentations obtenues par des agents RL non supervisés. Ceci peut avoir un effet néfaste sur la performance des agents lorsque leur perception est basée sur des images de haute dimension. Notre approche a base de modèles combine l'exploration et la planification sans récompense pour affiner efficacement les modèles pré-formés non supervisés, obtenant des résultats comparables à un agent entraîné spécifiquement sur ces tâches. Il s'agit d'une étape vers la création d'agents capables de généraliser rapidement à plusieurs tâches en utilisant uniquement des images comme perception.

**Mots clés.** Apprentissage des représentations, apprentissage non supervisé, modélisation générative, perception de scènes 3D, contrôle intrinsèque, modèles du monde, données faiblement supervisées.

# Abstract

Shapes, surfaces, events, and objects (living and non-living) constitute the world. The intelligence of natural agents, such as humans is beyond pattern recognition. We excel at building representations and distilling knowledge to understand and infer the structure of the world. Critically, the development of such reasoning capabilities can occur even with limited supervision. On the other hand, despite its phenomenal development, the major successes of machine learning, in particular, deep learning models are primarily in tasks that have access to large annotated datasets. In this dissertation, we propose novel solutions to help address this gap by enabling machine learning models to learn the structure and enable effective reasoning in the presence of weakly supervised settings.

The recurring theme of the thesis tries to revolve around the question of "How can a perceptual system learn to organize sensory information into useful knowledge under limited supervision?" And it discusses the themes of geometry, compositions, and associations in four separate articles with applications to computer vision (CV) and reinforcement learning (RL).

Our first contribution —Pix2Shape—presents an analysis-by-synthesis based approach(also referred to as inverse graphics) for perception. Pix2Shape leverages probabilistic generative models to learn 3D-aware representations from single 2D images. The resulting formalism allows us to perform a novel view synthesis of a scene and produce powerful representations of images. We achieve this by augmenting unsupervised learning with physically based inductive biases to decompose a scene structure into geometry, pose, reflectance and lighting.

Our Second contribution —MILe— addresses the ambiguity issues in single-labeled datasets such as ImageNet. It is often inappropriate to describe an image with a single label when it is composed of more than one prominent object. We show that integrating ideas from *Cognitive linguistic* literature and imposing appropriate inductive biases helps in distilling multiple possible descriptions using such weakly labeled datasets.

Next, moving into the RL setting, we consider an agent interacting with its environment without a reward signal. Our third Contribution —HaC— is a curiosity based unsupervised approach to learning associations between visual and tactile modalities. This aids the agent to explore the environment in an analogous self-guided fashion and further use this knowledge to adapt to downstream tasks. In the absence of reward supervision, intrinsic movitivation is useful to generate meaningful behavior in a self-supervised manner.

In our final contribution, we address the representation learning bottleneck in unsupervised RL agents that has detrimental effect on the performance on high-dimensional pixel based inputs. Our model-based approach combines reward-free exploration and planning to efficiently fine-tune unsupervised pre-trained models, achieving comparable results to task-specific baselines. This is a step towards building agents that can generalize quickly on more than a single task using image inputs alone.

**Keywords.** Representation learning, unsupervised learning, generative modeling, 3D scene understanding, intrinsic control, world-models, weakly labeled data.

# Contents

# List of tables

# List of figures

# List of abbreviations

| | |
|---|---|
| CNN | Convolutional Neural Network |
| ERM | Empirical Risk Minimization |
| GAN | Generative Adversarial Network |
| LSTM | Long Short-Term Memory |
| GRU | Gated Recurrent Unit |
| MLE | Maximum Likelihood Estimation |
| MLP | Multi-Layer Perceptron |
| MTL | Multi-task Learning |
| NLP | Natural Language Processing |
| SGD | Stochastic Gradient Descent |
| AE | Autoencoder |
| VAE | Variational Autoencoder |
| WGAN | Wasserstein Generative Adversarial Network |
| WGAN-GP | Wasserstein Generative Adversarial Network with Gradient Penalty |
| RL | reinforcement learning |
| e.g. | *exempli gratia* [for instance] |
| MDP | Markov decision process |
| TD | temporal difference |
| DQN | deep Q-network |
| KL | Kullback-Leibler |
| FB | forward-backward representations |
| ALI | Adversarially Learned Inference |
| IL | Iterated Learning |
| BCE | Binary Cross Entropy |
| OOD | Out of Distribution |

| | |
|---|---|
| PDF | Probability Density Function |
| PDDL | Planning Domain Definition Language |
| PMF | Probability Mass Function |
| RGB | Red-Green-Blue |
| RGB-D | Red-Green-Blue - Depth |
| SGD | Stochastic Gradient Descent |
| DQN | Deep Q-Networks |
| SAC | Soft Actor Critic |
| DDPG | Deep Deterministic Policy Gradient |
| ICM | Intrinsic Curioisty Module |
| RND | Random Network Distillation |
| MBRL | Model-based Reinforcement Learning |
| LBS | Latent Bayesian Surprise |
| MPC | Model Predictive Control |
| URL | Unsupervised Reinforcement Learning |
| URLB | Unsupervised Reinforcement Learning Benchmark |
| SF | Successor Features |
| KD | Knowledge Distillation |
| ResNet | Residual Networks |

# Acknowledgements

At the outset, I would like to thank all of the people who have helped me in my research journey which has been a gratifying ordeal.

First and foremost, I would like to express my gratitude to Prof. Aaron Courville, my Ph.D. advisor. I remember the day when I received an acceptance text from Aaron shortly after the Ph.D. admission interview, it was a dream come true. Over the years I have cherished an extended set of learning experiences with him and cannot be more fortunate to have a great mentor. He had my back when I needed moral support and was patient when I needed time in my Ph.D. Giving my first couple of lectures to more than a hundred students among other endeavors, would not have been possible without Aaron's support. His work ethic, humility, and clarity of thought will have a long-lasting impression on my journey ahead. Thank you for the direction, guidance, and sharing your time, I shall treasure it forever.

I had the fortune to work and collaborate with Prof. Yoshua Bengio and Prof. Chris Pal during the course of my Ph.D. It was inspiring to work alongside them. I also had the fortune of being co-advised by Prof. Derek Nowrouzezahrai (McGill University) for one of my projects. The energy and intellect he brings to our weekly meetings are unmatchable. It was again a great learning experience to work with him. I am eternally grateful to David Vazquez (ElementAI) who offered me invaluable mentorship and support during the course of my research journey over the last few years. It amazes me how every conversation with him ends on a personal note of optimism. I am also lucky to work closely with Volodymyr Mnih(DeepMind) and Catalin Ionescu (DeepMind) briefly during my internship. I would like to express my special thanks to Pedro Pinheiro, Pau Rodriguez, Pietro Mazzaglia, and Alexander Lacoste for the valuable collaborations and time during my research. I am thankful to Prof. Simon Lacoste and Bernhard Thomaszewski for valuable feedback and for being a generous committee during my predoc.

My journey at MILA would not have been fruitful without the company I had. I would like to express my thanks to Ishmael Belghazi for being a brother and research buddy in the first many months of my Ph.D. To Sandeep Subramanian for our collaborations and sharing the rollercoaster times in research. To Anirudh Goyal, Alex Lamb, Rosemary, and Chiheb for the help and team dinners. Sarath, Chinna, and Gaurav for the potlucks and board games.

To Fahim Mannan and Jerome for their persistent help. Chinwei, Florian Golemo for keeping the team vibrant. Cyril Ibrahim, Krishna Murthy, Nitin Surya, Soumye Singhal, and Issam Lardiji for the support, feedback, and collaborations.

I have had the pleasure of working and interacting with the interns of Aaron and Yoshua. Ishaan Gulrajani, Jennifer She, Jacob Leyone, Chitwan Saharia, Vikas Verma, Archit Sharma, and Aravind Srinivas had an impact on my research.

I have also had the privilege of standing on the shoulders of amazing mentors and collaborators over the years: Prof. Vineeth Balasubramanian, Prof. Santanu Chaudhury, and Anupama Ray tricked me into pursuing a Ph.D. during my Masters at IIT Delhi.

It is rightly said that friends are the family you choose. Thanks to my friends from India who have stayed in touch even after I moved to Montreal. And I would be remiss in not acknowledging friends from Montreal who I have interacted with regularly in the last five years: To Kundan and Sherjil for being accommodating and fun roommates.

In the end, I would like to extend my loving gratitude to my parents Vani and Srinivas, my brother Kamal, my wife Deepthi, and our newborn son Aadi for their unconditional support, patience, and for keeping me sane against all odds.

# Chapter 1

# Introduction

*«Imagination is the living power and prime agent of all human perception.»*
–Samuel Taylor Coleridge

Learning to make inferences and decisions using observations and effective computing has been a long standing goal of artificial intelligence (AI). Over the last two decades, there have been significant advances in the field due to the availability of large-scale datasets and computational resources across various areas of study from engineering to fundamental sciences [Hey et al., 2009, Reed and Dongarra, 2015]. Underlying this progress is the re-emergence of neural networks that constitute the core of deep learning. Deep learning is hailed as a truly task agnostic approach to computational modeling, relying on large volumes of carefully curated data and gradient descent to distill arbitrarily complex information into an artificial neural network. Despite advances in the field, there is still a fundamental gap in the potential of current deep models and those necessary for complex reasoning in science and engineering. Most success stories are restricted to deep learning applications where we have access to large annotated datasets such as object recognition [Donahue et al., 2014] or environment simulators with handcrafted goals [Silver et al., 2017, Hafner et al., 2021]. However, this reliance makes the learning approach largely impractical for many scientific domains, where the collection of annotations can be prohibitively expensive. There could be various costs associated with obtaining label information, such as privacy, labor costs, safety issues, and the requirement of domain experts together making the approach restricted.

However, such limitation rarely exists in the learning mechanism of natural agents in the real world. Real world sensory perception of natural agents primarily encompasses a wide range of interacting modalities, including visual, auditory, and tactile stimuli. There are convincing and conclusive arguments suggesting that humans do not depend on extensive amounts of labeled supervision for probabilistic reasoning and discovering meaningful structure over such high-dimensional sensory data [Barlow, 1989, Billman and Knutson, 1996, Connolly and Harris, 1971]. For example, when displayed a video, rather than examining individual pixels in each of the frames, one can learn to represent individual objects by grouping similarly colored pixels in close proximity and reasoning about their underlying dynamics across adjacent frames collectively. Similarly, when presented with some new gadget, we do not need days of interaction and constant supervision before obtaining an intuitive grasp of how it works. Since infancy, we employ unguided exploration to learn useful skills [Lindenberger and Lövdén, 2019, Angulo-kinzler, 2001]. In the process, we learn to represent the world in terms of entities centered around spatio-temporal principles of cohesion, continuity, and contact [Fields, 2013]. These instances are unremarkable endeavors of human learning but are surprisingly challenging for the current generation of artificial intelligence. A crucial aspect natural agents demonstrate in both examples is to be able to learn coherent structure while reasoning over such high-dimensional data. To be able to generalize in high dimensions, given only restricted guidance, we need to augment the learning ecosystem of artificial agents with similar perceptual abilities.

Deep learning-based methods have enabled significant progress in the field of vision [Krizhevsky et al., 2012, Dosovitskiy et al., 2020] and control [Levine et al., 2016a, Haarnoja et al., 2018]. However, the real world is much more complex than the curated datasets and the controlled settings where these methods are studied. Thus, when we attempt to apply current methods to more complex situations or tasks, these inevitably struggle [Tsipras et al., 2020b, Yi et al., 2017, Dulac-Arnold et al., 2019, 2020b]. Conversely, the human vision and control systems are robust across variations, occlusions, and noise, and thus they are able to generalize or to adapt fast to changes and novelties in the environment [Geirhos et al., 2018, Thoroughman and Taylor, 2005]. Consequently, the central question, we are interested in is, *"what are the key elements missing to achieve learning abilities of natural agents, such as learning with limited supervision and no prior knowledge about structures of the world?"*

In this thesis, we explore this question from a twofold perspective: (1) the perception process, through which we identify and understand the structure in the environment, and (2) the action selection process, through which we govern the agent's behavior to achieve specific accomplishments. As we show in this thesis, these processes are intertwined, in that meaningful perception of the environment is essential for selecting actions, and intrinsically motivated behaviors can encourage learning a better perception model of the world. In terms of perception, we assess candidates such as geometry aware representation learning, self-supervision from multi-modal signals, and compositional inductive biases to help address the above question. Other potential missing elements could be self-supervision from temporal signal, causal structure discovery and new inductive biases. In terms of action selection, we study the problem of developing intrinsic desires for the agents that encourage learning better models of the world. An improved cognitive model of the environment can then be used to ease the learning of more directed behaviors, such as tasks specified by the user through goals or rewards.

To motivate the learning paradigm, consider an image from the ImageNet dataset, an extensive database of natural images [Deng et al., 2009]. What could have been the underlying mechanism that created such an image in the first place? One can reason that there are many factors involved during the process of image synthesis. Some of these factors deal with the nuances of imaging technology, such as lighting options, camera type, and pose *etc*; whereas others include extrinsic physical and biological phenomena, such as the shapes, textures, objects, and other beings. Despite being characterized by many such factors and compositions during its creation, an image in the ImageNet dataset ultimately comes with limited information both with respect to its overall scene geometry and exogenous compositions. Such weak supervision can restrict a learning system's ability to reason and make inferences about the environment. And more critically it can lead to systematic errors in the learning signal of a downstream vision task [Beyer et al., 2020]. This thesis aims to demonstrate that, by incorporating appropriate inductive biases and with the help of probabilistic modeling and behavior learning, the potential of artificial agents can be further stretched to acquire structural knowledge.

Besides the aforementioned learning paradigms, the focus of this dissertation will be on using deep learning models for learning representations and behavior for a subset of supervision constrained settings, such as unsupervised and weakly supervised learning. Towards this end, we organize the dissertation around four key themes of contributions with applications to vision and reinforcement learning, which we discuss next.

**This thesis is arranged as follows:** The next section provides a synthetic overview of the research contributions that will be detailed in subsequent chapters. Readers less familiar with deep learning, unsupervised learning, and open challenges in these areas will benefit from first reading the background presented in Chapter 2. Since this thesis builds on the notions of inverse graphics and unsupervised exploration, the background chapter also covers the topics of differentiable rendering and reinforcement Learning. Each of the first three contributions corresponds to a published research paper while the fourth is a published workshop paper under review at a conference. Chapter 11 concludes the thesis, discussing future research directions.

As this is a thesis by article, the review, and discussion of the literature most relevant to each contribution is to be found, in its context, within that contribution's chapter, as in the corresponding article, rather than in a separate dedicated literature review chapter.

## 1.1. Research Contributions

### 1.1.1. Unsupervised 3D Aware Representations form Single Images

Images are comprised of shapes, textures, lighting, etc. There is an inherent statistical association between these structural components hidden behind the observant pixels. When provided with an image of scenery, we can picture how it would look from a different viewpoint. We can do it because, unconsciously, we have learned an implicit model for the 3D structure of the scene, which allows us to easily fit the current observation to our mental model and hallucinate the scene from a novel viewpoint. In computer vision, novel view synthesis given just a single view of a scene is an intrinsically ill-posed problem. Building structured generative models, however, has made the following question approachable:

*Given a single image, can we infer the underlying 3D structure of the scene without leveraging any depth based annotations?*

Generative models offer the promise of understanding data in an unsupervised setting. The formalism is directly useful for extrapolating missing information and making inferences about the world's structure. In *«Pix2shape: Towards unsupervised learning of 3d scenes from images using a view- based representation»* [Rajeswar et al., 2020] (Chapter 4) we present a neural network based analysis-by-synthesis approach, by combining ideas from unsupervised deep learning, probabilistic generative modeling and computer graphics. The proposed approach aims to learn 3D aware representations of images given single observations.

### 1.1.2. Multi-label Representations from Single-labelled Images

A 2D image is a projection of a 3D composition of different objects. However, most curated datasets ubiquitous in computer vision such as ImageNet [Deng et al., 2009] are limited to assigning a single label to an image. This is an inappropriate description of the content when images contain multiple, similarly prominent objects. Cognitively inspired linguistic literature suggests that compositionality emerges through imitation from previous generations in the presence of a learning bottleneck [Kirby, 2001]. Inspired from the same, in *«Multi-label iterated learning for image classification with label ambiguity»* [Rajeswar et al., 2022b] (Chapter 6) we try to address the following question:

*Given a singly-labeled dataset, is it feasible to learn multi-label predictions of images considering the underlying compositionality of the world?*

The proposed method is an automatic and iterative process to compose multi-label annotations from a weakly supervised dataset. It mimics the aforementioned imitation procedure by transmitting labels through multiple generations of a perceptual model. A bottleneck is introduced in the form of limited learning iterations between each generation.

The approach has been effective in finding multiple objects in the images without explicit supervision.

### 1.1.3. Touch based Pre-training in the Absence of Dense Supervision

We live in a 3D world where a majority of our learning occurs through observation or trial and error in an unsupervised manner. On the other hand, most success cases of **Reinforcement Learning** (RL) are realized through building specialist agents that are competent at solving individual tasks using well-defined dense rewards (*e.g.* Atari games). In Computer Vision (CV) and Natural Language Processing (NLP) one can leverage unsupervised learning to build generalist adaptable models using large banks of available data. In RL, however, the key is to generate structured behavior in a self-supervised manner using intrinsic exploration. In *«Haptics-based curiosity for sparse-reward tasks»* [Rajeswar et al., 2021] (Chapter 8) we ask a different question:

*Given an environment with more than a single modality, how do we develop a limited supervision agent in a Reinforcement Learning setting?*

In the proposed framework, we show that an agent can leverage tactile modality to explore the environment in a self-guided fashion. This aids in training an RL agent when the rewards are sparse. The acquired knowledge through exploration can in turn be used to adapt to downstream manipulation based tasks. It is well known that self-supervised learning can be used to extract useful structure from data that is unlabeled, similarly, we use self-guided exploration to help agents attain a functional understanding of the environment for better adaptation.

### 1.1.4. Unsupervised Model-based Pre-training for RL from Images

Here I present a model-based approach for perception in pursuance of Unsupervised RL. Control from high-dimensional raw sensory input, such as image pixels, is an arduous task, especially in a data-efficient regime. Such limitation rarely exists in humans, as they learn a repertoire of motor skills by interacting and observing the world with limited supervision. To address this, unsupervised reinforcement learning proposes to collect data through self-supervised pretraining to accelerate task-specific finetuning. In *«Unsupervised Model-based Pre-training for Data-efficient Reinforcement Learning from Pixels»* [Rajeswar et al., 2022a] (Chapter 10), we ask this question:

*Can unsupervised RL lead to improved generalization capabilities when the input observations are high-dimensional?*

We investigate this question against the backdrop of the Unsupervised Reinforcement Learning Benchmark (URLB), a collection of tasks to be solved in a data-efficient manner. In our work, we advance the field by closing the performance gap in the URLB on image inputs.

This is achieved by training a latent dynamics world model via unsupervised exploration during pretraining. And adopting a hybrid planner to efficiently find the actions in order to solve the downstream task during finetuning.

## 1.2. List of Excluded Contributions

During my PhD, I have had the opportunity to work on a variety of projects. I produced other contributions on topics aimed towards understanding how do we represent knowledge of the world, that are not included in this thesis. Some of the relevant works are outlined below.

- **Unsupervised Representation Learning:**
  (1) Ishmael Belghazi*, **Sai Rajeswar***, Olivier Mastropietro, Negar Rostamzadeh, Jovana Mitrovic, Aaron Courville. *Hierarchical Adversarially Learned Inference.* ICML 2018 Workshop on Theory and Applications of Deep Generative Models. [Rajeswar et al., 2018]
  (2) Amjad Almahairi, **Sai Rajeswar**, Alessandro Sordini and Aaron Courville. *Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data.* ICML 2018. [Almahairi et al., 2018]
  (3) Sandeep Subramanian, **Sai Rajeswar**, Alessandro Sordini, Adam Trischler, Christopher Pal, Aaron Courville. *Towards Text Generation with Adversarially Learned Neural Outlines.* NeurIPS 2019. [Subramanian et al., 2018]
  (4) Ishmael Belghazi, Aristided Baratin, **Sai Rajeswar**, Sherjil Ozair, Yoshua Bengio, Aaron Courville, R Devon Hjelm. *MINE: Mutual Information Neural Estimation.* ICML 2018. [Ishmael et al., 2018]

# Chapter 2

# Background

Machine Learning is a study of computational systems that can learn and adapt from underlying data without explicitly being programmed. Although the terminology remained unchanged since the Samuel Checkers-playing [Samuel, 1959], the landscape of learning has evolved in multi-folds. It involves designing **learning algorithms** to solve complex tasks where explicitly articulated "recipes" are difficult (or effectively impossible, a priori) to construct. Some of these included detecting complex patterns, extracting information, reasoning, decision-making, etc. As any agent considered intelligent ought to be able (and should frequently find it useful) to adapt its behavior in light of observation and experience, the study of machine learning is an essential element in the pursuit of artificial intelligence.

Designing algorithms to solve the aforementioned tasks is extremely difficult using rule-based systems where a set of rules are predefined by humans (via imperative or functional programming). Machine learning algorithms leverage statistical regularities in the data to *learn* from examples. An objective of such a learning procedure is *generalization* which distinguishes them from template matching algorithms or look-up tables. It is desirable for such learnt models to generalize to previously unseen situations and examples.

In this dissertation, we integrate this formalism of learning with ideas from other related areas including computer vision, computer graphics, and reinforcement learning. We begin by describing the kinds of learning found in machine learning, followed by detailing some of the learning settings including *Neural Networks*, *Deep Generative Models*, *Differentiable Rendering* and *Deep Reinforcement Learning* which forms the backbone of this thesis.

## 2.1. Machine learning Problem

In modern machine learning problems, the primary task is to leverage observed data (the training set) to specify a function that estimates (predicts) values of interest over unobserved data (the test set). Let the data $D$ be a sample coming from an unknown probability distribution $P$ over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the target or output space. Let $D = \{(x_1,y_1), ..., (x_N, y_N)\}$ be an i.i.d. sample of $N$ pairs of **inputs** $x_i$ and their corresponding **labels** or **targets** $y_i$ drawn from $P$. A machine learning model is (usually) a deterministic function:

$$f : \mathcal{X} \to \mathcal{Y}$$

that predicts outputs given inputs. The models that we consider in this document are all **parametric**: the prediction of a parametric model $f$ is not only a function of an input $x \in \mathcal{X}$, but also of a parameter vector $\theta \in \Theta$, where $\Theta$ is the **parameter space**. We denote $f_\theta$ as the predictive model corresponding to the parameter $\theta$.

We define a **loss** function $l$ that measures a distance (in the loose sense of the term) between the ground-truth label $y$ and a prediction $\hat{y} = f_\theta(x)$. We want to minimize this distance, or equivalently, minimize the mistakes the model makes. Given a set of parameters $\Theta$ and training data $D_{train}$, the goal of a **learning algorithm** is to find the most suitable parameters $\theta \in \Theta$. The learning algorithm **trains** or **fits** the model to the data so that the model has a low generalization error.

The simplest learning principle is **Empirical Risk Minimization** (ERM). It selects the parameters that minimize the **empirical risk** or **train loss**, which is defined similarly to the test loss:

$$L_{train}(\theta) = \frac{1}{|D_{train}|} \sum_{(x,y)\in D_{train}} l(y, f_\theta(x)) \tag{2.1.1}$$

To solve this optimization problem, we need optimization methods adapted to the loss function and the class of models. This problem is tackled in the specific case of neural networks in Section 2.3. Here, we simply assume that ERM is used with a perfect optimization procedure.

## 2.2. Degrees of supervision and Categorization

Machine learning can be studied and segregated over two different dimensions. The first factor is to be able to learn with or without a **teacher**, *i.e.* either train a model with some kind of supervision, be it labels, rewards and other forms of annotations which is called supervised learning or learning without any form of supervision which is often referred to as unsupervised learning. The other distinguishing factor is the nature of underlying data distributions to learn from (see Figure 2.1). A setting is considered **active** when the model or agent is actively changing the data distributions. The active setting is typically where most research in Reinforcement Learning (RL) falls. In a **passive** learning setting, there is a fixed data distribution to learn from.

(1) **Supervised Learning** belongs to the category of passive learning with a teacher. It deals with the discovery of input-output mappings for some task of interest given correct or approximately correct examples of said mapping. The ERM problem setting discussed in the previous section is commonly applicable to supervised learning.

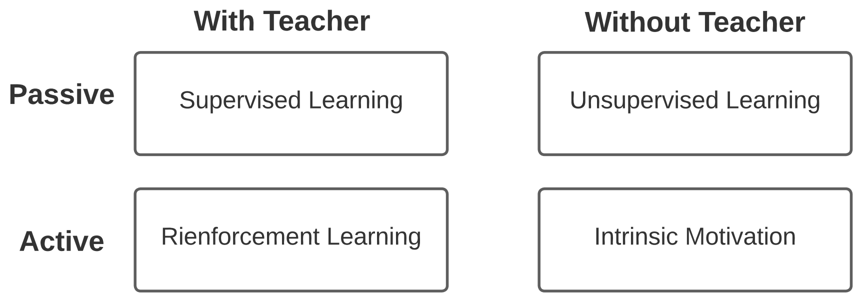


**Fig. 2.1.** Categories of Machine Learning problems

(2) **Unsupervised Learning** comprises of passive learning without a teacher, and generally refers to any procedure that operates on only the "input". This constitutes approaches that attempt to uncover some sort of structure in the distribution of input signals. Input refers to any observed data point $x \in \mathcal{R}^d$ sampled from some fixed data distribution $p_{\mathcal{X}}$. Here, $d$ is the data dimensionality and typically in the order of hundreds to thousands for the data modalities of interest, e.g., the number of pixels in high-resolution images. $p_{\mathcal{X}}$ is unknown analytically; there is only access to a dataset $D_{train}$ of training data points drawn independently from $p_{\mathcal{X}}$. The setting is challenging compared to supervised learning due to the absence of an objective to learn and the lack of annotations. Unsupervised learning is an umbrella term that encompasses a variety of methods:

(a) **Representation Learning** is the problem of learning to produce useful representations of the data. It is still debated what makes a good representation, but one might desire sparse representations or low-dimensional representations for example. *Clustering* is a particular case where we aim to associate each element of the sample space with an element of a discrete set.

(b) **Density Estimation** consists of fitting a probability density function that approximates the density of the unknown data distribution. This approximate density can then be used for example to detect low-probability data points or outliers, which can be interpreted as anomalies.

(c) **Learning generative models**: methods that could model the distribution of interest, either explicitly or implicitly i,e. from which it is easy to generate samples that *look like* they are drawn from the data distribution.

A common underlying learning principle for these methods often times is to search for the parameters $\theta^*$ that best minimize some notion of discrepancy (e.g., a probabilistic divergence) between the data distribution $p_{\mathcal{X}}$ and the model distribution $p_\theta$ where $\theta$ belongs to a set of real-valued parameters $\mathcal{M}$. Since there is only access to samples from $p_{data}$, in practice the discrepency is effectively minimized between $\hat{p}_{\mathcal{X}}$ and $p_\theta$ following the aforementioned principle of empirical risk minimization, where $\hat{p}_{\mathcal{X}}$ denotes the empirical data distribution.

(3) **Reinforcement Learning** falls into the category of learning actively with a teacher. It concerns systems that implement a policy mapping sequences of stimuli (i.e. the state of the "world", as experienced by an autonomous agent) to actions with an objective to maximize the future expected rewards. For instance, an autonomous car (the agent) could be trained to stay driving on the road until it reaches its destination. Each time frame spent on the road is rewarded along with its proximity to the destination, and crashes (and accidents) are penalized. The agent would learn a policy mapping its state (e.g. linear speed, angular speed, linear acceleration, angular acceleration) to actions (e.g. adjusting the steering and brakes) that allows it to safely reach the destination. However, an examination of this paradigm lies beyond the scope of this thesis.

(4) **Intrinsic Motivation** or reward-free exploration deals with learning without a teacher, primarily in the active regime. In many real-world scenarios, rewards extrinsic to the RL agent are extremely sparse or missing altogether, and it is often not possible to construct a shaped reward function. Motivation/curiosity have been used to explain the need to explore the environment and thereby discover novel states and structure of the environment. More generally, this is a way of learning new information and expertise which might be beneficial for pursuing rewards in the future.

In between supervised and unsupervised learning, there are other settings that vary the type and the quantity of information that guides the learning process. On one hand there is supervised learning *i.e.* learning from annotated data. On the other end of the spectrum, **unsupervised learning** problems do not have supervision at all and the data is unlabeled. In between, there exists **weakly-supervised** and **semi-supervised learning** approaches. **Semi-supervised learning** techniques are employed when only a small amount of data is labeled. **Weakly-supervised learning** is an umbrella term covering a variety of studies that attempt to construct predictive models by learning with weak supervision. In this thesis, weakly-supervised learning refers to problems focusing on learning with incomplete and inaccurate supervision.

## 2.3. Deep Learning

Deep learning is concerned with learning multiple levels of representation of data and coming up with higher levels of abstraction [LeCun et al., 2015, Schmidhuber, 2015]. This formalism is inspired by the studies in the field of neuroscience suggesting that the human brain processes signals in multiple stages [McCulloch and Pitts, 1943]. Furthermore, the computational model naturally fits the characteristics of the data itself. Most of the data observed can be explained in a hierarchical form: in language, words make up sentences, which make up the paragraphs of a document. In vision, pixels form edges, and edges form basic shapes, which in turn form more complex shapes in a natural image.

While there are not many assumptions concerned with the learned representation, the concept of distributed representations of data is often central to deep learning [Hinton et al., 1986]. Different from learning local representations, the main idea here is to distribute information about data observations across several dimensions of the feature space. As a simple example, we can think of the binary representation of a set of $N$ integers as a distributed representation ($\log_2 N$ space), while a one-hot vector representation is a local representation ($N$ space). This is particularly important in natural language processing. An intuitive example is representing words in a vocabulary as vectors in $R^d$, also called word embeddings, where each dimension contributes to the word meaning (distributed representation), as opposed to a one-hot vector where dimensions are independent of each other (local representation).

### 2.3.1. Neural Networks

Feedforward Neural networks or multi-layer perceptrons [Rumelhart et al., 1988] are a very popular type of learning function family that form the core of the deep learning formalism. These models consist of one or more layers of arbitrary (usually, differentiable) functions. A single layer is a linear transformation followed by a non-linear transformation. Let $s_i \in \mathbb{N}$ be the dimension of the input of the layer and $s_{i+1} \in \mathbb{N}$ the dimension of the output. The layer $i$ is defined by the function

$$\begin{aligned} f_i : \mathbb{R}^{s_i} &\to \mathbb{R}^{s_{i+1}} \\ x &\mapsto g(W_i x + b_i) \end{aligned}$$

where $W_i \in \mathbb{R}^{s_{i+1} \times s_i}$ is the **weight matrix**, $b_i \in \mathbb{R}^{s_{i+1}}$ is the **bias** and $g$ is the **non-linearity** or the **activation function** such as logistic function, hyperbolic tangent function or rectified linear function $ReLU(x) = max(0, x)$ . The predicted output of this layer for the input $x$ is $f_i(x)$.

A **deep neural network** (DNN) is a neural network composed of several such stacked layers. The function $f$ that computes the predicted output of the whole DNN is the composition of the functions of the layers. In supervised learning, that is

$$\begin{aligned} f : \mathcal{X} &\to \mathcal{Y} \\ x &\mapsto f_N \circ f_{N-1} \circ ... \circ f_1(x) \end{aligned}$$

where $N$ is the number of layers used in the computation, or the **depth** of the network. All the layers except the first $f_1$ and the last $f_N$ are called **hidden layers**. Assume that the parameter set $\theta$ is a union of disjoint mutually exclusive parameter sets $\theta_1, \theta_2, \cdots, \theta_N$ such that $\theta_i$ is the set of parameters of the function $f_i$, and the feedforward neural network specified by $f_\theta$ is $f_{\theta_N} \circ f_{\theta_{N-1}} \circ \cdots \circ f_{\theta_2} \circ f_{\theta_1}$. The parameters $\theta$ tuned by the learning algorithm are the bias vectors and the weight matrices: $\theta = \{\theta_1, \ldots, \theta_N\} = \{W_1, b_1, \ldots, W_N, b_N\}$.

The last layer of a neural network, called the output layer, usually differs from other layers and its form depends on the type of problem we are trying to solve. If we are doing regression, the output layer might look like:

$$y : W_{out} f_N + b_{out}$$

whereas for classification problems we would use

$$a_{out} : W_{out} f_N + b_{out}$$
$$y : softmax(a_{out})$$

where the softmax function is a convenient way to output a normalized distribution over K classes:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}}$$

### 2.3.2. Backpropagation

**Gradient-based** optimization is used to minimize the empirical risk when training neural networks. These iterative methods use the gradient of the objective with regard to the parameters to guide the search. This is possible because neural networks use differentiable building blocks: linear layers and nonlinear activation functions are differentiable, thus their composition is also differentiable.

In order to compute the gradients efficiently, modern deep learning libraries implement the **backpropagation** algorithm. A graph of computation is defined, where nodes are results of computations and a directed edge from node $u$ to $v$ denotes that $v$ requires $u$ to be computed. For every edge $(u,v)$, we say that $u$ is a parent of $v$ or $u \in \pi(v)$. Intermediate results of computations can be expressed as functions of their parents only: $a_i = f_i(a_{\pi(i)})$. Recall that the predictive function $f$ that is computed by a neural network is expressed as a composition of functions $f_\theta = f_N \circ \ldots \circ f_1$ where $\theta$ are the parameters of the network. Given a pair $(x,y)$ of input and target, the loss for the prediction $f(x)$ is $l(y, f_\theta(x))$. The computation of $l$ corresponds to the final node of the computational graph. The empirical risk $L_{train}$ is the average of the loss $l$ over examples from the training set $L_{train}(\theta) = \frac{1}{|D_{train}|} \sum_{(x,y) \in D_{train}} l(y, f_\theta(x))$ so that the gradient $\nabla_\theta L_{train}$ can be computed as

$$\nabla_\theta L_{train} = \frac{1}{|D_{train}|} \sum_{(x,y)\in D_{train}} \frac{\partial l(y, f_\theta(x))}{\partial \theta} \tag{2.3.1}$$

$L_{train}$ is then minimized by using the following gradient-descent update rule to gradually update the parameters $\theta$ of the model $f_\theta$ (i.e., the neural network):

$$\theta^{i+1} \leftarrow \theta^i - \eta \nabla_\theta L_{train}(f_\theta) \tag{2.3.2}$$

So all we need is to be able to efficiently compute, for each (input,target) pair ($x$,$y$), its contribution to the gradient $\frac{\partial l(y,f_\theta(x))}{\partial \theta}$ or in short $\frac{\partial l}{\partial \theta} = \{\frac{\partial l}{\partial W_1}, \frac{\partial l}{\partial b_1}, \ldots, \frac{\partial l}{\partial W_N}, \frac{\partial l}{\partial b_N}\}$ .

During the forward pass, the input is *propagated*, that is, the loss is computed and all the intermediate results $a_i$ are stored. During the backward pass, the gradients are propagated in the other direction, from the loss to the inputs. In other words, this first computes gradient updates for the outermost parameter set $\theta_N$, as $\theta_N \leftarrow \theta_N - \eta\frac{\partial L_{train}(f_\theta)}{\partial \theta_N}$. Subsequently, the inner layers of parameters are gradually expanded, one layer at a time, until the chain rule traces back to the parameter set $\theta_1$. Concretely, the weights are updated along the way using the chain rule as follows:

$$\frac{\partial l}{\partial W_i} = \sum_{j:i\in\pi(j)} \frac{\partial l}{\partial a_j}\frac{\partial a_j}{\partial W_i}$$

and similarly for the bias terms $b_i$. In practice, gradient computation for various numerical operations can be automated, enabling easy implementation while also reducing the scope for execution errors.

## 2.3.3. Convolutional Neural Networks

Convolutional neural networks (CNN) are an important class of algorithms that have been shown to be effective on machine vision problems. Over the years CNNs have been treated and used as the default mechanistic models of the visual system [Hubel and Wiesel, 1962]. Abiding by the general principles of deep learning, CNNs learn a hierarchy of features, which extract higher level of representations as one goes *deeper* in the hierarchy while also taking advantage of the topological structure of input data. This allows the learning of powerful representations of the visual information with efficient (computationally and statistically) architectures. While CNNs are especially popular for 2D image data, they have been extensively applied to other types of data (*e.g.*, text and video).

**CNN Architecture**. The architectural design of a CNN extends a fully-connected MLP while bearing direct parallels to the architecture of the visual system. A typical CNN architecture comprises of one or more instances of a convolutional layer followed by a pooling layer. The convolutional layer can be characterized as follows: given a 2D input $x \in R^{N\times M}$ (*e.g.*, an image with width $N$ and height $M$), the convolutional layer uses a kernel or filter denoted by $W \in R^{K \times K}$, where typically $K \ll N, M$, and computes a feature map $f \in R^{(N-K+1)\times(M-K+1)}$ as follows:

$$f_{i,j} = \phi(W^T x_{i,j}) \tag{2.3.3}$$

where $x_{i,j}$ is an input patch of size $K \times K$ centered at the location $(i,j)$, $f_{i,j}$ is the result of the dot product of the matrices at the location $(i,j)$ of the feature map, and $\phi$ is a nonlinearity or activation function. The feature map is computed by convolving the kernel on the entire input. Typically a convolutional layer generally uses several kernels and is applied to inputs with multiple channels. A spatial decimation operation is applied after a convolutional layer to subsample the resulting feature map. This includes taking the average or maximum value in a given local neighbourhood (average pooling or max pooling, respectively), though more recently simple subsampling has become a popular alternative [Springenberg et al., 2015]. Pooling helps in achieving invariance of the resulting representation with respect to particular changes in the input.

**CNN Properties**. Convolutional architecture benefits from the following properties which makes them a preferable learning model of the visual system:

(1) It allows for parameter sharing across many spatial locations, which results in using much fewer parameters and much sparser connectivity compared to a fully connected layer.

(2) It produces outputs (representations) that are equivariant to input translations. This is especially useful in image representation because it means that a specific input feature (e.g., an edge) can be detected regardless of its location in the input.

(3) Stacking of convolution-nonlinearity-pooling creates *receptive fields* for individual neurons that increase in size deeper in the network and the features of the image that they respond to become more complex.

## 2.4. Maximum likelihood estimation

So far our discussion is tuned towards deterministic models, i.e. functions $f : \mathcal{X} \to \mathcal{Y}$. We now turn to probabilistic models and outline how they are trained to fit the underlying data distribution and how to compute an optimal prediction with regard to a loss function. Note that we saw a similar treatment while discussing unsupervised learning in the previous section. MLE is an innate learning formalism for unsupervised learning, especially for generative modeling.

## 2.4.1. MLE for Supervised Learning

This subsection assumes we are in the supervised learning setting, the unsupervised learning setting is described in the following subsection. These sections are important with respect to our later discussion on Generative modeling.

Neural networks are often used to define probability mass or density functions on the conditional distribution $P_{\mathcal{Y}|\mathcal{X}}$. In statistics, such probabilistic models can be fit to the data using different methods such as the method of moments or (conditional) **maximum likelihood estimation** (MLE). MLE is a fundamental learning objective in machine learning and deep learning. It allows one to learn the parameters that maximize the probability of the data:

$$\begin{aligned}\hat{\theta} &= \underset{\theta\in\Theta}{\operatorname{argmax}} \prod_{(x,y)\in D_{train}} p_\theta(y|x) \\ &= \underset{\theta\in\Theta}{\operatorname{argmin}} \sum_{(x,y)\in D_{train}} -\log p_\theta(y|x)\end{aligned} \tag{2.4.1}$$

Thus, MLE is equivalent to ERM with the **log loss**:

$$l(y, f_\theta(x)) = -\log p_\theta(y|x)$$

## 2.4.2. MLE for Unsupervised Learning

Recall from the earlier introduction to unsupervised learning that $p_\mathcal{X}$ denotes the data distribution, $\hat{p}_\mathcal{X}$ denotes the empirical data distribution which assigns uniform probability mass to all points in $D_{train}$ and zero outside, and $p_\theta$ is the model distribution.

The Maximum Likelihood Estimation (MLE) principle, in this case, is equivalent to minimizing the Kullback-Leibler (KL) divergence between the data distribution and the model distribution. Accordingly, MLE optimizes for the model parameters using the following objective:

$$D_{KL}(p_\mathcal{X}, p_\theta) = \mathbb{E}_{p_\mathcal{X}}[\log p_\mathcal{X}(x)] - \mathbb{E}_{p_\mathcal{X}}[\log p_\theta(x)]$$

The first term on the right-hand side of the equation corresponds to the entropy of the data distribution. Since the quantity does not depend on the model parameters and is constant w.r.t. $\theta$, it can be ignored. Therefore the Maximum Likelihood Estimate objective for the unsupervised learning amounts to minimizing the expected negative log-likelihood assigned by the model to the training dataset:

$$\min_{\theta\in\mathcal{M}} -\mathbb{E}_{p_\mathcal{X}}[\log p_\theta(x)] \tag{2.4.2}$$

Since $p_{\mathcal{X}}$ is not accessible, in practice it is approximated using the empirical data distribution $\hat{p}_{\mathcal{X}}$ similar to Equation 2.1.1.

## 2.5. Generative Models

Generative Modeling is one of the key components of unsupervised learning and plays a major role in analyzing and understanding the data. There are two inference tasks predominantly associated with a generative model:

(1) **Density estimation**: For any given data point, compute the probability assigned by the model $p_\theta(x)$.

(2) **Sampling**: To be able to generate samples from the learnt model distribution $x \sim p_\theta$

Some model categories additionally include latent variables $Z$ along with the observed variables $\mathcal{X}$. For this class of models, we will include an additional problem of inferring the latent variables for a given data point $x$.

The classical generative models were initially confined to simple model families, namely mixtures of Bernoulli and finite Gaussian distributions to model discreet and continuous data respectively using the Expectation-Maximization (EM) principle. Such models do not scale to high-dimensional data modalities such as images, videos, and audio due to the curse of dimensionality. As the number of data dimensions increases, the target data distribution $p_{\mathcal{X}}$ can be highly complex (spanning a huge number of modes). This requires a very large number of mixture components to fit underlying data distributions and consequently, optimization can be largely challenging in this setting. Traditionally, dimensionality reduction approaches (also known as manifold learning techniques) such as variants of Principal Component Analysis (PCA) [F.R.S., 1901], Local Linear Embedding (LLE) [Roweis and Saul, 2000] among others are used to deal with the curse of dimensionality issue.

Recently, afore-mentioned deep neural network based models have proven highly effective for a variety of tasks and modalities involving high-dimensional data such as in image classification [Donahue et al., 2014, Zeiler and Fergus, 2014] natural language processing [Vaswani et al., 2017] and speech recognition [Hinton et al., 2012]. Their learning mechanism and underlying optimization algorithm display inductive biases that can help generalize beyond the training data. In this section, we study how these deep networks can be useful for probabilistic generative modeling of high-dimensional data. For that, we briefly describe and differentiate four major frameworks of generative modeling well-known today: energy-based models, autoregressive models, variational autoencoders, and generative adversarial networks. Normalizing Flows [Rezende and Mohamed, 2015] is another generative model paradigm that has been recently successful, however, it is beyond the scope of this thesis.

### 2.5.1. Energy Based Models

Inspired from statistical physics, an energy-based model can be seen as a parameterized representation of the Boltzmann distribution [LeCun et al., 2006]. That is, the probability density of any data point $x \in \mathcal{R}^d$ is given by the Boltzmann distribution:

$$p_\theta(x) \propto \exp(-E_\theta(x)), \tag{2.5.1}$$

The real-valued function $E_\theta(x)$ denotes the energy of a data point $x$ with parameters $\theta$ and is low for points with high probability under $p_\theta$. EBMs can include latent variables Z and efficiently learned by restricting the connectivity between the latent and observed variables in a framework called Restricted Boltzmann Machines (RBM). Here, we outline the learning and inference for the EBM setting in Equation 2.5.1. An extension of this formulation to more general RBMs that are applicable to high-dimensional data can be found in Hinton [2012], Courville et al. [2014]. Lately, the energy function for an EBM is parameterized by a deep neural network and the learning objective for the model parameters is derived via Maximum Likelihood Estimate (MLE). Substituting the Boltzmann distribution Equation 2.5.1 in Equation 2.4.2, we get:

$$\mathcal{L}_{EBM}(\theta) = \mathbb{E}_{p_\mathcal{X}}[\log E_\theta(x)] - \log \mathcal{Z}_\theta \tag{2.5.2}$$

where $\mathcal{Z}_\theta = \int exp(-E_\theta(x))dx$ is the partition function and is intractable to compute for high-dimensional spaces. The gradients of the loss function are given as:

$$\nabla_\theta \mathcal{L}_{EBM}(\theta) = \mathbb{E}_{p_\mathcal{X}}[\nabla_\theta E_\theta(x)] - \mathbb{E}_{p_\theta}[\nabla_\theta E_\theta(x)] \tag{2.5.3}$$

An intuitive way to think about Equation 2.5.3 is that the first gradient term minimizes the energy at data points sampled from the training dataset (also called positive examples) and the second term increases the energy at samples drawn from the model $p_\theta$ (negative examples). Markov Chain Monte Carlo (MCMC) methods are used in order to draw samples from the model to be able to estimate the expectation in the second term. These approaches perform sampling by running a carefully constructed Markov chain.

### 2.5.2. Autoregressive Models

Nonetheless, one major drawbacks with aforementioned energy-based models is that the sampling and density estimation using these models is intractable since they are unnormalized. Here, we examine a family of self-normalized models called autoregressive models. In an autoregressive model (ARM), the joint distribution over n random variables $x = (x_1, x_2, ..., x_n)$ can be factorized using the chain rule as:

$$p_\theta(x) = \prod_{i=1}^{n} p_\theta(x_i|x_1, x_2, ..., x_{i-1}) = \prod_{i=1}^{n} p_\theta(x_i|x_{<i}). \tag{2.5.4}$$

If the conditionals $_\theta(x_i|x_{<i})$ are expressive enough to fit and represent any unidimensional probability distribution, then the overall model $p_\theta(x)$ can also constitute and represent any arbitrary data distribution over x. The reasoning for the argument is straightforward from the chain rule applied to the data distribution. Practically, this is realized by picking an ordering for the indices (*e.g.*, raster scan for images) and parameterizing the respective conditionals using deep neural networks. The parameters of the resulting models are learned by maximizing the log-likelihood over the training data. Substituting Equation 2.5.4 in Equation 2.4.2, we get:

$$\mathcal{L}_{ARM}(\theta) = \sum_{i=1}^{d} \mathbb{E}_{p_X}[\log p_\theta(x_i|x_{<i})] \tag{2.5.5}$$

Sampling from these models follows ancestral sampling where every dimension is sampled one at a time in order. Autoregressive models are among the state-of-the-art generative models, impressively successful in image, audio, text and video modalities [Van Den Oord et al., 2016, Salimans et al., 2017, van den Oord et al., 2016]

## 2.5.3. Variational Auto Encoder

As hinted in Section 2.2, representation learning is one of the critical aspects of unsupervised learning. Leveraging latent variable generative models is an effective approach towards this goal. Formally, a latent variable generative model consists of a set of latent or hidden random variables $\mathbb{Z} \in R^k$ in addition to the set of observed variables $\mathbb{X} \in R^d$. Latent variables increase the expressivity of the learned density model and can help model complex distributions by uncovering hidden structure in the data. Mathematically, a latent variable model expresses a joint probability distribution $p_\theta$ over the observed data $\mathbb{X}$ and the latent $\mathbb{Z}$. Such models can be appealing because the structure of a learned latent space can often be statistical much simpler than that of observed space for further analysis. We focus on two key realizations of this formalism: variational autoencoders and generative adversarial networks.

Note that the objective here is to maximize the log-likelihood of the observed data, as we did earlier. However, since we additionally have latent variables as part of the model, we need to marginalize out these latent variables in order to evaluate the loglikelihood. Now, let us examine the marginal log-likelihood of a latent variable model for a point $x$:

$$\log p_\theta(x) = \log \int p_\theta(x, z)dz \tag{2.5.6}$$

The integral for one $(x, z)$ exclusive configuration can be computed in a tractable manner using the chain rule as long as we choose a reasonably simple prior $p_\theta(z)$ and conditional distribution $p_\theta(x|z)$. However, evaluating the integral in the above equation is either slow or often impossible ( especially if $z$ is continuous or has many potential values). In such case we need to resort to an approximate estimation. A variational approximation is a mathematical tool for decomposing such log-sum structured likelihood into a tractable expectation over both $x$ and $z$. It leverage a distribution that is easy to sample and evaluate to approximate the posterior distribution $p_\theta(z|x)$. We thereby obtain an evidence lower bound (ELBO) on the marginal likelihood by considering the variational approximation as a parameterized distribution $q_\phi(x|z)$ with parameters $\phi$:

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] = \text{ELBO} \tag{2.5.7}$$

The proof is based on Jensen's inequality (we omit the details of the proof here, see Kingma and Welling [2014] for a detailed derivation). The estimation gap in the equation is geven by the KL divergence between $q_\phi(z|x)$ and $p_\theta(z|x)$ and the bound is tighter when the two distributions match. Therefore, inorder to train a variational autoencoder (VAE) we need to maximize the ELBO w.r.t. both the model parameters $\theta$ and the variational parameters $\phi$ [Kingma and Welling, 2014, Rezende et al., 2014]. It is easy to see that this can be implemented similar to an autoencoder [Bengio et al., 2013]. However, in a VAE, an encoding that maps data points to a latent space corresponds to a mapping that parameterizes the variational posterior distribution $q_\phi(z|x)$. While the decoding that reconstructs the data points corresponds to the mapping that parameterizes the generative model $p_\theta(x,z)$. Furthermore, we can decompose the lower bound Equation 2.5.7 into two terms:

$$\text{ELBO}(\theta,\phi) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - D_{KL}(q_\phi(z|x), p_\theta(z)) \tag{2.5.8}$$

The first term represents the performance of a VAE in reconstructing the input. The second term minimizes the KL divergence between the approximate posterior and the prior, thereby forcing the variational distribution towards the latent prior. And so has reasonably been the focus of many attempts to alleviate posterior collapse. On a more practical note, Kingma and Welling [2014] proposed the use of reparameterization to propagate the gradients from the decoder network to the encoder network (by computing low-variance Monte Carlo gradient estimators of the ELBO). Over the years, there have been many research extensions of VAE that contributed to the empirical success of the latent variable model. Extensions include improved optimization, expressive parameterizations, powerful objectives, etc.

### 2.5.4. Generative Adversarial networks

A generative adversarial network (GAN) is a latent variable model that is different from the likelihood maximization approaches discussed so far either exactly (autoregressive models) or approximately (energy-based models, variational autoencoders). As noted earlier VAE is a bit restricted in that they make simplifying assumptions on the generated and true data distributions in order to make the computation tractable. Adversarial approach, on the other hand, takes the form of estimating the density ratio between the generating distribution $p_\theta$ and the real data distribution $p_{\mathcal{X}}$. Modeling the difference between a generating distribution and the data distribution allows the model to be sensitive to any notable difference between real samples and generated samples, which may be a much easier pretext objective than determining the density of a distribution at a given point. On a practical note, such a learning mechanism is equivalent to learning a classifier between the real data and the model's distribution. And as noted earlier, the methods for successfully training classifiers using deep networks have been widely studied.

More formally, as outlined earlier in Section 2.4, Maximum Likelihood Estimation minimizes the KL divergence between the data and model distributions. A large family of probabilistic divergences and distances can be conveniently expressed as a difference in expectations w.r.t. $p_{\mathcal{X}}$ and $p_\theta$ :

$$\max_{\phi \in \mathcal{F}} \mathbb{E}_{p_\theta} h_\phi(x) - \mathbb{E}_{p_{\mathcal{X}}} h'_\phi(x) \tag{2.5.9}$$

where $\mathcal{F}$ denotes a set of parameters, $h_\phi$ and $h'_\phi$ are appropriate real-valued functions parameterized by $\phi$. A Monte Carlo estimate of Equation 2.5.9 can be derived using only samples from the model. Combining Equation 2.4.2 with Equation 2.5.9, we get:

$$\min_{\theta \in \mathcal{M}} \max_{\phi \in \mathcal{F}} \mathbb{E}_{p_\theta} h_\phi(x) - \mathbb{E}_{p_{\mathcal{X}}} h'_\phi(x) \tag{2.5.10}$$

A GAN is a differentiable model that learns adversarially by optimizing minimax objectives of the form given in Equation 2.5.10. The model is composed of two critical functions parametrized by neural networks: (a) a generator function $G_\theta : R^k \to R^d$ that maps a latent vector $z \in R^k$ to an observed data point $x \in R^d$, and (b) a discriminator or critic function $D_\phi : R^d \to R$ that maps $x$ to a scalar score. The discriminator is trained to classify between samples from the real data and samples produced by the generator. To obtain a data sample from the generator function, a latent vector is sampled in a tractable manner from a fixed prior distribution $z \sim p(z)$ (*e.g.*, isotropic Gaussian) and then perform a forward pass through the generator function as $x = G_\theta(z)$. Since the generator and discriminator are differentiable functions, we can leverage Equation 2.5.9 to optimize an appropriate divergence metric. As a special case, we obtain the original GAN objective proposed by Goodfellow et al. [2014a] when the discriminator minimizes the binary cross-entropy loss:

$$\min_{\theta \in \mathcal{M}} \max_{\phi \in \mathcal{F}} \mathbb{E}_{p_\theta}[\log(1 - D_\phi(G_\theta(z)))] - \mathbb{E}_{p_{\mathcal{X}}}[\log(D_\phi(x))] \tag{2.5.11}$$

The generator and the discriminator parameters are learned via gradient updates in an alternating fashion. In practice, GANs have been successful in generating photorealistic samples of high-resolution image datasets [Karras et al., 2018]. However, with the above formalism, they cannot be directly used to learn useful representations of the data, unlike VAEs. In addition to representation learning, another limiting factor of adversarial models is their evaluation mechanism. Family of models that maximize the likelihood have a straightforward, if not completely well-motivated, way to quantitatively evaluate the performance. For

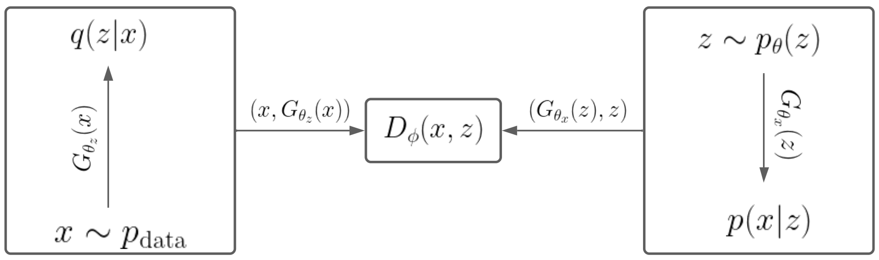


**Fig. 2.2.** Model depicting Adversarially Learned Inference (ALI)

adversarial approaches, no such criteria is readily apparent. Although classifier induced metrics are widely used in measuring the quality of adversarial models [Salimans et al., 2016, Heusel et al., 2017] they still lack statistical consistency guarantees and/or are often correlated to visual sample quality.

### 2.5.5. Adversarially Learned Inference

Adversarial Learned Inference(ALI) [Dumoulin et al., 2016] or BiGAN [Donahue et al., 2016] has shown that the adversarial learning paradigm can be extended to incorporate the learning of an inference network. Along with generator and discriminator networks like GAN, ALI consist of additional encoder network similar to a VAE. However, unlike a variational approach, ALI takes a pure adversarial approach to learn the encoding function.

While the encoder, maps training examples $\boldsymbol{x}$ to a latent space variable $\boldsymbol{z}$, the decoder plays the role of the standard GAN generator mapping from space of the latent variables (that is typically sampled from some factorial distribution) into the data space. In ALI, the critic is trained to distinguish between the encoder and the decoder, while the encoder and decoder are trained to conspire together to fool the critic. In particular, The critic is then trained to discriminate between the joint distribution of the data and latent causes coming from the generator and inference network. The model is shown in the 2.2

Thus, the ALI objective encourages a matching of the two joint distributions, which also results in all the marginals and conditional distributions being matched. This enables inference on the latent variables. The adversarial game played between the discriminator and the generator is formalized by the following value function:

$$\mathcal{L}_{ALI}(G,D) := \min_{\theta \in \mathcal{M}} \max_{\phi \in \mathcal{F}} \mathbb{E}_{p_\theta}[\log(1 - D_\phi(G_{\theta_x}(z),z))] + \mathbb{E}_{p_\mathcal{X}}[\log(D_\phi(x, G_{\theta_z}(x)))] \qquad (2.5.12)$$

In addition to stabilize the training dynamics of the GAN, such constrained optimization gives rise to useful representations of the data. These rich representations can further be used for downstream tasks such as semi-supervised learning that outperform the features learned by GAN [Salimans et al., 2016]. In addition to learnt representations, having an encoding-decoding frameworks allows one to reconstruct back the input data points from the

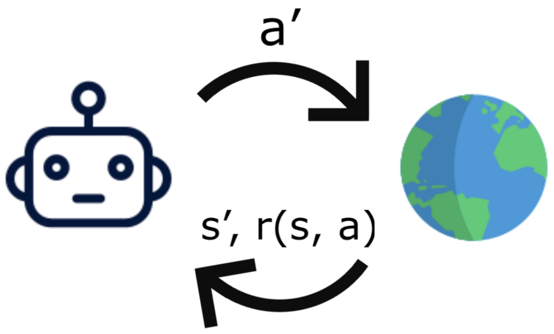


**Fig. 2.3.** Agent - Environment interaction in a Reinforcement Learning setup

feature space. Though the reconstructions are not perfect unlike VAEs(Note that ALI does not have an explicit term in its objective to encourage pixel-wise reconstruction) they retain salient features of the underlying data. This allows a mechanism to interpolate along the input space smoothly with the intermediate data points exhibiting a closeness to the data distribution in its characteristics.

ALI enjoys the benefit of both realistic generations and learning rich representations of the data.

## 2.6. Reinforcement Learning

Reinforcement learning (RL) allows an agent model to learn to make better decisions according to an ecosystem and a reward. The emphasis is on learning from experience without having to worry about expert behavior or supervised data. Deep RL combines RL with the expressiveness of neural networks to provide the agent with better representations so that it can perceive and understand the world better.

### 2.6.1. Learning Problem

The principal objective of RL is to obtain an optimal behavior for an agent deployed in an ecosystem or environment. The optimal behavior policy $\pi^*$ is desired to maximize the expected cumulative discounted reward (also known as value), as shown in Equation 2.6.1.

$$\pi^* = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t))\right] \tag{2.6.1}$$

Before describing solutions to learn the optimal policy, it is crucial to understand each component of Equation 2.6.1 and conceptualize the reinforcement learning setting. The RL setting can be formalized as a Markov Decision Process (MDP), denoted with the tuple $\{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$, where $\mathcal{S}$ is the set of states; mainly possible environmental states the agent might encounter, $\mathcal{A}$ is the set of actions; possible interventions the agent can perform on the environment. $P$ denotes the state transition dynamics; $P(s,a,s')$ is the transition function, which maps state-action-successor-state tuples to the probability of that transition occurring. It can sometimes be useful to think of the transition function as a mapping between state-action pairs and successor states $P(s,a) \to s'$. For the case of a stochastic environment setting, it denotes a sample from a distribution over successor states. $r(s,a)$ is the reward function, which maps state-action pairs to environmental rewards. Finally, $\gamma$ is the discount factor (often, a scalar constant), which exponentially discounts future rewards, thereby controlling the relative importance of nearby versus distant rewards. $\gamma$ is normally considered as known or given just like the reward function. An overview of agent world interaction is shown in Figure 2.3.

### 2.6.2. Q-Learning

Having laid the groundwork for the reinforcement learning problem, we now move on to discussing possible solutions. Learning to maximize cumulative discounted rewards in MDPs requires the understanding of value function formalism. A value function is denoted by $V(s)$ or $Q(s,a)$ depending on whether or not it is a function of states or state-action pairs. Value functions map states of the environment to their reward to go or expected cumulative discounted reward (i.e. value). The recursive Bellman operator (shown in Equation 2.6.2) updates an estimate of the value function according to the results of environmental interaction and the previous value estimate. The idea is that the model minimizes the difference between when it predicts in next state and the reward that it actually obtained from the environment, to make sure those are accurate.

$$V(s) = \max_a \sum_{s'} P(s'|s,a)(r(s,a) + \gamma V(s')) \tag{2.6.2}$$

It can be shown that repeated application of Bellman operator is guaranteed to converge to the optimal value function in certain circumstances [Bellman, 1952]. Learning based on this equation is known as temporal difference (TD) learning.

Looking closely, we can see that the weighting factor of the Bellman update (Eq 2.6.2) is given by how probable the transitions are in the environment $P(s,a,s')$. Since the agent can wander around in this environment, it is convenient to view it in terms of sampled transitions instead of all possible pairs of transitions. Therefore we can replace the explicit usage of $P$ with an expectation over sampled transitions made in the environment. Taking this a step further, we can also switch the values of states to the values of state-action pairs. This is more convenient since the state values require access to the dynamics model to select actions ('choosing favorable next state), while the values of state-action pairs allow for choosing actions without the information of their resulting state ('choosing favorable action in the current state'). Merging these two changes we obtain the form shown in Equation 2.6.3.

$$Q(s,a) = \mathbb{E}\left[r(s,a) + \gamma \max_{a'} Q(s',a')\right] \tag{2.6.3}$$

Since we avoided the need to use the model transition function, the resultant second equation can be computed without a dynamics model in a straightforward manner by taking actions and observing their consequences. One can view this process as sampling from the underlying state and reward transition functions. Note that we can interpret Equation 2.6.3 as associating an initial estimate of the value of a state-action pair ($Q(s,a)$) to a new estimate taking into account the knowledge about the actual immediate reward sampled from the environment. For the optimal value function, there should be no difference between the two estimates in expectation. This is because the initial estimate should already have considered how likely different immediate rewards were and how valuable the next state would be. We can thus view any differences between these two estimates as an error signal and use it to rectify the initial estimate. This difference between estimates taken at two different times, the temporal difference (TD) error, can be used as a loss function for training a neural network with gradient updates by minimizing the mean squared error shown in Equation 2.6.4

$$\mathcal{L}_{TD}(\pi) = \mathbb{E}\left[(Q(s,a) - (r(s,a) + \gamma \max_{a'} Q(s',a')))^2\right] \tag{2.6.4}$$

In order to act greedily with respect to a value function, we need to learn the underlying policy $\pi(s) = \operatorname{argmax}_a Q(s,a)$, a function mapping states to actions. This is trivial for small discrete action spaces, as $\operatorname{argmax}_a Q(s,a)$ can be determined by iterating over all possible actions, but intractable for continuous action spaces as we cannot iterate over an infinite set in finite time. Hence maximization in such setting is nontrivial. In continuous action settings, one popular approach to predict the most rewarding actions is to combine a model that learns to output the best action given a certain state, referred to as the actor model, and a model that learns to estimate the expected value of the actor's actions over time, given a certain state, referred to as the critic model. This gives rise to the actor-critic framework, which is primarily used in the later chapters of this thesis.

### 2.6.3. Deep Q-Networks

There are at least two reasons why optimizing temporal difference loss in Equation 2.6.4 could pose difficulties. Firstly, i.i.d sampling from an MDP is non-trivial, and secondly, there is a dependency between the current estimated values and target values. Both problems are addressed in Deep Q Network (DQN), an algorithm for solving reinforcement learning problems involving arbitrary non-linearities in the value function [Mnih et al., 2015]. In order to address i.i.d concerns, DQN uses random samples from a 'replay buffer' of past transition samples $s$,$a$,$r$,$s'$. Furthermore, to break the dependence between estimated and target values a slow changing *'target network'* is used.

DQN has become the standard baseline for the follow-up approaches. In some sense, this represents the pinnacle of reinforcement learning as a field, with most of the follow-up work focused on improving the network architecture and filling out appropriate optimization procedures to solve more large-scale complex problems.

Similar to DQN, Actor-critic algorithms can be combined with the expressiveness of neural network models to solve complex continuous control tasks [Haarnoja et al., 2018, Lillicrap et al., 2016, Schulman et al., 2017]. However, both DQN and actor-critic approaches are data hungry, as they require thousands of interactions with the environment before attaining the optimal behavior. With this background, we breifly discuss the notion of model-based control, which helps avoid enormous amounts of interactions necessary to attain optimal behavior via learning an environment model.

### 2.6.4. Model-based Control

In Equation 2.6.2 we estimated the value function without explicitly knowing the state transition dynamics function $P$. Instead, we could aim to increase the data-efficiency by learning the dynamics model. Having the knowledge about a model of the world would allow one to interact with our model in addition to the real environment, improving the sample-efficiency. Furthermore, the additional model information turns the Bellman equation 2.6.2 into an update operator. The resulting value iteration procedure has been shown to converge to a global optimum for sufficiently large number of iterations [Bellman, 1952]. However, the update requires an explicit enumeration of all possible states in the environment, which makes it difficult to scale on non-tabular environments.

Model-based Reinforcement Learning (MBRL) is related to the optimal control and planning literature, where most typical approaches resort to sampling action sequences. A simplest version of this approach entails sampling candidate actions from a fixed distribution (*e.g.* multivariate Gaussian), evaluating them under a model, and choosing the most-promising action [Nagabandi et al., 2018]. Cross-Entropy Method (CEM [Rubinstein and Kroese, 2004]) iteratively adjusts the sampling distribution for better performance. In continuous control, model-based RL combined with powerful search methods has led to impressive results on a wide variety of tasks [Hafner et al., 2019a]. For a discrete setting, search methods such as Monte Carlo Tree search [Coulom, 2006] and Sequential Monte Carlo planning [Piché et al., 2018] are more popular. This thesis makes use of the Model-Predictive Control (MPC) in the later chapters, which is based on CEM. These methods perform trajectory optimization by fitting a multivariate Gaussian distribution to the imagined future actions allowing them to search the space efficiently.

Critically, in MBRL, it is a non-trivial task to learn the model, especially on a high-dimensional state space. Nonetheless, obtaining the model can simplify the policy learning as noted earlier, allows model-based planning, and can lead to improved generalization. Once the dynamics model has been learnt, it can be reused even if the underlying reward structure changes in the environment. Furthermore, by being able to learn in the absence of interaction with the environment, model-based learning can adapt more rapidly than model-free methods. However, a model-free or a model-based learning paradigm that relies on dense reward supervision and can only learn after thousands of mistakes isn't generally a scenario we are interested in. With these details, we progress to the notion of unsupervised exploration and intrinsic motivation which is an important component of the thesis.

### 2.6.5. Intrinsic Motivation

Reward maximization is the fundamental learning objective in RL as shown earlier in Equation 2.6.1. For the sake of convenience, in many RL problem settings, dense reward supervision is given for the task at hand, such as a game score or inverse exertion for motor control. However, the notion of a reward is sparse or could be missing completely in many real-world tasks. The issue of sparse and delayed rewards is also related to the notion of intrinsic motivation in AI and Psychology. Especially, in the context of how natural agents learn, the notion of reward supervision is often fuzzy and ill-defined, being comprised of signals accumulated over evolutionary, cultural, and developmental timescales. Research in this area spans many fields, from computational accounts of useful intrinsic motivations [Barto et al., 2004] to empirical evidence for certain intrinsic costs in humans [Kool et al., 2013].

In reinforcement learning, the notion of intrinsic motivation/rewards is useful and gains prominence whenever extrinsic task rewards are sparse. Intrinsically motivated agents can explore new behavior for their own sake with a desire to fill missing knowledge than to directly solve problems. Such intrinsic behavior learning could aid an RL agent to efficiently adapt across tasks posed by the environment. Although it is unclear as to the exact nature and origin of good intrinsic reward functions, there have been extensive studies recently that streamlined the way one can categorize such approaches. Oudeyer and Kaplan [2008] classified intrinsic motivation algorithms into three different kinds: knowledge-based, competence-based and data-based models. Knowledge-based models rely on a prediction-error signal to build pseudo-rewards. They either maximize prediction error or uncertainty of some observable variable. Schmidhuber [2010] provided a coherent formulation of knowledge-based intrinsic motivation, which is measured by the performance of a predictive world model trained by the learning algorithm. Competence models aim to maximize the mutual information between the trajectory, observations and a goal state. Mohamed and Rezende [2015] have proposed a notion of intrinsically motivated learning within the framework of mutual information maximization. Frank et al. [2014] demonstrated the effectiveness of artificial curiosity using information gain maximization in a humanoid robot. Data-based methods try to increase the diversity of the dataset, often times through explicit maximum entropy objectives or count based objectives. Bellemare et al. [2016] proposed a count-based model for training deep RL agents to facilitate deep exploration.

There have been advances in studying intrinsic rewards from an evolutionary perspective [Singh et al., 2010]. Intrinsic motivation in humans has been shown to exhibit close resemblance to competence-based mechanisms. There is compelling evidence in the cognitive science literature that human babies and children prefer to represent the visual world in terms of coherent visual entities, centered around spatio-temporal principles of cohesion, continuity, and contact [Spelke and Kinzler, 2007, Lake et al., 2017]. Probing various objects with their hands and mouth, infants learn to differentiate relative sizes of objects, distances and higher order structural knowledge such as the ratio of object sizes. In the course of curiosity-driven exploration, infants use the acquired knowledge and inductive biases to generate self-acquired goals such as stacking stable block structures.

Although the thesis specifically focuses on knowledge-based models in the later chapters, the boundary across the categories is blurry. More generally, one can view knowledge-based and competence based models as specific forms of data-based exploration that aims to increase overall entropy of the action space to make progress.

## 2.7. Computer Graphics

A part of this thesis is aimed towards understanding scene structure from a single view of the scene using advances in computer graphics topics such as differentiable rendering and inverse graphics. This section provides a swift background on the same.

### 2.7.1. Image Rendering

Realistic image synthesis has long been a fundamental component of computer graphics with applications ranging from entertainment (e.g. feature films, special effects, or video games) to lighting design and architecture. Rendering process maps a 3D description of a scene to its corresponding set of 2D images (see Figure 2.4). The 3D information includes lighting (*e.g.*, the location and geometry of light sources in the environment), textures, shapes, material properties, poses, camera positions, and camera properties (collectively called the "scene parameters"). Modern renderers, with sufficiently detailed scene information, can produce photo-realistic images.

Traditionally, the rendering process follows a general formulation of the recursive shading equation (Equation 2.7.1) [Kajiya, 1986]. The equation describes how light reflects off objects in a scene to produce color. It is based on the simulation of the inter-reflection of light in an

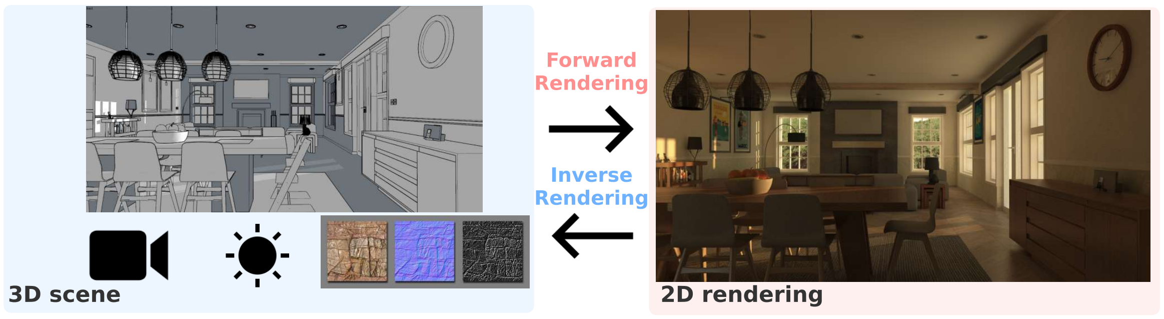


**Fig. 2.4.** Overview of Image Rendering. It is a mapping that takes a 3D scene as an input and outputs a 2D image.

environment in an effort to accurately describe the appearance of objects, while accounting for all of the paths that light can take from light sources to objects in the scene. The "intensity" of light traveling from one point $x$ to another $y$ is determined by

$$L_{out} = L_e + \int_{\mathcal{M}} L_i \cdot f \cdot cos(\theta) \cdot d\omega \tag{2.7.1}$$

where:
- $L_{out}$ is the reflected or outgoing radiance at a point.
- $L_e$ is the emitted radiance at this point.
- $L_i$ is incoming radiance to this point.
- $f$ is the Bidirectional Reflectance Distribution Function (BRDF) of the scene surface, that quantifies how light is reflected.
- $\theta$: angle between incoming direction and the normal to the surface.

The integral is over the area $\mathcal{M}$ of all scene surfaces and weighted by a purely geometric factor involving cosines of the incident and outgoing angles. The camera model that is used is a pinhole camera model [Sturm, 2014] which is defined by its origin and the image plane.

The equation can be further simplified depending on the different assumptions one takes into consideration for a given problem (*e.g.* surface reflectance properties). This formulation as an integral, based on physical principles, converts the problem of rendering a scene into the problem of evaluation. This provides a mechanism to apply novel analytical tools to rendering.

## 2.7.2. Inverse Graphics

Inverse rendering aims at recovering the scene parameters that produced the images. Unfortunately, the inverse rendering problem is ill-posed. This is because a set of images can be potentially explained by many possible scenarios. For example, if an object appears small in an image, it may be due to any of the following scene parameters: (a) pose: the object is far away from the camera, (b) shape: the object is tiny, or (c) camera properties: the camera

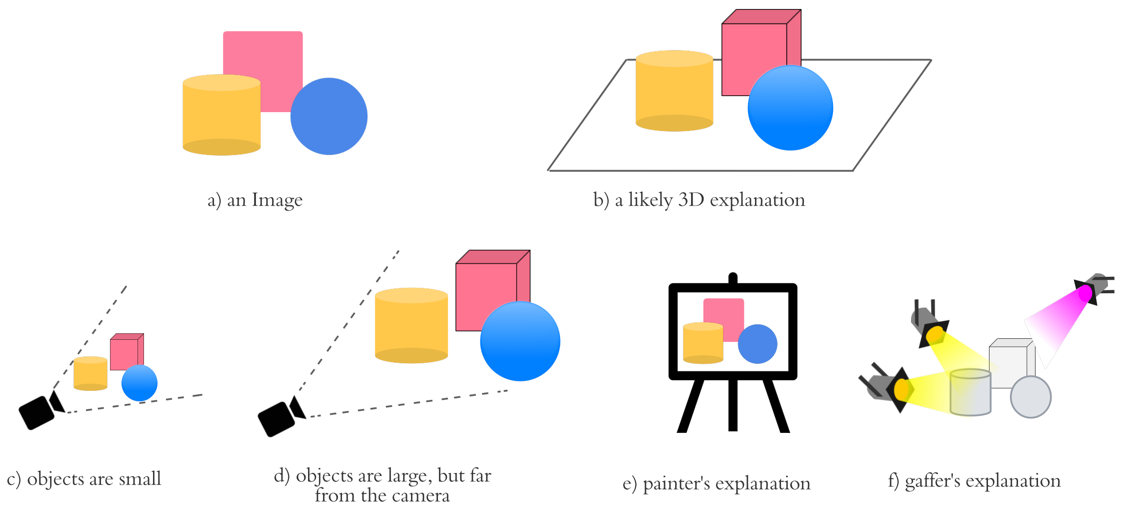


**Fig. 2.5.** The image in (a) clearly corresponds to the scene configurration in (b), but it could be that the objects all have a small size as in (c), or they are large but farther away from the camera, or it could be a painting, or an arrangement of lights over objects with uniform reflectance.

has a wide field of view. In fact, in the extreme setting, all images could theoretically be generated by a scene wherein a large 2D canvas is placed directly in front of the camera and painted with precisely the contents of the image as illustrated in Figure 2.5

### 2.7.3. Differentiable Rendering

Making rendering differentiable would allow one to leverage gradient based learning algorithms such as deep learning for the problem solving. Once a renderer is differentiable, it can be integrated into the optimization of the aforementioned neural network pipelines that leverage gradient descent. These pipelines can then be used to solve inverse graphics problems such as 3D reconstruction from a 2D image. Modern breakthroughs in computer graphics with reasonable assumptions and simplifications to Equation 2.7.1 have enabled differentiable rendering [Loper and Black, Liu et al., 2019b, Kato et al., 2018], which can be used towards addressing the inverse rendering problem. A differentiable renderer converts scene parameters into an image in an end-to-end differentiable way. Therefore, the inverse rendering problem can be solved by fixing the images and learning the scene parameters iteratively via gradient descent. However, optimizing these parameters naively results in poor reconstructions because the optimization landscape is filled with local minima and, as previously stated, the inverse rendering problem is ill-posed.

Some differential rendering techniques use voxel based representations to explain color, some neural weight [Lombardi et al., 2019], or a function measuring the distance to the nearest surface [Jiang et al., 2020]. Unfortunately, voxels have limited resolution and are memory intensive. Meshes do not suffer from these problems, however, differentiable mesh rendering is a difficult problem because of the non differentiability of visibility terms in the rendering equation [Li et al., 2018]. In our work described in the later chapter 4 we overcome some of these issues by using a novel surfel-based implicit representation which is viewpoint-dependent and it adapts the available surfels depending on the camera position.

# Chapter 3

# Prologue to Article 1

## 3.1. Article Details

**Pix2Shape: Unsupervised Learning of 3D Scenes from Images using a View-based Representation**. Sai Rajeswar, Fahim Mannan, Florian golemo, David Vazquez, Derek Nowrouzezahrai, Aaron Courville. Published at International Journal of Computer Vision (IJCV), 2020.

*Personal Contribution* This work was led by the author. The project began with joint discussion sessions involving the author, Aaron Courville and Derek Nowrouzezahrai about trying to represent the structure of a scene using a latent variable model, where the learned latent space captures the geometry of a scene. During the process the author was involved in designing the generative model and writing the code, running the experiments, and writing parts of the paper. Aaron Courville and Derek Nowrouzezahrai advised on the project and were involved in discussing results, suggesting experiments to run, and writing parts of the paper. Fahim Mannan was instrumental in making our differentiable rendering component efficient and compatible with the Generative model. All authors participated in weekly discussions and helped with preparing the manuscript.

## 3.2. Context

The goal of this work was to infer the geometry of a scene from a 2D image using probabilistic generative modeling. Given a single view of a scene, human perception has the ability to imagine the scene from a different view. We propose to learn interpretable 3D representation of the scene(world) analogous to human mental imagination of the scene. The learnt latent representation of an image can then be used to decode the scene from a novel view. The subsequent chapter (Chapter 4) explores this method in detail.

## 3.3. Research Impact

The paper addresses the drawbacks of existing 3D aware synthesis and reasoning models at the time. Prior methods either leveraged multiple views of the scene and/or were limited to simple shapes. The proposed method alleviates both the issues by using novel view-based 3D surfel representations and probabilistic modeling. Ours is one of the preliminary works that combine Adversarial learning and differentiable rendering, and subsequent works added to the lines of work presented in this chapter [Nguyen-Phuoc et al., 2019, Chan et al., 2021]. Purely depending on the imaginative capabilities of a generative model could lead to inconsistencies in the underlying geometry [Nguyen-Phuoc et al., 2019]. While relying solely on procedural rendering could constrain the scale of 3D modelling [Rezende et al., 2016]. This work balances the trade-off by leveraging the scaling strengths of GANs and grounding the extracted geometry on physical principles. In addition, as part of the work, we open-sourced the 3D-IQTT, which is one of the first prominent benchmarks that quantifies a representation learning algorithm for a 3D spatial reasoning task.

**Note:** This paper is presented as-is, with minor cosmetic changes to adhere to the Universite de Montreal thesis template

# Chapter 4

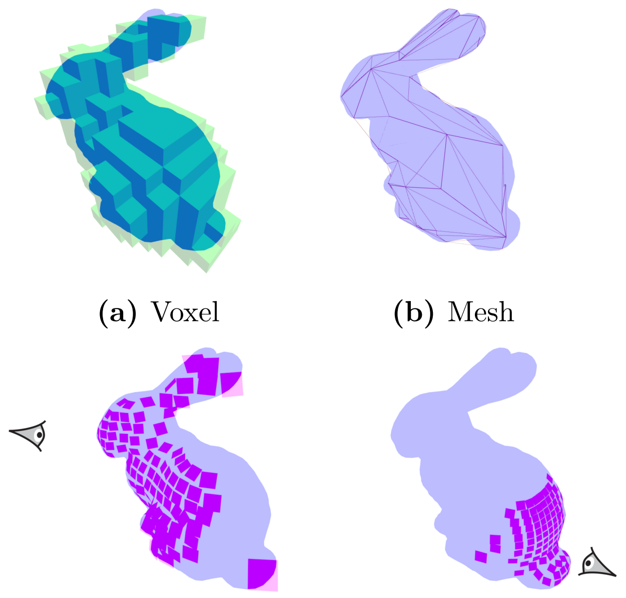
# Pix2Shape – Towards Unsupervised Learning of 3D Scenes from Images using a View-based Representation

## Abstract

We infer and generate three-dimensional (3D) scene information from a single input image and without supervision. This problem is under-explored, with most prior work relying on supervision from, e.g., 3D ground-truth, multiple images of a scene, image silhouettes or key-points. We propose **Pix2Shape**, an approach to solve this problem with four component: (i) an encoder that infers the latent 3D representation from an image, (ii) a decoder that generates an explicit 2.5D surfel-based reconstruction of a scene – from the latent code – (iii) a differentiable renderer that synthesizes a 2D image from the surfel representation, and (iv) a critic network trained to discriminate between images generated by the decoder-renderer and those from a training distribution. Pix2Shape can generate complex 3D scenes that scale with the view-dependent on-screen resolution, unlike representations that capture world-space resolution, i.e., voxels or meshes. We show that Pix2Shape learns a consistent scene representation in its encoded latent space, and that the decoder can then be applied to this latent representation in order to synthesize the scene from a novel viewpoint. We evaluate Pix2Shape with experiments on the ShapeNet dataset as well as on a novel benchmark we developed – called 3D-IQTT – to evaluate models based on their ability to enable 3d spatial reasoning. Qualitative and quantitative evaluation demonstrate Pix2Shape's ability to solve scene reconstruction, generation and understanding tasks.

## 4.1. Introduction

Humans sense, plan and act in a 3D world despite only directly observing 2D projections of their 3D environment. Automatic 3D understanding seeks to recover a realistic underlying



**(a)** Voxel **(b)** Mesh

**(c)** Surfels (distant camera) **(d)** Surfels (up close)

**Fig. 4.1. Comparison of 3D representations.** Voxels and meshes (4.1a and 4.1b) are viewpoint-independent representations. These representations require storage space proportional to the required level of detail. Our implicit representation captures the full scene in a fixed-length latent vector, which, given a viewpoint, can be decoded into an explicit viewpoint-dependent "surfels" representation with arbitrary level of detail (4.1c and 4.1d).

3D structure of a scene using only 2D image projection(s). This long-standing challenge in computer vision has recently admitted learning-based solutions. Many such approaches leverage 3D supervision, such as from images annotated with ground truth 3D shape information [Girdhar et al., 2016, Wu et al., 2015, 2016b, Choy et al., 2016]. Recent approaches rely on using other forms of 3D supervision, such as multiple views of the same object [Yan et al., 2016, Tulsiani et al., 2017, Li et al., 2019c], 2.5D supervision [Wu et al., 2016a, 2017], key-point [Kar et al., 2014, Novotný et al., 2019] and silhouette annotations [Wiles and Zisserman, 2017, Henderson and Ferrari, 2018, Chen et al., 2019]. Our work treats the problem of *unsupervised single image 3D scene understanding*. This form of the problem is challenging, as we aim to infer an encoding of 3D structure from only a *single image*, and this too without any form of 3D ground truth supervision during training. We do not rely on any 3D scene supervision, however we employ camera pose, scene reflectance profiles and outgoing/observed radiance as weak supervision signals.

While the benefits of employing supervision can certainly be argued for in this context – i.e., with the growing number of datasets with labelled 3D ground truth for objects [Chang et al., 2015] and cityscapes [Caesar et al., 2019] – one benefit of approaching the problem from an unsupervised perspective is that we are not limited to the types of 3D objects represented in these datasets. Indeed, however vast, existing datasets fall far from capturing all possible artificial and natural 3D scenes and objects. Moreover, datasets with depth annotations often contain incomplete or noisy depth maps due to limitations in depth capture hardware.

Unsupervised single image 3D understanding is a relatively under-explored area, with only a few works treating this setting Rezende et al. [2016], Yan et al. [2016]. These methods rely on deformable 3D mesh or voxel representations of the world, and have only been applied to simple 3D primitives (e.g., cubes, spheres) or single objects over a clean background.

One approach to this problem is to leverage prior knowledge on how 2D images are formed from the 3D world, including the effects of shading and occlusion. Building machine learning architectures with an explicit knowledge of this *forward rendering* model could help better disambiguate the 3D structure of geometry from 2D observations. In this spirit, we propose the **Pix2Shape** architecture for unsupervised single image 3D understanding: a model that learns abstract latent encodings of the geometry of an entire scene geometry, and all from a single image. These implicit learnt scene representations can be decoded – when combined with a targe viewing/camera position – into a view-dependent realization of 2.5D surfaces (depth map and surface normals) visible only from that view. We can then readily re-render these explicit view-dependent surface elements (surfels) at their corresponding 2D image projections in order to synthesize an unseen view of the scene.

Our model builds atop Adversarially Learned Inference (ALI) [Dumoulin et al., 2016], an extension of Generative Adversarial Networks (GANs) [Goodfellow et al., 2014b] that infers a latent code from an image using an encoder network. In Pix2Shape, the encoder network learns a latent representation that embeds the 3D information of *an entire scene* from an image. We map the latent representation to view-dependent depth and normal maps using a decoder before projecting these maps onto image space using a differentiable renderer. We evaluate the resulting image using an adversarial critic. Our model remains unsupervised as it does not require ground truth depth maps nor any other kind of 3D supervision, as in previous works (e.g., observing the same object from multiple views, key-point registration or image silhouettes). Note that, at any given instant, our model outputs the depth and surface normals conditioned on a specific camera view; we never produce/synthesize the entirety of the 3D world structure. That being said, the latent space we learn embeds the 3D geometry of the entire underlying scene, which allows our decoder and renderer to smoothly extrapolate and synthesize scene geometry from unseen camera views during inference. We refer to this indirect process of embedding 3D information in the latent code as "*implicit*" inference.

An ambitious long-term goal is to infer the 3D structure of photographs of the real-world, and our work takes a first step in this direction: We rely on physically based rendering in-order to build a model of the world. However, in order to make the training tractable we experiment exclusively with synthetically constructed scenes, adopting several simplifying assumptions. Of note, we assume that the world is composed of piece-wise smooth 3D elements and that,

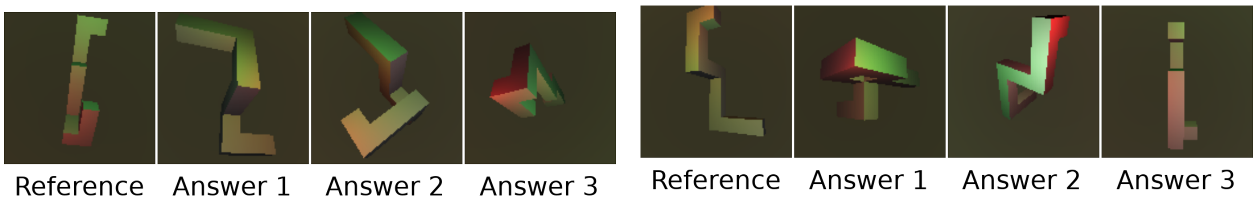


**Fig. 4.2. Sample questions from 3D-IQTT.** For this "mental rotation" task, we present a reference image and three possible answers. The test is a classification task where the goal is to find the rotated view of the model from the reference image. To solve this task, the 3D shape of the reference must be inferred from the 2D image and compared to the inferred 3D shapes of the answers (see footnote for correct answers).

for each input image, the illumination, view and object materials are known. Since each pixel in an image is a function of geometry, illumination, view and texture, our focus in this work is to learn the underlying geometry of a scene keeping the other parameters fixed.

We evaluate our model's ability to recover accurate and consistent depth from a single image, for both seen and unseen viewpoints, using Hausdorff and Chamfer distance metrics between generated and ground truth depth maps. In addition to reconstruction, we can sample novel scenes (at novel views) using the generative nature of our adversarial network. Finally, we propose a new 3D understanding benchmark – *3D IQ Test Task* (3D-IQTT) – to evaluate models' understanding of the underlying 3D structure of an object: the test consists of matching a rotated view of a reference object (Figure 4.2). To perform this task, we develop a novel 3D-IQ dataset to train and test against. In this setting, we can additionally estimate camera pose in our learnt latent 3D world embedding. Our contributions are as follows:

- an approach for unsupervised single image 3D understanding that builds a latent embedding of an entire 3D scene,
- a decoding scheme that leverages view-dependent, explicit surfel representations to sample scene information more efficiently than (world-space) voxels and meshes,
- a differentiable 3D renderer that we leverage, and that can be included as a layer in any learning-based neural network architecture, and
- 3D-IQTT, a new 3D understanding benchmark.

## 4.2. Related Work

Several works in recent years have applied recent machine learning advances to SLAM or have reformulated a subset of *components* of the full SLAM system in a differentiable manner.

### 4.2.1. Single view 3D Reconstruction and Generation

3D generation and reconstruction has been studied extensively in the computer vision and graphics communities [Saxena et al., 2009, Chaudhuri et al., 2011, Kalogerakis et al., 2012, Chang et al., 2015, Rezende et al., 2016, Soltani et al., 2017, Kulkarni et al., 2015, Tulsiani et al., 2016, Huang et al., 2019a, Jiang et al., 2019]. Most methods in the literature focus on recovering the 3D structure from 2D images by using explicit 3D supervision. Choy et al. [2016], Girdhar et al. [2016], Wu et al. [2016b, 2015], Zhu et al. [2018a] reconstruct and/or generate 3D voxels from a latent representation by directly comparing with available 3D shapes. Wu et al. [2017], Zhang et al. [2018] use 2.5D supervision during training, i.e., depth maps. More recent methods tend to use weaker forms of supervision for single image reconstruction. Wu et al. [2016a], Kato and Harada [2018], Henderson and Ferrari [2018], Chen et al. [2019] use image based annotations like silhouettes, 2D keypoints or object masks. Kanazawa et al. [2018] learn both texture and shape from 2D images leveraging multiple learning signals such as keypoints and mean shape.

Rezende et al. [2016], Yan et al. [2016], Gadelha et al. [2016], Novotný et al. [2017] learn 3D shapes by using multiple views and approximately differentiable rendering mechanisms. However, one of Rezende et al. [2016]'s experiments show reconstruction 3D objects trained using a single view. As far as we know, theirs is the only fully unsupervised method for explicit 3D reconstruction from a single image. Their method is limited to reconstructing relatively simple 3D primitives floating in space due to the strong priors required for the model to work. Concurrent to our work, HoloGAN [Nguyen-Phuoc et al., 2019] can synthesize 2D images of more realistic scenes (e.g., cars, bedrooms) under camera view rotation. However their model can not recover the geometry from its implicit representation. Compared to Rezende et al. [2016], our model can learn to represent more complex synthetic indoor scenes composed of multiple ShapeNet[Chang et al., 2015] objects and, while we do not address real image inputs (i.e., as HoloGAN), we can infer explicit geometry for visible surfaces from each given view. As such, our model can also be applied to 3D reconstruction (like Rezende et al. [2016] but only for visible parts of the scene) and novel viewpoint image generation (like Nguyen-Phuoc et al. [2019]).

### 4.2.2. Differentiable Rendering

In order to facilitate deep neural network based models to infer 3D structures from their 2D projections (images), it is required to compute and propagate the derivatives of image pixels with respect to 3D geometry and other properties. Gradient estimation through rendering process is a challenging task. In both rasterization and ray-tracing techniques the visibility mapping step is often non-differentiable. Loper and Black is one of the well known methods for differential rendering, but has limited applicability due to high computational and memory costs. Kato et al. [2017], Rezende et al. [2016] approximate the gradients of the rendering process and are often limited to a rasterization based rendering scheme. OpenDR [Loper and Black], as used by Henderson and Ferrari [2018], applies first order Taylor approximation to compute gradients. Liu et al. [2019a] computes the gradients analytically by softly assigning contribution of each triangle face to a pixel in mesh-based representations. Chen et al. [2019] improved this soft assignment and allow the use of textures by interpolating local mesh properties for foreground pixels. Insafutdinov and Dosovitskiy [2018] proposed a differentiable re-projection mechanism for point clouds to infer 3D shapes. However learning methods built on these approaches so-far require either more than one view per object or 2D silhouette as supervision and can only reconstruct single objects. In our work we circumvent the non-differentiablity challenge as follows: (1) Our network is trained to output only “visible" surface elements (surfels) of the scene conditioned on the view, i.e. a 2.5D representation and (2) We maintain one-to-one correspondence between the output surfels and the pixels. In other words our model outputs exactly one surfel in object space per pixel in the output image, and the final image is then formed by a differentiable shading operation. This makes our model differentiable, easily adaptable across image resolutions and allows end-to-end training.

## 4.3. Pix2Shape

Our method follows the ALI architecture [Dumoulin et al., 2016], where we have an encoder branch that learns to convert images into latent representations, a decoder branch that learns to generate images from randomly sampled latent representations, and a critic that tries to predict if pairs of latent code and image are real or fake. The critic and encoder pathways are implemented as convolutional neural networks but the decoder pathway contains an additional differentiable renderer, usable like a layer of a neural network, that converts the 2.5D surfel representation into a 2D image by computing shading at each surfel. Additionally, the decoder is conditioned on a camera pose. See Figure 4.3 for an overview. In the following section, we drill down on the individual components of this architecture.

### 4.3.1. 3D Representation and Surfels

Representing 3D structure as voxels or meshes presents different challenges for generative models [Kobbelt and Botsch, 2004]. Representing entire objects using voxels scales poorly given its $O(n^3)$ complexity. Additionally, the vast majority of the generated voxels are not relevant to most viewpoints, such as the voxels that are entirely inside objects. A common workaround is to use a surface representation such as meshes. However, these too come with their own drawbacks, such as their graph-like structure. This makes mesh representation

[1] three [2] two

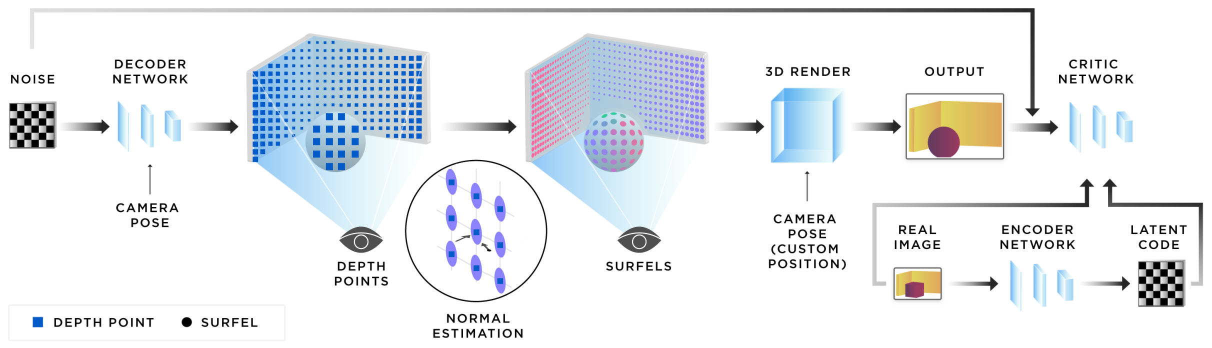


**Fig. 4.3. Model.** Pix2Shape generates realistic 3D views of scenes by training on 2D single images only. Its decoder generates the surfel depth map $p_z$ from a noise vector $\boldsymbol{z}$ conditioned on the camera pose. The surfel normals are estimated from the predicted depth. The surfels are rendered into a 2D image and, together with image samples from the target distribution, are fed to the critic, which generates a gradient for both encoder and decoder paths.

difficult to generate using neural networks. Current mesh based methods mainly rely on deforming a pre-existing mesh and are thus limiting the object topology to have the same genus as the template mesh.

Our approach represents the 3D scene implicitly in a high-dimensional latent variable. In our framework, this latent variable (i.e., a vector) is decoded using a decoder network conditioned on the camera pose into a viewpoint-dependent representation of surface elements (i.e., surfels [Pfister et al., 2000], square-shaped planes that are scaled based on depth to roughly fit the size of a pixel) that constitute the visible part of the scene. This representation is very compact: given a renderer's point of view, we can represent only the part of the 3D surface needed by the renderer. As the camera moves closer to a part of the scene, surfels become more compact and thereby increase the amount of visible detail. For descriptive purpose we discuss surfels as squares, but in general they can have any shape. Figure 4.1 compares surfels with different representations. Surfels differ from other explicit representations in that they are view-dependent, i.e., this representation changes for different camera poses (but the implicit latent vector representation does not).

Formally, surfels are represented as a tuple $(P, N, \rho)$, where $P = (p_x, p_y, p_z)$ is its 3D position, $N = (n_x, n_y, n_z)$ is the surface normal vector, and $\rho = (k_r, k_g, k_b)$ is the albedo of the surface material. Note that $\rho$ represents the material properties at the point $P$ and could take a different size for a different shading model. Since we are only interested in modelling structural properties of the scenes, i.e. geometry and depth, we assume that objects in the scene have uniform material properties and thus keep $\rho$ fixed. We also estimate the normals from depth by assuming locally planar surfaces. We represent the surfels in the camera coordinate system and generate one surfel for each pixel in the output image. This makes our representation very compact. Thus, the only necessary parameter for the decoder network to generate is $p_z$, i.e. a depth map.

### 4.3.2. Differentiable 3D Renderer

Since our architecture is GAN-like and uses 2D images as input to the critic network, we need to project the generated 3D representations down to 2D space using a renderer. In our setting, each stage of the rendering pipeline must be differentiable to allow us to take advantage of gradient-based optimization and backpropagate the critic's error signal to the surfel representation. Our proposed rendering process is differentiable because: (1) each output pixel depends exactly on one surfel, and (2) we employ a differentiable shading operation to compute the color of each pixel. Our PyTorch implementation of the differentiable renderer can render a $128 \times 128$ surfel-based scene in under 1.4 ms on a mobile NVIDIA GTX 1060 GPU. Further details about the rendering implementation can be found in appendix A.1.

### 4.3.3. Model

The adversarial training paradigm allows the generator network to capture the underlying target distribution by competing with an adversarial critic network. Pix2Shape employs bi-directional adversarial training [Dumoulin et al., 2016, Donahue et al., 2016] to model the distribution of surfels from 2D images.

4.3.3.1. Bi-Directional Adversarial Training. ALI [Dumoulin et al., 2016] or Bi-GAN [Donahue et al., 2016] extend the GAN [Goodfellow et al., 2014b] framework by including the learning of an inference mechanism. Specifically, in addition to the decoder network $G_x$, ALI provides an encoder $G_z$ which maps data points $\boldsymbol{x}$ to latent representations $\boldsymbol{z}$. In these bi-directional models, the critic, $D$, discriminates in both the data space ($\boldsymbol{x}$ versus $G_x(\boldsymbol{z})$), and latent space ($\boldsymbol{z}$ versus $G_z(\boldsymbol{x})$) jointly, maximizing the adversarial value function over two joint distributions. The final min-max objective can be written as:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{ALI}(G,D) = \; & \mathbb{E}_{q(\boldsymbol{x})}[\log(D(\boldsymbol{x},G_z(\boldsymbol{x})))] \\ & + \mathbb{E}_{p(\boldsymbol{z})}[\log(1 - D(G_x(\boldsymbol{z}),\boldsymbol{z}))],\end{aligned}$$

where $q(\boldsymbol{x})$ and $p(\boldsymbol{z})$ denote encoder and decoder marginal distributions.

4.3.3.2. Modelling Depth and Constrained Normal Estimation. The encoder network captures the distribution over the latent space of the scene given an image data point $\boldsymbol{x}$. The decoder network maps a fixed scene latent distribution $p(\boldsymbol{z}_{scene})$ (a standard normal distribution in our case) to the 2.5D surfel representation from a given viewpoint $\boldsymbol{z}_{view}$. The surfel representation is rendered into a 2D image using our differentiable renderer. The resulting image is given as input to the critic to distinguish it from the ground truth image data. To emphasize on the notation, note that the output of the encoder is $\boldsymbol{z}_{scene}$ and the input to decoder is $(\boldsymbol{z}_{scene}, \boldsymbol{z}_{view})$

A straightforward way to design the decoder network is to learn a conditional distribution to produce the surfels' depth ($p_z$) and normal ($N$). However, this could lead to inconsistencies between the local shape and the surface normal. For instance, the decoder can fake an RGB image of a 3D shape simply by changing the normals while keeping the depth fixed. To avoid this issue, we exploit the fact that real-world surfaces are locally planar, and that surfaces visible to the camera have normals constrained to be in the half-space of visible normal directions from the camera's view point. Considering the camera to be looking along the $-z$ axis direction, the estimated normal has the constraint $n_z > 0$. Therefore, the local surface normal is estimated by solving the following problem for every surfel:

$$N^T \nabla P = 0 \text{ subject to } \|N\| = 1 \text{ and } n_z > 0, \tag{4.3.1}$$

where the spatial gradient $\nabla P$ is computed for each of the 8 neighbour points, and $P$ is the position of the surfels in the camera coordinate system obtained by back-projecting the generated depth along rays.

This approach enforces consistency between the predicted depth field and the computed normals and provides a gradient signal to the depth from the shading process. If the depth is incorrect, the normal-estimator outputs an incorrect set of normals, resulting in an inconsistent RGB image with the data distribution, which in turn would get penalized by the critic. The decoder network is thus incentivized to produce realistic depths.

4.3.3.3. Unsupervised Training. The Wasserstein-GAN [Arjovsky et al., 2017] formulation provides stable training dynamics using the first Wasserstein distance between the distributions. We adopt the gradient penalty setup as proposed in Gulrajani et al. [2017] for more robust training. However, we modify the formulation to take into account the bidirectional training.

The architectures of our networks, and training hyper-parameters are explained in detail in the supplementary material section A.2. Briefly, we used Conditional Normalization [Dumoulin et al., 2016, Perez et al., 2017] for conditioning the viewpoint (or camera pose) in the encoder, decoder and the discriminator networks. The viewpoint is a three dimensional vector representing positional coordinates of the camera. In our training, the affine parameters of the batch-normalization layers [Ioffe and Szegedy, 2015] are replaced by learned representations based on the viewpoint. The final objective includes a bi-directional reconstruction loss:

$$\begin{aligned} \mathcal{L}_{recon} = \mathbb{E}_{q(\boldsymbol{x})}[||\boldsymbol{x} - \textsc{rend}(G_x(G_z(\boldsymbol{x})))||_2] + \\ \mathbb{E}_{p(\boldsymbol{z})}[||\boldsymbol{z} - G_z(\textsc{rend}(G_x(\boldsymbol{z})))||_2], \end{aligned} \tag{4.3.2}$$

where the $\textsc{rend}(\cdot)$ function synthesizes images through view-dependent decoding and projection and $\boldsymbol{z}$ is $(\boldsymbol{z}_{scene}, \boldsymbol{z}_{view})$. This objective enforces the reconstructions from the model to stay close to the corresponding inputs. This reconstruction loss is used for the encoder and

**Algorithm 1** Semisupervised classification

```
1: while $iter < max\_iter$ do
2:     $D \leftarrow$ MiniBatch()
3:     $z_x \sim Enc(\boldsymbol{x}); \forall x \in \{\boldsymbol{x}_{ref}, \boldsymbol{x}_{d_1}, \boldsymbol{x}_{d_2}, \boldsymbol{x}_{ans}\} \in D$
4:     $L \leftarrow \mathcal{L}_{ALI} + \mathcal{L}_{recon} + I_\Theta(z_{scene}, z_{view})$
5:     if supervised-training-interval($iter$) then
6:         $L \leftarrow L + \mathcal{L}_\theta$
7:     end if
8:     optimize networks with $L$
9: end while
```

decoder networks as it has been empirically shown to improve reconstructions in ALI-type models [Li et al., 2017a].

4.3.3.4. Semi-supervised Training for Classification. Our model can be also trained in a semi-supervised setting (see Algorithm 1) for solving image classification tasks that require 3D understanding such as the 3D-IQTT (See Figure 4.2). The idea is to use labeled examples to streamline the learned latent representations in order to solve the task. In this case, we do not assume that we know the camera position for the unlabeled training samples. Ass mentioned earlier, part of the latent vector $\boldsymbol{z}$ encodes the actual 3D object (denoted as $\boldsymbol{z}_{scene}$) and the remainder estimates the camera-pose (denoted as $\boldsymbol{z}_{view}$). For the supervised samples, two additional loss terms were used: (a) a loss that enforces the object component ($\boldsymbol{z}_{scene}$) to be the same for both the reference object and the correct answer, (b) a loss that maximizes the distance between the reference object and the distractors. This loss is expressed as:

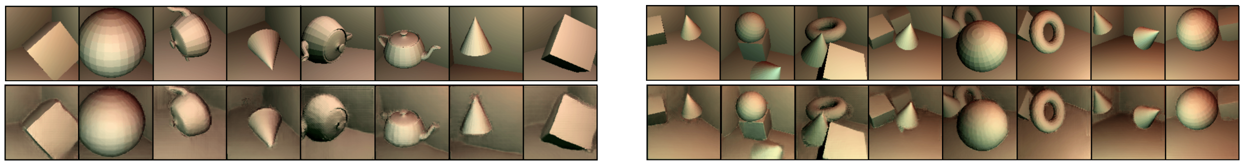
$$\mathcal{L}_\theta = \frac{1}{2} D_\theta(\boldsymbol{x}_{ref}, \boldsymbol{x}_{ans}) - \frac{1}{2} \sum_{i=1}^{2} D_\theta(\boldsymbol{x}_{ref}, \boldsymbol{x}_{d_i}) \qquad (4.3.3)$$

where $\boldsymbol{x}_{ref}$ is the reference image, $\boldsymbol{x}_{ans}$ is the correct answer, $d_i$ denotes the distractors, $D_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2) = (||\boldsymbol{z}^{\boldsymbol{x}_1}_{scene} - \boldsymbol{z}^{\boldsymbol{x}_2}_{scene}||_2)^2$ and $\boldsymbol{z}^{\boldsymbol{x}} = Encoder_\theta(\boldsymbol{x})$.

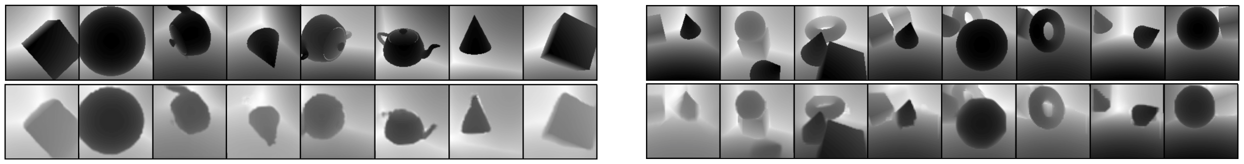
During training, we also minimize the mutual information between $\boldsymbol{z}_{scene}$ and $\boldsymbol{z}_{view}$ to explicitly disentangle both. This is implemented via MINE [Ishmael et al., 2018]. The strategy of MINE is to parameterize a variational formulation of the mutual information in terms of a neural network:

$$I_\Theta(z_s, z_v) = \sup_{\theta \in \Theta} \mathbb{E}_{\mathbb{P}_{z_s z_v}}[T_\theta] - \log\left(\mathbb{E}_{\mathbb{P}_{z_s} \otimes \mathbb{P}_{z_v}}[e^{T_\theta}]\right). \qquad (4.3.4)$$

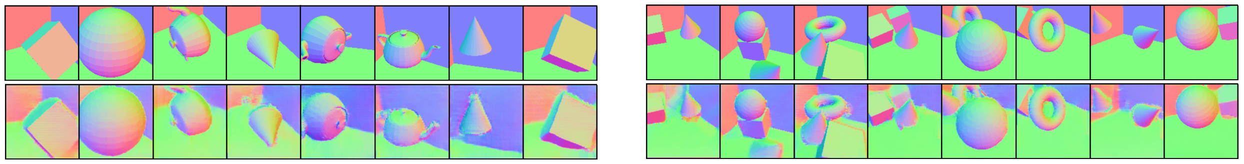
This objective is optimized in an adversarial paradigm where $T$, the statistics network, plays the role of the critic and is fed with samples from the joint and marginal distribution. We use this loss to minimize the mutual information estimate in both unsupervised and supervised training iterations. Once the model is trained, we answer 3D-IQTT questions, by inferring the latent 3D representation for each of the four images and we select the answer closest to the reference image as measured by $L_2$ distance on latent representations.



**(a)** Top: Input images. Bottom: Reconstructed images



**(b)** Top: Ground-truth depth maps. Bottom: Reconstructed depth maps



**(c)** Top: Ground-truth normal maps. Bottom: Reconstructed normal maps

**Fig. 4.4. Shape scenes reconstruction.** Pix2Shape reconstruction of single objects in a room (left) and multiple objects into a room (right). On both sides, the ground truths for RGB, depth, and normals are in the upper row, the inferred image, depth and normals are in the respective lower rows. Our model is able to correctly recover the depth and normal of the scenes from a single 2D image.

## 4.4. Experimental Setup

We evaluate Pix2Shape on three different tasks: scene reconstruction, scene generation, and 3D-IQTT.

### 4.4.1. Scene Reconstruction

The goal of this task is to produce a 2.5D representation (depth and normals) from a given input image. Moreover, we also evaluate if the model can extrapolate to unobserved views of the scene.

For this task we have created two datasets of scene images composed of a room containing one or more objects placed at random positions and orientations. *Shape scenes* dataset is created with rendered images of multiple basic 3D shapes (i.e., box, sphere, cone, torus, teapot etc)placed inside a room. *ShapeNet scenes* dataset is constructed from renderings of multiple objects of different categories from the ShapeNet dataset [Chang et al., 2015] (i.e., bowls, bottles, mugs, lamps, bags, etc).

Each 3D scene is rendered into a single $128 \times 128 \times 3$ image taken from a camera in a random position sampled uniformly on the positive octant of a sphere containing the room. The probability of seeing the same configuration of a scene from two different views is near zero.

We evaluate the performance of scene reconstruction using three different metrics: (1) Chamfer distance, (2) Hausdorff distance [Hausdorff, 1949] (on surfels' position), and (3) Mean Squared Error (MSE).

Chamfer distance (CD) gives the average distance from each point in a set to closest point in the other set. For any two point sets $A, B \subset \mathbb{R}^3$ Chamfer distance is measured using:

$$CD(A,B) = \frac{1}{|A|}\sum_{x\in A}\min_{y\in B}\|x-y\|_2 + \frac{1}{|B|}\sum_{x\in B}\min_{y\in A}\|x-y\|_2$$

Hausdorff distance (HD) measures the correspondence of the model's 3D reconstruction with the input for a given camera pose. Given two point sets, $A$ and $B$, the Hausdorff distance is,

$$\max\left\{\max D_H^+(A,B), \max D_H^+(B,A)\right\},$$

where $D_H^+$ is an asymmetric Hausdorff distance between two point sets. E.g., $\max D_H^+(A,B) = \max D(a,B)$, for all $a \in A$, or the largest Euclidean distance $D(\cdot)$, from a set of points in $A$ to $B$, and a similar definition for the reverse case $\max D_H^+(B,A)$. In both the evaluations, we mesaure compare our reconstructed view-centric surfels (3D positions and normals) to the groundtruth surfels.

### 4.4.2. Scene Generation

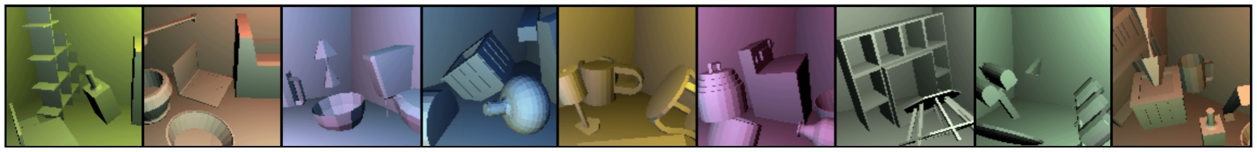
In the second task we showcase the generative ability of our model by using our generator to sample class conditioned shapes from ShapeNet dataset. We evaluate the 3D scene generation task qualitatively.

### 4.4.3. 3D-IQTT

In the final task we evaluate the 3D understanding capability of the model on 3D-IQTT: a spatial reasoning-based semi-supervised classification task. The goal of the 3D-IQTT is

| | **Ours** | | **PTN** | |
|---|---|---|---|---|
| | Shape scenes | ShapeNet scenes | Shape scenes | ShapeNet scenes |
| Chamfer distance (CD) | 0.103 | 0.133 | 0.145 | 0.181 |
| Hausdorff (HD) | 0.191 | 0.215 | 0.229 | 0.254 |
| MSE-depth | 0.038 | 0.053 | 0.056 | 0.103 |

**Table 4.1. Scene reconstruction results.** Evaluation of Pix2Shape on scene reconstruction with Chamfer distance and Hausdorff metric on 2.5D surfels and MSE on the depth maps. Table also compares with view-centric reconstruction of PTN Yan et al. [2016],


**(a)** Input images

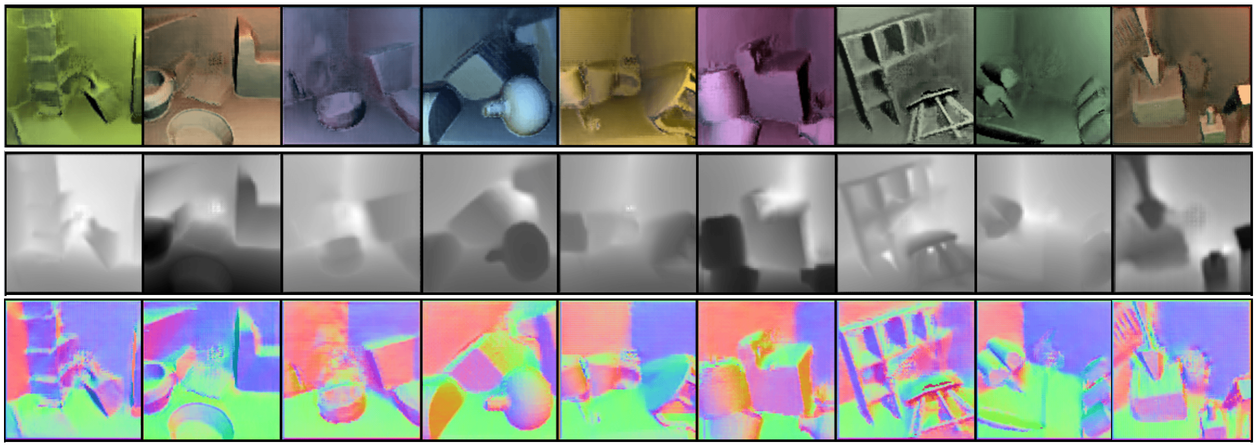

**(b)** Reconstructed images, depths and normals

**Fig. 4.5. ShapeNet scenes reconstruction.** Implicit 3D reconstruction of scenes composed by multiple ShapeNet objects.

to quantify the ability of our model to perform a 3D spatial reasoning test by using large amounts of the unlabeled training data and a small set of labeled examples.

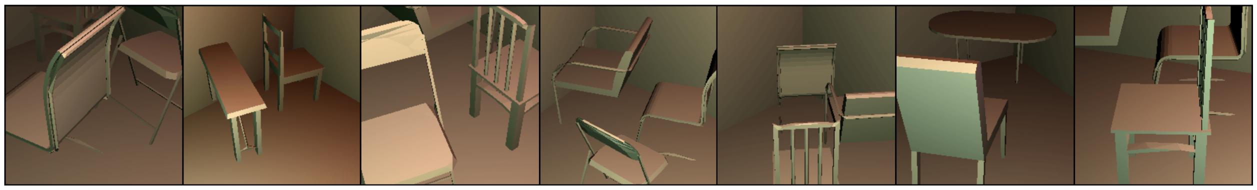
For this 3D-IQTT task, we generated a dataset where each IQ question consists of a reference image of a Tetris-like shape, as well as three other images, one of which is a randomly rotated version of the reference (see Figure 4.2 for an example). The training set consists of 100k questions where only a few are labeled with the information about the correct answer (i.e. either 1% (1k) or 0.2% (200) of the total training data). The validation and test sets each contain 100K labeled questions. Earlier literature related to 3D-IQTT is elaborated in supplementary material section A.8. We evaluate the 3D-IQTT task with the percentage of questions answered correctly.

More details on experimental setup and evaluation can be found in supplementary material sections A.4 and A.6.

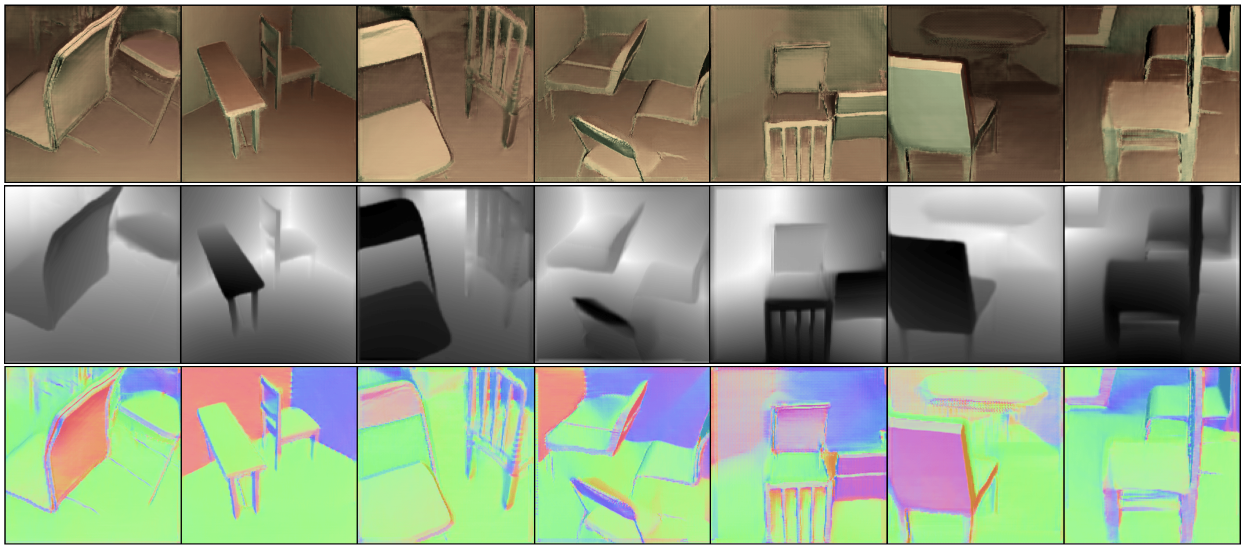
## 4.5. Experiments and results

### 4.5.1. Scene Reconstruction

Figure 4.4 shows the input *shape scenes* data and its corresponding shading reconstructions, along with its recovered depth and normal maps. The depth map is encoded in such a way that the darkest points are closer to the camera. The normal map colors correspond to



**(a)** Input images



**(b)** Reconstructed images, depths and normals

**Fig. 4.6. ShapeNet** $256 \times 256$ **scenes reconstruction.** Implicit 3D reconstruction of scenes composed by multiple ShapeNet objects.

the cardinal directions (red/green/blue for x/y/z axis respectively). Table 4.1 shows a quantitative evaluation of the Chamfer and Hausdorff distances on Shape scene and shapenet scene datasets from a given observed view. The table also depicts mean squared error (MSE) of the generated depth map with respect to the input depth map. The shading reconstructions are almost perfect in this simple dataset. Our model successfully learns the depth of the scenes and thereby the relative positions of the surfels. It also estimates the normal maps from the depth consistently. However the absolute distance is not always recovered perfectly.

Figure 4.5 shows the reconstructions from the model on challenging *ShapeNet scenes* where the number of objects as well as their shape varies. Note how our model is able to handle geometry of varying complexity. Figure 4.6 shows reconstructions on $256 \times 256$ resolution scenes(on the right) constructed out of more difficult thin-edged chairs and tables from ShapeNet dataset in random configurations.

To showcase that our model can reconstruct unobserved views, we first infer the latent code $\boldsymbol{z}_{scene}$ of an image $\boldsymbol{x}$ and then decode and render different views while rotating the camera around the scene. Table 4.2 shows the Chamfer and Hausdorff distances and MSE loss of reconstructing a scene from different unobserved view angles. As the view angle increases from 0°(original) to 80° for *shape scenes* the reconstruction error and MSE tend to increase.

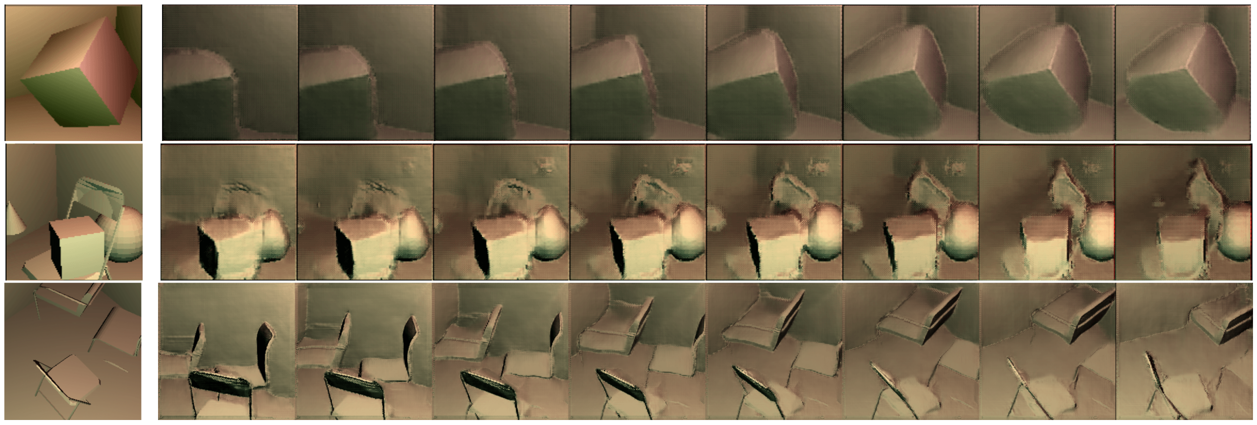


**Fig. 4.7. Viewpoint reconstruction.** Given a scene (first column), we rotate the camera around it to visualize the unseen parts of the scene. The model correctly infers the unobserved geometry of the objects, demonstrating true 3D understanding of the scene. Videos of these reconstructions can be seen at https://bit.ly/2zADuqG.

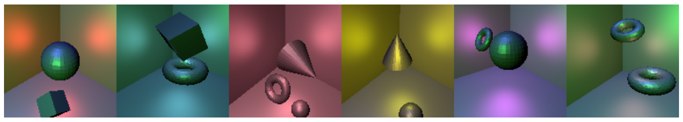
However, for the *ShapeNet scenes* the trend is not as clear because of the complexity of the scene and inter-object occlusions. We compare our method with the PTN baseline Yan et al. [2016]. Note that PTN reconstructs the 3D object in voxels explicitly, where as we output a 2.5D representation. Therefore, for a fair comparison we rotate and render per pixel depth map from a desired view and obtain the Chamfer distance with respect to ground truth projection for that view. Figure 4.7 qualitatively shows how Pix2Shape correctly infers the scene parts not in view demonstrating true 3D understanding.

In all our datasets and further experiments we use diffuse materials with uniform reflectance. The reflectance values are chosen arbitrarily and we use the same material properties for both the input and the generator side. However, our differentiable rendering setup also supports Phong illumination model. As an instance Figure 4.8 shows the input *shape scenes* data with specular reflection and its corresponding shading reconstructions, along with its recovered depth.

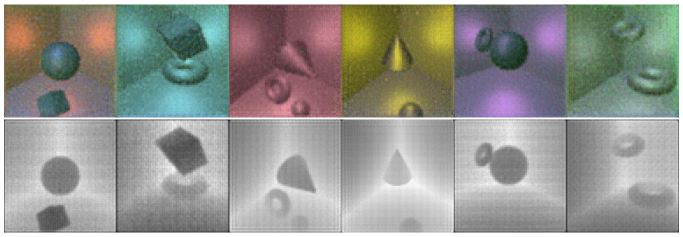
## 4.5.2. Scene Generation

We trained Pix2Shape on scenes composed of a single ShapeNet object in a room. The model was trained conditionally by giving the class label of the ShapeNet object present in the scene to the decoder and critic networks [Mirza and Osindero, 2014]. Figure 4.10 shows the results of conditioning the decoder on different target classes. Our model was able to generate accurate 3D models for the target class. We can also train the model in an unconditional fashion without giving any object category information (see supplementary material A.5 for more details and results).

In order to explore the manifold of the learned representations, we selected two images $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ from the held out data. We then linearly interpolated between their encodings



**(a)** Input images



**(b)** Reconstructed images and depths

**Fig. 4.8. Shape scenes reconstruction with specular reflectance.** Pix2Shape reconstruction of multiple objects into a room. The input RGB images are in the upper row, the inferred image and depth are in the respective lower rows.

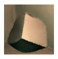
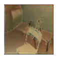
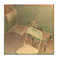
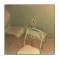
| | | **Shape scenes** | | | | **Multiple-shape scenes** | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | |  | | | |  |  |  |
| | | 5° | 35° | 55° | 80° | 5° | 35° | 55° | 80° |
| | HD | 0.156 | 0.191 | 0.189 | 0.202 | 0.308 | 0.355 | 0.329 | 0.316 |
| **Ours** | CD | 0.098 | 0.112 | 0.110 | 0.126 | 0.141 | 0.148 | 0.134 | 0.108 |
| | MSE | 0.012 | 0.021 | 0.022 | 0.027 | 0.070 | 0.091 | 0.088 | 0.083 |
| | CD | 0.143 | 0.189 | 0.219 | 0.202 | 0.174 | 0.293 | 0.334 | 0.387 |
| **PTN** | MSE | 0.066 | 0.112 | 0.142 | 0.157 | 0.083 | 0.1969 | 0.1982 | 0.190 |

**Table 4.2. View point reconstruction.** Quantitative evaluation of scene reconstruction for unseen views by extrapolating the view angle from 0°(original) to 80°. We observe that our method does better when compare to view-centric reconstruction of PTN Yan et al. [2016] Note that PTN model is tuned to perform better for silhouette based single object reconstruction with just plane background. HD is the Hausdorf distance, CD denotes Chamfer Distance and MSE is mean-squared error

| Labeled Samples | CNN | Siamese CNN | Human Evaluation | Persp. Transf. Nets Yan et al. [2016] | Rezende et al. Rezende et al. [2016] | Pix2Shape **(Ours)** |
|---|---|---|---|---|---|---|
| 0 | 0.3385 | 0.3698 | 0.7329 ± 0.148 | 0.5344 | 0.5202 | **0.5519 ± 0.013** |
| 200 | 0.3350 | 0.3610 | - | 0.6011 | 0.6155 | **0.6312 ± 0.031** |
| 1,000 | 0.3392 | 0.3701 | - | 0.6645 | 0.7001 | **0.7012 ± 0.021** |

**Table 4.3. 3D-IQTT results.** Evaluation on the 3D-IQTT task of our model, two CNN-based baselines, Perspective Transformer Nets and [Rezende et al., 2016]. This table also includes comparison with human performance. Although Pix2Shape performs well compared to other baselines, it is still lagging behind the human level by a good margin.

$\boldsymbol{z_1}_{scene}$ and $\boldsymbol{z_2}_{scene}$ and decoded the intermediary points into their corresponding images using a fixed camera pose. Figure 4.9 shows this for two different settings.

### 4.5.3. 3D-IQ Test Task

We trained our model using the aforementioned semi-supervised training described in Section 4.3.3.4 on the 3D-IQTT task. We compared our model to different baselines listed below and with human performance.
Human. We created an online test where 40 random graduate students from our lab answered 20 randomly selected questions from the test set (similar to Figure 4.2). No student had seen these images before. More details can be found in Appendix A.9.
CNN.. The first baseline is composed of four ResNet-50 modules [He et al., 2016] with shared weights followed by three fully-connected layers and a softmax output for the class label (answer 1 to 3). We trained this CNN only on the labeled samples. The architecture is depicted in the appendix, Figure A.3.2.
Siamese Network. Our second baseline is a Siamese CNN with a similar architecture as the previous one but with the fully-connected layers removed. Instead of the supervised loss provided in the form of correct answers, it was trained with contrastive loss [Koch et al., 2015]. This loss reduces the feature distance between the reference and correct answer and maximizes the feature distance between the reference and incorrect answers.
Perspective Transformer Nets. As our third baseline, we used the open source implementation of the Perspective Transformer Nets [Yan et al., 2016] to solve the IQTT task using the learnt latent code.
Rezende et al. [2016]: Since there is no open source code available for this work, we implemented our own interpretation of this work. We were able to reproduce the results from their paper (see appendix A.10 before attempting to use it as baseline for our model.

A more detailed description of the networks and contrastive loss function can be found in the supplementary material A.3.

Table 4.3 shows 3D-IQTT results for our method and baselines. The CNN-based baselines were not able to infer the underlying 3D structure of the data and their results are only slightly better than random guess. The poor performance of the Siamese CNN might be in part because the contrastive loss rewards similarities in pixel space and has no notion of 3D similarity. However, Pix2Shape achieved significantly better accuracy by leveraging the learned 3D knowledge of objects. Our method also outperformed the other 2 baseline approaches, but with a smaller margin.

### 4.5.4. Analyzing the Loss Functions

In this section, we do an ablation study of the different loss functions used to train our model. Our final objective is a combination of bi-directional adversarial loss $\mathcal{L}_{ALI}$ and a reconstruction loss $\mathcal{L}_{recon}$. For the 3D-IQTT task we augmented the above losses with a mutual information based objective $I_{\Theta}(z_s,z_v)$ to make sure that different parts of the latent

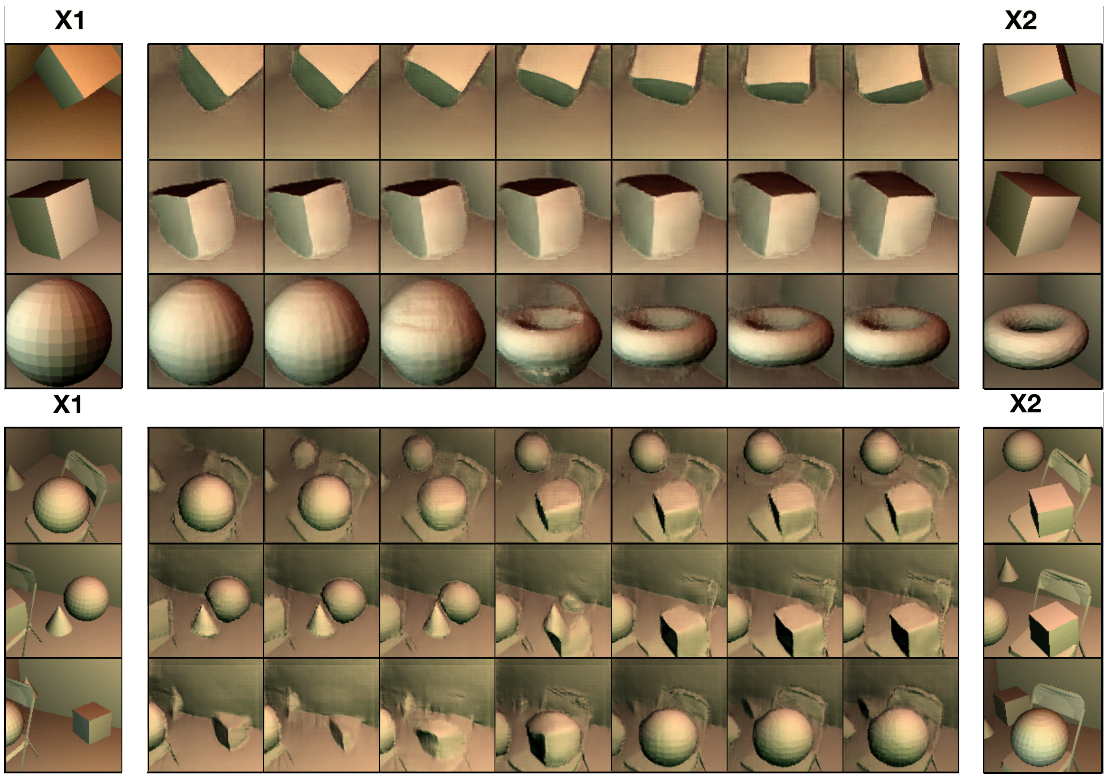


**Fig. 4.9. Manifold exploration.** Exploration of the learned manifold of 3D representations. Generated interpolations (middle columns) between two images $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ (first and last columns).

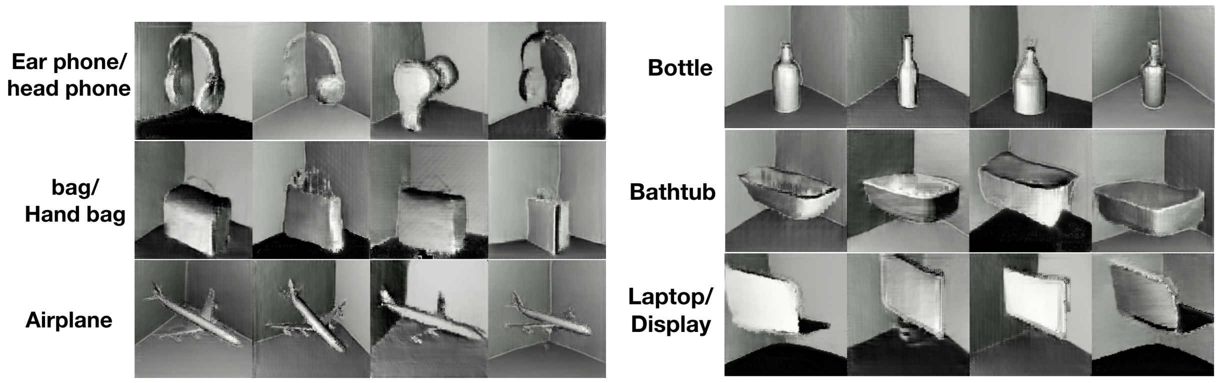


**Fig. 4.10. Conditional scene generation.** Class-conditionally generated samples for ShapeNet dataset. These images are not part of the training data.

code encode distinctive pieces of information present in a scene. This allows us to disentangle view point and geometry. Table 4.4 shows our results for both the reconstruction task on the ShapeNet scenes dataset and the 3D-IQTT task when considering, i) only adversarial loss ($\mathcal{L}_{ALI}$); ii) only reconstruction loss($\mathcal{L}_{recon}$); iii) adversarial and reconstruction but not mutual info ($\mathcal{L}_{ALI}$); and ($\mathcal{L}_{recon}$) (note that this does not effect reconstruction task); and, iv) all three ($\mathcal{L}_{ALI}$, $\mathcal{L}_{recon}$ and $I_{\Theta}(z_s, z_v)$).

| Adversarial Loss $\mathcal{L}_{ALI}$ | Reconstruction Loss $\mathcal{L}_{ALI}$ | Mutual Info Loss | Reconstruction on ShapeNet scenes (Chamfer dist) | | | | 3D-IQTT No Labels | 3D-IQTT 1000 labels |
|---|---|---|---|---|---|---|---|---|
| | | | 0° | 35° | 55° | 80° | | |
| ✓ | | | 1.927 | 2.341 | 2.350 | 2.281 | 0.4024 | 0.4454 |
| | ✓ | | 0.1066 | 0.1935 | 0.201 | 0.1925 | 0.3321 | 0.3789 |
| ✓ | ✓ | | 0.1120 | 0.1522 | 0.1483 | 0.1433 | 0.4224 | 0.4921 |
| | ✓ | ✓ | - | - | - | - | 0.4943 | 0.6272 |
| ✓ | ✓ | ✓ | - | - | - | - | 0.5526 | 0.7020 |

**Table 4.4. Loss analysis**. Ablation study of the different losses used to train our model on both reconstruction task and 3D-IQTT task. We evaluate the contribution of each of the objectives in the table. Having the adversarial loss alone negatively affects the reconstructions considerably because, although output images look realistic, they do not match input images very well. Although the reconstruction loss alone does better for reconstruction without view-extrapolation, the performance degrades as we extrapolate to novel views. Note that the reconstruction loss only fixes the indivisibility issues in ALI based models, but considerably affects its generalization ability [Li et al., 2017a]

.

We observe each loss term improves the performance of the model on both the tasks. Using adversarial loss alone is not enough to faithfully reconstruct the surfels. On the other hand we observe that having the reconstruction loss alone affects the performance of the model while extrapolating the shape from unseen views (e.g., view angle 35° to 80°). However, this scenario yields better performance when reconstructing from the given input view point, i.e., 0°. We also notice that having a reconstruction loss alone affects the quality of the samples generated. We observe that the adversarial loss ($\mathcal{L}_{ALI}$) plays a major role in obtaining detailed and high quality samples. For the 3D-IQTT task, the role of ($\mathcal{L}_{ALI}$) is more evident. ($\mathcal{L}_{ALI}$) encourages the latent code to learn meaningful representations by constraining the model to match the joint distributions. Results also indicate clearly that skipping mutual information loss degrades the performance of the model on 3D-IQTT task. This is expected because of the mix-up of view information with geometrical information in the latent representation.

## 4.6. Conclusion

In this paper we propose a generative approach to learn 3D structural properties from single-view images in an unsupervised and implicit fashion. Our model receives an image of a scene with uniform material as input, estimates the depth of the scene points and then reconstructs the input image through a differentiable renderer. We also provide quantitative evidence that support our argument by introducing a novel IQ Test Task in a semi-supervised setup. We hope that this evaluation metric will be used as a standard benchmark to measure the 3D understanding capability of models across different 3D representations. The main drawback of our current model is that it requires the knowledge of lighting and material properties. Future work will focus on tackling the more ambitious setting of learning complex materials and texture along with modelling the lighting properties of the scene.

# Chapter 5

# Prologue to Article 2

## 5.1. Article Details

**Multi-label Iterated Learning for Image Classification with Label Ambiguity**. Sai Rajeswar*, Pau Rodriguez*, Soumye Singhal, David Vazquez, Aaron Courville. Published and presented at International Conference on Computer Vision and Pattern Recognition (CVPR) 2022. (First two authors contributed equally)

*Personal Contribution* The idea was conceived after a joint brainstorming session involving the author and Aaron Courville. While the author focused on implementing the iterated learning pipeline and label-ambiguity studies, Pau Rodriguez focused on self-supervised learning and OOD generalization components. Critical aspects of trying suitable multi-label objectives and making ImageNet experiments scale efficiently with the compute were pair programmed by the author and Pau Rodriguez. Aaron Courville advised on the project discussing results, suggesting experiments to run, and writing parts of the paper. All authors participated in weekly discussions and helped with preparing the manuscript.

## 5.2. Context

The project highlights and discusses the weakly labeled nature of curated datasets like ImageNet. It is known that the generalization performance of deep models is affected by the combination of objects that are not present in the datasets. The project took shape by initially noting that the existing datasets are themselves biased toward certain combinations of objects that are more "organically natural" (*e.g.* an image with a "cow" sitting on a "beach" is a rare occurrence compared to sitting on a "grass"y landscape). We agreed that these spurious correlations can be addressed if we take into account the compositional aspect of natural scenes in these datasets while training the models.

## 5.3. Research Impact

The work remains one of the best performing weakly-supervised classification methods on datasets with label-ambiguity [Li et al., 2017b]. To the best of our knowledge, this is the first application of the iterated learning framework in the visual domain. This can be potentially extended to other perceptual problems where it is crucial to expand on the knowledge of privileged information with only weak supervision. Similar to our line of work, Iscen et al. [2022] presents an alternate way to deal with learning from noisy labels that leverage similarities between training examples in feature space.

**Note**: This paper is presented as-is, with minor cosmetic changes to adhere to the Universite de Montreal thesis template.

# Chapter 6

# Article 2: Multi-label Iterated Learning for Image Classification with Label Ambiguity

## Abstract

Transfer learning from large-scale pre-trained models has become essential for many computer vision tasks. Recent studies have shown that datasets like ImageNet are weakly labeled since images with multiple object classes present are assigned a single label. This ambiguity biases models towards a single prediction, which could result in the suppression of classes that tend to co-occur in the data. Inspired by language emergence literature, we propose multi-label iterated learning (MILe) to incorporate the inductive biases of multi-label learning from single labels using the framework of iterated learning. MILe is a simple yet effective procedure that builds a multi-label description of the image by propagating binary predictions through successive generations of teacher and student networks with a learning bottleneck. Experiments show that our approach exhibits systematic benefits on ImageNet accuracy as well as ReaL F1 score, which indicates that MILe deals better with label ambiguity than the standard training procedure, even when fine-tuning from self-supervised weights. We also show that MILe is effective reducing label noise, achieving state-of-the-art performance on real-world large-scale noisy data such as WebVision. Furthermore, MILe improves performance in class incremental settings such as IIRC and it is robust to distribution shifts.

## 6.1. Introduction

Large-scale datasets with human-annotated labels have been central to the development of modern state-of-the-art neural network-based artificial perception systems [Krizhevsky et al., 2012, He et al., 2016, 2017]. Improved performance on ImageNet [Deng et al., 2009] has led to remarkable progress in tasks and domains that leverage ImageNet pretraining [Carreira

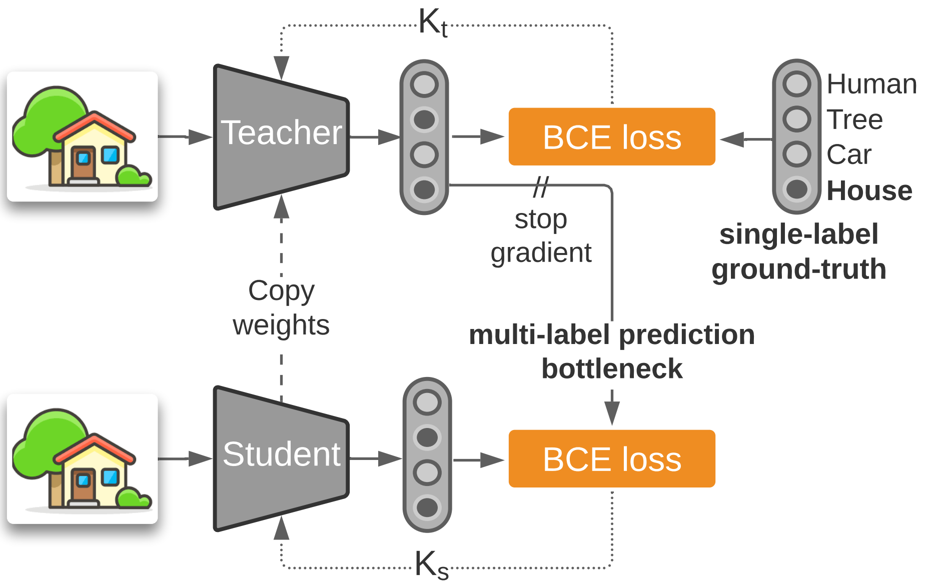


**Fig. 6.1. Multi-label Iterated Learning (MILe)** builds a multi-label representation of the images from singly-labeled ground-truth. In this example, a model produces multi-label binary predictions for the next generation, obtaining *Car* and *House* for an image weakly labeled with *House*.

and Zisserman, 2017, Long et al., 2015, Zhao et al., 2017]. However, these weakly-annotated datasets and models tend to project a rich, multi-label reality into a paradigm that envisions one and only one label per image. This form of simplification often hinders model performance by asking models to predict a single label, when trained on real-world images that contain multiple objects.

Given the importance of the problem, there is growing recognition of single-labeled datasets as a form of weak supervision and an increasing interest in evaluating the limits of these singly-labeled benchmarks. A series of recent studies [Stock and Cisse, 2018, Tsipras et al., 2020b, Shankar et al., 2020, Beyer et al., 2020, Yun et al., 2021] highlight the problem of label ambiguity in ImageNet. In order to obtain a better estimate of model performance, Beyer et al. [2020] and Shankar et al. [2020] introduced multi-label evaluation sets. They identified softmax cross-entropy training as one of the main reasons for low multi-label performance since it promotes label exclusiveness. They also showed that replacing the softmax with sigmoid activations and casting the output as a set of binary classifiers results in better multi-label validation performance. Several other studies have explored ways to overcome the shortcomings in existing validation procedures by improving the pipelines for gathering labels Barbu et al. [2019], Tsipras et al. [2020a], Recht et al. [2019a].

In order to obtain a more complete description of images from weakly-supervised or semi-supervised data, a number of methods leverage a noisy signal such as pseudo-labels [Yun et al., 2021] or textual descriptions crawled from the web [Radford et al., 2021]. In this work, we observe that the process of building a rich representation of data from a noisy source shares some properties with the process of language emergence studied in the cognitive science literature. In particular, Kirby [2001] proposed that structured language emerged from an inter-generational *iterated learning* process [Kirby, 2001, 2002, Kirby et al., 2014]. According to the theory, a compositional syntax emerges when agents learn by imitation from previous generations in the presence of a learning bottleneck. This bottleneck forces noisy fragments of the language to be forgotten when transmitted to new generations. Conversely, those fragments that can be reused and composed to enrich the language tend to be passed to subsequent generations. We show that the same procedure can be applied to settings that leverage a weak or noisy supervisory signal such as [Yun et al., 2021, Radford et al., 2021] to build a richer description of images while reducing the noise.

In this work, we propose multi-label iterated learning (MILe) to learn to predict rich multi-label representations from weakly supervised (single-labeled) training data. We do so by introducing two different learning bottlenecks. First, we replace the standard convolutional neural network output softmax with a hard multi-label binary prediction. Second, we transmit these binary predictions through successive model generations, with a limited training iterations between each generation.

In our experiments, we demonstrate that MILe alleviates the label ambiguity problem by improving the F1 score of supervised and self-supervised models on the ImageNet ReaL [Beyer et al., 2020] multi-label validation set. In addition, experiments on WebVision [Li et al., 2017b] show that iterated learning increases robustness to label noise and spurious correlations. Finally, we show that our approach can help in continual learning scenarios such as IIRC Abdelsalam et al. [2021] where newly introduced labels co-occur with known labels. Our contributions are:

- We propose MILe, a multi-label iterated learning algorithm for image classification that builds a rich multi-label representation of data from weak single labels.
- We show that models trained with MILe are more robust to noise and perform better on ImageNet, ImageNet-ReaL, WebVision, and multiple setups such as supervised learning (Section 6.3.1), self-supervised fine-tuning and semi-supervised learning (Section 6.3.2), continual learning (Supplementary 2), and domain generalization (Supplementary 5).
- We provide insights on the predictions made by models trained with iterated learning (Section 6.3.3).

## 6.2. MILe: Multi-Label Iterated Learning

We propose MILe to counter the problem of label ambiguity in singly-labeled datasets. We delineate the details of our approach to perForm multi-label classification from weak singly-labeled ground truth.

### 6.2.1. EnForcing Multi-label Prediction.

Singly-labeled datasets such as ImageNet usually represent their labels as one-hot vectors (all dimensions are zero except one). Training on these one-hot vectors Forces models to predict a single class, even in the presence of other classes. Forcing models to predict a single class exposes them to biases in the image labeling process such as the preference For centered objects. Besides, constraining the model to output a single label per image limits the capability of perceptual models to capture all the content of the image accurately. In order to solve this problem, we propose to relax the model's output predictions from singly-labeled softmax prediction to multi-label binary prediction with sigmoids. Thus, we treat the singly-labeled classification problem as a set of independent binary classification problems. Since the ground-truth labels are still represented as one-hot vectors and training on them would still result in singly-labeled predictions, we propose an iterated learning procedure to bootstrap a multi-label pseudo ground truth.
Multi-label Iterated Learning. Our learning procedure is composed of two phases. In the first phase, a *teacher* model interacts with the single-labeled data to improve its predictions. The interaction is limited to a few iterations to prevent the binary classification model from overfitting to one-hot vectors. In the second phase, we leverage the acquired knowledge to train a different model, the *student*, on the multi-label predictions of the teacher. This yields a better initialization of the model For further iterations as we repeat this two-phased learning multiple times (see Alg. 2).

Specifically, we consider two parametric models, the teacher $f(.;\theta^T_\tau)$ and the student $f(.;\theta^S_\tau)$. Parameters of the teacher $\theta^T_\tau$ are initialized using the student parameters $\theta^S_\tau$ at iteration $\tau$. First, we train the teacher For $k_t$ learning steps on the labeled images from the dataset, obtaining $f(.;\theta^T_{\tau+1})$. This constitutes the interaction phase of an iteration. We then move to the imitation phase, where we train the student to fit the teacher model For $k_s$ steps, obtaining $f(.;\theta^S_{\tau+1})$. This is done by training the student on the pseudo labels generated by the teacher on the data. Finally, we instantiate a new teacher by duplicating the parameters of this new student and iterate the process until convergence. In addition to yielding a smooth transition during the imitation phase, this procedure ensures that each iteration yields an improvement over the previous one (unless it is already optimal). Note that in the supervised learning regime we do not pseudo label any unlabeled data. In Sec. 6.3.2

we provide additional experiments showing that MILe can leverage unlabeled data in the semi-supervised learning regime.

Both the teacher and the student are trained on the same dataset $\mathcal{D}$ composed of input-label pairs $\{\mathcal{X}, \mathcal{Y}\} \in \mathcal{D}$. We train the teacher to maximize the likelihood $p(\hat{y} = y|x, \theta) = \sigma(f(x, \theta))$, where $\hat{y}$ is the label predicted by the model, $y \in \mathcal{Y}$ is the true label, and $\sigma$ is a normalization function such as the sigmoid. In order to alleviate the problem of label ambiguity, we consider $\mathcal{Y}$ a multi-label binary vector in $\mathbb{Z}_2^C$ where $C$ is the number of classes and optimize the binary cross-entropy loss:

$$\mathcal{L}_{BCE} = -\frac{1}{B}\sum_{i=1}^{B}\frac{1}{C}\sum_{j=1}^{C} y_{i,j} \cdot log(\hat{y}_{i,j}) + (1 - y_{i,j}) \cdot log(1 - \hat{y}_{i,j}), \tag{6.2.1}$$

where $B$ is the number of samples in a batch when using batched stochastic gradient descent. We show in our experiments that iterated learning along with multi-label objective provides a strong inductive bias For modeling the effects of label ambiguity. Note that optimizing the binary cross-entropy on one-hot labels would not solve the label ambiguity problem. Thus, during each cycle, we train the teacher For a few iterations in order to prevent it from overfitting the one-hot ground truth. During student training, we threshold the teacher's output sigmoid activations to obtain multi-label pseudo ground-truth vectors $\tilde{y} = f(x,\theta^T) > \rho$. The threshold $\rho$ is 0.25 unless otherwise Stated.

### 6.2.2. The MILe Learning Bottleneck.

EnForcing the imitation phase with some Form of a learning budget is an essential component of the iterated learning framework [Kirby, 2001]. This bottleneck regularizes the student model not to be amenable to the specific irregularities in the data. Kirby [2001] argue that such a bottleneck enForces innate constraints on language acquisition. We believe that incorporating such a mechanism into the prediction models could prevent them from overfitting label noise [Liu et al., 2020], improving the quality of pseudo labels. There are two common ways to impose a learning bottleneck. One way is to allow a newly initialized student to only obtain the knowledge from a limited number of data instances generated by the teacher [Kirby, 2001, Liu et al., 2021]. Another is by limiting the number of student learning updates while imitating the teacher [Lu et al., 2020a]. In our setting, we find it helpful to enForce the bottleneck via the number of learning updates.

As illustrated in Fig. 6.1 and Alg. 2, we iteratively refine a teacher network that is trained with the original labels and a student network that is trained with labels produced by the teacher. In order to prevent the student from overfitting the teacher, we restrict the amount of training updates [Lu et al., 2020a] For each of the modules. Formally, let $N$ be the size of the dataset, $k_t$ be the number of training iterations of the teacher, and $k_s$ the number of student iterations. In general, we set $k_t << N$ to prevent the teacher from overfitting one-hot

**Algorithm 2** MILe

```
Require: Initialize Student network $\boldsymbol{\theta}^S_\tau$, $\tau = 0$.        ▷ Prepare Iterated Learning
 1: repeat
 2:     Copy $\boldsymbol{\theta}^S_\tau$ to $\boldsymbol{\theta}^T_{\tau+1}$        ▷ Initialize Teacher
 3:     for $i = 1$ to $k_t$ do
 4:         Sample a batch $(\boldsymbol{x_i}, \boldsymbol{y_i}) \in \mathcal{D}_{train}$
 5:         $\hat{\boldsymbol{y}}_{\boldsymbol{i}} = f_{\boldsymbol{\theta}^T}(\boldsymbol{x_i})$
 6:         $\boldsymbol{\theta}^T_{\tau+1} \leftarrow \boldsymbol{\theta}^T_{\tau+1} + \alpha\nabla\mathcal{L}^{BCE}(\boldsymbol{\theta}^T_{\tau+1}; \boldsymbol{y_i}, \hat{\boldsymbol{y}}_{\boldsymbol{i}})$        ▷ Update $\boldsymbol{\theta}^T$ to minimize L
 7:     end for        ▷ Finish Interactive Learning
 8:     for $i = 1$ to $k_s$ do
 9:         Sample a batch $(\boldsymbol{x_i}, \boldsymbol{y_i}) \in \mathcal{D}_{train}$
10:         $\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \sigma(f_{\boldsymbol{\theta}^T_{\tau+1}}(\boldsymbol{x_i})) > \rho$        ▷ Generate Pseudo Labels
11:         $\tilde{\boldsymbol{y}}_{\boldsymbol{i}} = f_{\boldsymbol{\theta}^S}(\boldsymbol{x_i})$
12:         $\boldsymbol{\theta}^S_\tau \leftarrow \boldsymbol{\theta}^S_\tau + \alpha\nabla\mathcal{L}^{BCE}(\boldsymbol{\theta}^S_\tau; \tilde{\boldsymbol{y}}_{\boldsymbol{i}}, \hat{\boldsymbol{y}}_{\boldsymbol{i}})$        ▷ Update $\boldsymbol{\theta}^S$ to minimize L
13:     end for        ▷ Finish Imitation
14:     Copy $\boldsymbol{\theta}^S_\tau$ to $\boldsymbol{\theta}^S_{\tau+1}$
15:     $\tau \leftarrow \tau + 1$
16: until Convergence or maximum $\tau$ reached
```

labels and $k_s <= k_t$ to prevent the student from overfitting the teacher. In other words, each of our iterations is composed of two finite loops of (a) model improvement (teacher learning) and (b) model imitation (student learning).
Computational Cost. We train MILe For the same total number of epochs as standard supervised classification models. Thus, the total number of *backward* passes through the model (counting both the teacher and the student) is the same as the standard supervised training. Thus, the only additional computational cost comes from producing pseudo-labels with the teacher model. Moreover, the pseudo-labeling only happens once per teacher-student cycle and the network is in inference mode. Assuming $k_s + k_t = 10K$ (see Figure 6.3) and a batch size of 256, this inference pass only happens every 2.1 epochs For the ImageNet. Thus, the computational impact of MILe only constitutes a small fraction of the overall computational cost of training a neural network on the ImageNet. This computational cost could be easily compensated by skipping validation on alternate epochs or by validating in a different parallel process.

## 6.3. Experiments

We provide experiments showing the effects of iterated learning in multiple setups. In Sec. 6.3.1, we study the robustness to label ambiguity and noise on ImageNet Real and WebVision. In Sec. B.5, we explore the benefits of iterated learning for domain generalization. In Sec. 6.3.2, we study the effect of MILe on models pre-trained with self-supervised objectives. Finally, in Sec. 6.3.3, we provide ablation experiments on the different hyperparameters as well as a more challenging synthetic setup with greater label ambiguity. Additional experiments

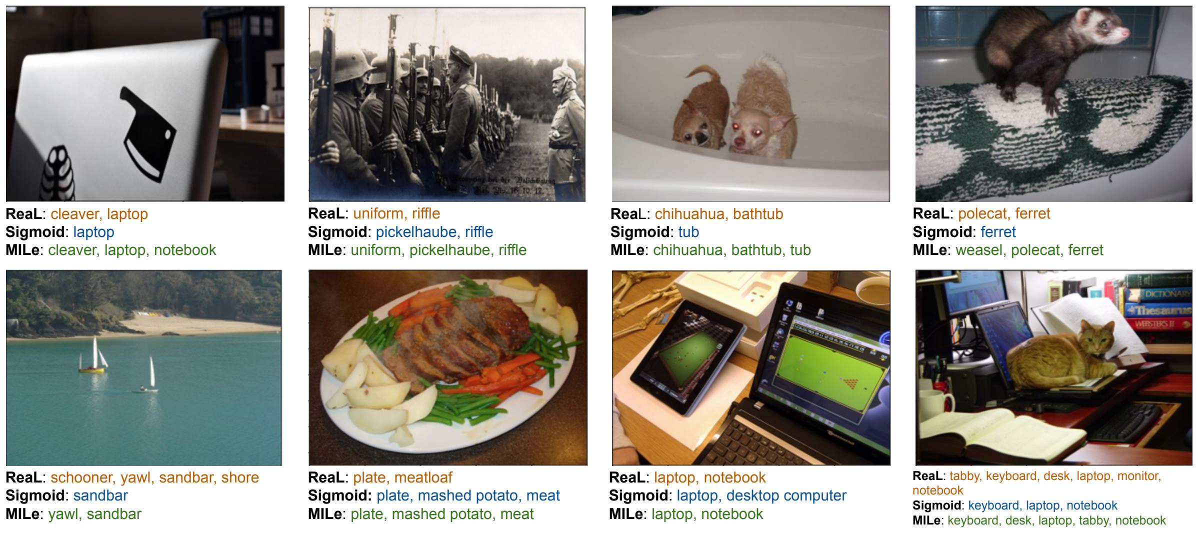

ReaL: cleaver, laptop
Sigmoid: laptop
MILe: cleaver, laptop, notebook
ReaL: uniform, riffle
Sigmoid: pickelhaube, riffle
MILe: uniform, pickelhaube, riffle
ReaL: chihuahua, bathtub
Sigmoid: tub
MILe: chihuahua, bathtub, tub
ReaL: polecat, ferret
Sigmoid: ferret
MILe: weasel, polecat, ferret
ReaL: schooner, yawl, sandbar, shore
Sigmoid: sandbar
MILe: yawl, sandbar
ReaL: plate, meatloaf
Sigmoid: plate, mashed potato, meat
MILe: plate, mashed potato, meat
ReaL: laptop, notebook
Sigmoid: laptop, desktop computer
MILe: laptop, notebook
ReaL: tabby, keyboard, desk, laptop, monitor, notebook
Sigmoid: keyboard, laptop, notebook
MILe: keyboard, desk, laptop, tabby, notebook

**Fig. 6.2. Qualitative results.** ReaL: original labels. Sigmoid: ResNet-50 with sigmoid output activations. MILe: multi-label iterated learning (ours).

in the Supplementary Material include a comparison with noisy student, multi-label learning on CelebA, and continual learning on IIRC.

### 6.3.1. Label Ambiguity and Noise

**Datasets:** We train our models on the standard ImageNet image classification benchmark [Russakovsky et al., 2015], which is known to contain ambiguous labels [Beyer et al., 2020]. Therefore, in addition to the validation set performance, we also report the performance on ReaL [Beyer et al., 2020], an additional set of multi-labels for the ImageNet validation set gathered using a crowd-sourcing platform. ReaL contains a total of 57,553 labels for 46,837 images. We report results when using fractions of the total amount of training examples (i.e., 1%, 5%, 10%, 100%). To test the robustness of our method to label noise, we provide results on WebVision [Li et al., 2017b], which contains more than 2.4 million images crawled from the Flickr website and Google Images search. The same 1,000 concepts as the ImageNet ILSVRC 2012 dataset are used for querying images. It is worth noting that many ImageNet (ReaL) samples contain a single object and a single label. In Sec. 6.3.3, we explore the limits of MILe on a synthetic dataset. In addition, we provide results on CelebA [Liu et al., 2015] in the supplementary material.

**Baselines:** We train a ResNet-18 and a ResNet-50 [He et al., 2016] model. Note that we favored vanilla ResNets over more advanced architectures and training procedures in order to focus on the advantages of MILe, rather than showing state-of-the-art results. We compare three different methods. (i) *Softmax*: standard softmax cross-entropy loss used to train the original ResNet backbone [He et al., 2016]. (ii) *Sigmoid*: we substitute the cross-entropy loss for a binary cross-entropy (BCE) loss. (iii) *MILe*: the proposed method as described

| | ImageNet fraction: | 1% | 5% | 10% | 100% | 1% | 5% | 10% | 100% |
|---|---|---|---|---|---|---|---|---|---|
| Metric | Method | ResNet-50 | | | | ResNet-18 | | | |
| Accuracy | Softmax | 6.32 | 36.71 | 53.50 | 76.33 | 6.61 | 31.5 | 48.82 | 70.41 |
| | ELR [Liu et al., 2020] | 7.91 | 38.88 | 56.15 | 76.75 | 6.93 | 32.95 | 49.95 | 70.83 |
| | Sigmoid | 6.70 | 36.9 | 55.01 | 76.35 | 6.88 | 31.10 | 49.14 | 70.46 |
| | MILe (ours) | **9.10** | **42.52** | **57.29** | **77.12** | **8.20** | **36.20** | **51.31** | **71.12** |
| ReaL-Acc | Softmax | 7.19 | 42.55 | 60.21 | 82.76 | 8.80 | 35.88 | 55.11 | 77.77 |
| | ELR [Liu et al., 2020] | 8.78 | 44.24 | 63.13 | 83.07 | 8.92 | 38.08 | 56.13 | 78.85 |
| | Sigmoid | 8.38 | 46.04 | 62.96 | 83.22 | 9.04 | 37.66 | 57.52 | 81.01 |
| | MILe (ours) | **11.50** | **48.36** | **65.42** | **83.75** | **9.18** | **41.65** | **58.57** | **81.52** |
| ReaL-F1 | Softmax | 6.77 | 40.51 | 57.33 | 78.5 | 8.28 | 34.20 | 52.51 | 73.83 |
| | ELR [Liu et al., 2020] | 7.83 | 42.45 | 58.52 | 78.5 | 8.41 | 35.52 | 53.22 | 73.41 |
| | Sigmoid | 7.17 | 41.11 | 58.46 | 78.61 | 8.39 | 33.56 | 52.12 | 73.85 |
| | MILe (ours) | **10.76** | **45.02** | **62.11** | **79.89** | **8.55** | **38.49** | **53.80** | **74.48** |
| Label Coverage | Softmax | 1.00 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| | ELR [Liu et al., 2020] | 1.00 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| | Sigmoid | 1.09 | 1.11 | 1.10 | 1.11 | 1.07 | 1.10 | 1.15 | 1.15 |
| | MILe (ours) | 1.05 | 1.08 | 1.09 | 1.16 | 1.06 | 1.07 | 1.12 | 1.17 |

**Table 6.1. ImageNet results.** The first row displays the fraction of the ImageNet data used to train the models. Softmax: Vanilla ResNet with softmax loss. Sigmoid: Vanilla ResNet trained for multi-label binary classification with single labels. ELR: early-learning regularization [Liu et al., 2020]. MILe: multi-label iterated learning. Label coverage refers to the fraction of additional labels predicted by each model. All the models are trained for 100 epochs.

in Sec. 4.3. For WebVision experiments, we also train an additional ResNet-50-D [He et al., 2019b] backbone following more recent methodologies [Yang et al., 2020].

**Metrics:** We report accuracy on the original [Russakovsky et al., 2015] and the ReaL [Beyer et al., 2020] ImageNet validation set. ReaL is a multi-label dataset, so we calculate the accuracy as described by Beyer et al. [2020]. Namely, we consider a top-1 prediction correct if it coincides with any of the ground-truth labels, i.e. ReaL-Acc $= \frac{1}{N}\sum_{i=1}^{N} |\hat{y}_i \cap Y_i| > 0$, where $\hat{y}_i$ is the predicted label for the $i$th sample, $Y_i$ is the set of ReaL labels, and $|.|$ counts the the number of elements in a set. Additionally, we report the F1-score, which represents the proportion of correct predicted labels to the total number of actual and predicted labels, averaged across all examples: ReaL-F1 $= \frac{1}{N}\sum_{i=1}^{N} \frac{2 \cdot TP_i}{2 \cdot TP_i + FP_i + FN_i}$, where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives. Finally, we report the label coverage, which indicates the total fraction of labels per sample predicted by the multi-label classifier. A number 1.15 indicates an additional 15% of labels was predicted.

**ImageNet results.** We report the results in Table 6.1. MILe surpasses baseline methods on all metrics and all fractions of training data. With Sigmoid, we observe a substantial improvement on ReaL-Acc of $\sim 2\%$ and $\sim 4\%$ for ResNet-18 and ResNet-50 respectively.

| Method | Architecture | WebVision | | ImageNet | |
|---|---|---|---|---|---|
| | | Top-1 | Top-5 | Top-1 | Top-5 |
| CrossEntropy [Tu et al., 2020] | ResNet-50 | 66.4 | 83.4 | 57.7 | 78.4 |
| MentorNet [Jiang et al., 2018] | InceptionRes-V2 | 70.8 | 88.0 | 62.5 | 83.0 |
| CurriculumNet [Guo et al., 2018] | Inception-V2 | 72.1 | 89.1 | 64.8 | 84.9 |
| CleanNet [Lee et al., 2018] | ResNet-50 | 70.3 | 87.8 | 63.4 | 84.6 |
| CurriculumNet [Guo et al., 2018, Tu et al., 2020] | ResNet-50 | 70.7 | 88.6 | 62.7 | 83.4 |
| SOM [Tu et al., 2020] | ResNet-50 | 72.2 | 89.5 | 65.0 | 85.1 |
| Distill [Zhang et al., 2020] | ResNet-50 | - | - | 65.8 | **85.8** |
| MoPro (dec.) [Li et al., 2021] | ResNet-50 | 72.4 | 89.0 | 65.7 | 85.1 |
| Multimodal [Shah et al., 2019] | Inception-V3 | 73.15 | 89.73 | - | - |
| Sigmoid | ResNet-50 | 72.1 | 89.5 | 65.4 | 85.0 |
| MILe (ours) | ResNet-50 | **75.2** | **90.3** | **67.1** | 85.6 |
| Initial Vanilla Model | ResNet-50-D | 75.08 | 89.22 | 67.23 | 84.09 |
| SCC [Yang et al., 2020] | ResNet-50-D | 75.36 | 89.38 | 67.93 | 84.77 |
| SCC+GBA [Yang et al., 2020] | ResNet-50-D | 75.69 | 89.42 | 68.35 | 85.24 |
| MILe (ours) | ResNet-50-D | **76.5** | **90.9** | **68.7** | **86.4** |

**Table 6.2. WebVision results.** Methods are trained on Webvision-1000 and validated both on WebVision and ImageNet. MoPro (decoupled) is pre-trained on the same set as our method. CleanNet [Lee et al., 2018] and Distill [Zhang et al., 2020] require data with clean annotations. dec: refers to "decoupled".

This is in agreement with the results reported by Beyer et al. [2020]. Incorporating iterative learning results in an extra $\sim 1\%$ performance improvement when using all the training data and up to 5% of ReaL-F1 when using a smaller fraction of the data. Interestingly, we find that using smaller fractions of data reduces the label coverage. We hypothesize that using a smaller fraction of the data leads to memorization and overfitting for the Softmax method and Sigmoid, which results in more confident predictions on a single class. Additional results focused on ReaL label recovery can be found in the supplementary material.

We report qualitative results in Fig. 6.2. As it can be seen, MILe produces more complete descriptions of the image, sometimes capturing labels that were not included in the ReaL ground truth. For instance, our method was able to detect a pickelhaube (pointy hat) that was not labeled in the ground truth.

**WebVision results.** We report results in Table 6.2 and put them in context with other state of the art. We adopt the same class rebalancing strategy as [Li et al., 2021]. For all setups, we observe that MILe attains the best performance, up to 2 points better than methods using better architectures such as Inception-V3 [Shah et al., 2019]. We also validate the WebVision-trained model on the ImageNet validation set, outperforming the previous state of the art and keeping results consistent with the WebVision validation set. These results suggest that the iterated learning bottleneck acts as a regularizer that prevents the model from learning noisy labels which are more difficult to fit. This hypothesis is in agreement with

| Method | ImageNet Validation | | | ImageNet ReaL-F1 | | |
|---|---|---|---|---|---|---|
| | 1% | 10% | 100% | 1% | 10% | 100% |
| SimCLR [Chen et al., 2020b] | 48.3 | 65.6 | 76.25 | 51.54 | 69.16 | 76.91 |
| BYOL [Grill et al., 2020] | 53.2 | 68.8 | 77.2 | 54.32 | 70.81 | 78.85 |
| SwAV [Caron et al., 2020] | 53.9 | 70.2 | 77.74 | 55.79 | 71.22 | 79.18 |
| MoCo-v2 [Chen et al., 2020d] | 51.72 | 66.5 | 77.12 | 53.34 | 70.75 | 79.04 |
| MILe (Ours) + [Chen et al., 2020d] | **52.62** | **67.4** | **77.38** | **56.08** | **71.48** | **80.03** |
| SimCLR-v2-sk0 [Chen et al., 2020c] | 58.18 | 68.9 | 76.3 | 57.25 | 70.11 | 78.83 |
| MILe (Ours) + [Chen et al., 2020c] (sk0) | **61.85** | **70.5** | **77.29** | **60.49** | **72.76** | **79.38** |
| SimCLR-v2-sk1 [Chen et al., 2020c] | 64.7 | 72.4 | 78.7 | 62.77 | 74.21 | 79.43 |
| MILe (Ours) + [Chen et al., 2020c] (sk1) | **69.4** | **74.7** | **79.5** | **65.04** | **76.40** | **81.53** |

**Table 6.3. Self-supervised finetuning.** The second row displays the fraction of ImageNet training data used for fine-tuning. Accuracy of top-1 predictions are used for reporting the numbers.

| Method | Teacher | Label fraction | |
|---|---|---|---|
| | | 1% | 10% |
| Distilled [Chen et al., 2020c] | R50 (2×+SK) | 69.0 | 75.1 |
| Self-distilled [Chen et al., 2020c] | R50 (1x+SK) | 70.15 | 74.43 |
| MILe (ours) | R50 (1x+SK) | **73.08** | **75.3** |

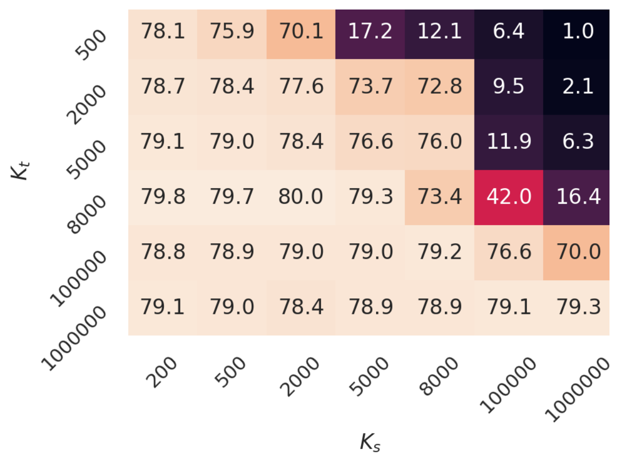
**Table 6.4. Self- semi-supervised learning.** ImageNet top-1 accuracy for ResNet-50 (R50) distilled from a SimCLR [Chen et al., 2020b] model. 2×: teacher has 2× parameters than the student.

Arpit et al. [2017], Zhang et al. [2021], Liu et al. [2020], who showed that noise memorization happens later in the training procedure.

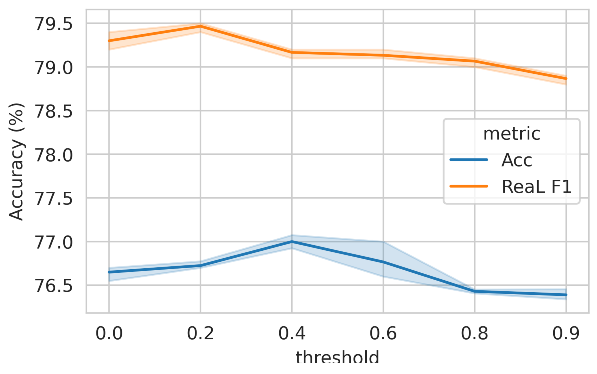
### 6.3.2. Self-supervised Fine-tuning

ImageNet's label ambiguity [Stock and Cisse, 2018, Tsipras et al., 2020b, Shankar et al., 2020, Beyer et al., 2020, Yun et al., 2021] might be problematic for fully-supervised methods but it is possible that self-supervised pre-training procedures such as MoCo [He et al., 2019a] or SimCLR [Chen et al., 2020b] are immune to it. We explore whether iterated learning improves the performance of self-supervised models in the fully- and semi-supervised fine-tuning regimes. We perform experiments on the ImageNet dataset and report validation accuracy and ReaL-F1 as described in Sec. 6.3.1.

**Baselines.** We report results with ResNet-50 pre-trained with SimCLR [Chen et al., 2020b], SimCLR-v2 [Chen et al., 2020c], BYOL [Grill et al., 2020], MoCo-v2 [Chen et al., 2020d], and SwAV [Goyal et al., 2021]. Results are reported after fine-tuning weights with 1%, 10%, and 100% of the ImageNet training set. We use the same data subsets as Chen et al. [2020c]. We



**(a)** Iterations



**(b)** Threshold

**Fig. 6.3. Ablation study.** Comparison between different iteration schedules. (a) Sweep over length of teacher training $k_t$ and length of student training $k_s$. We report the ReaL-F1 score. (b) ReaL F-1 and accuracy scores for a threshold value sweep ($\rho$).

incorporate the proposed iterative learning procedure in the fine-tuning process of MoCo-v2 and SimCLR-v2. For SimCLR-v2, we also tested the "sk1" variant which was improved with selective kernels [Li et al., 2019a, Chen et al., 2020c], while "sk0" is the vanilla version. For the semi-supervised learning experiments, we compare with SimCLR-v2's distillation experiments, where a teacher predicts pseudo-labels on unlabeled data. We compare with ResNet-50 (2×+SK), where the teacher has 2× capacity than the student, and ResNet-50 (1×+SK) where the teacher and the student are the same models.

**Results.** We report fine-tuning results in Table 6.3. Iterated learning improves the performance of MoCo-v2, SimCLR, and SimCLR-v2 for all fine-tuning data fractions. Interestingly, the improvement gap grows when using better self-supervised initializations. For example, the ReaL improvement from the best performing SimCLR-v2-sk1 with 100% of the validation data is 4.6% while it is around 3% for MoCo-v2 and SimCLR-v2-sk0. We hypothesize that more accurate models lead to better teachers, improving the overall performance of the iterated learning procedure.

We report semi-supervised learning results in Table 6.4. Iterated learning performs 2.9% better with 1% of the training labels and 0.9% with 10% of the training labels when compared with the self-distillation procedure presented in SimCLR-v2 Chen et al. [2020c]. Interestingly, we find that iterated learning attains better performance than distilling from a teacher twice the size of the student.

|  | **F1@0.25** | **F1@0.5** |
|---|---|---|
| Softmax | 28.69 | 28.69 |
| Sigmoid | 29.10 | 28.67 |
| MILe (ours) | **41.35** | **34.32** |

**Table 6.5. Results on multi-label MNIST.** The first column displays the F1 score when the threshold for positive labels is set to 0.25 and the second column shows the F1 score for a threshold of 0.5.

### 6.3.3. Analysis

In this section we explore the behavior of MILe under different hyperparameter settings as well as more challenging setups with synthetic data.

Number of Iterations. We investigate the effect of the number of teacher iterations ($k_t$) and student iterations ($k_s$) per cycle on the final performance (Fig. 6.3a). We report the ReaL-F1 for different $k_t$ values (rows) and $k_s$ values (columns). In general, we find that good performance can be achieved with a wide range of $k_t$ and $k_s$ combinations. The best performance is achieved with smaller values of $k_t$ and $k_s$. Extreme values of $k_t$ and $k_s$ lead to lower performance, with the model being most sensitive to large values of $k_s$ (dark regions). This is expected since a small $k_t$ would let the imitation phase constantly disrupt supervised learning via interaction with the data, while a large $k_t$ does not reap the benefits of distillation. For a given $k_t$ we find that the optimal $k_s$ lies in the mid-range and the other way around. Regarding the influence of the dataset size, we observe that it mostly influences the optimal number of teacher iterations ($k_t$). We hypothesize that it takes few iterations for the teacher to overfit small datasets, which leads to one-hot predictions and prevents the model from learning a multi-label hierarchy.

Pseudo-label Threshold Ablation Study. In this section, we conduct an ablation study on the threshold value ($\rho$) used by MILe to produce multi-pseudo-labels from sigmoid output activations (see Section 4.3 and Algorithm 2). Fig. 6.3b shows the validation accuracies and ReaL-F1 scores for different threshold values. Lower thresholds bias the student towards producing multi-label outputs, even for low-confidence classes. Larger threshold values make the student tend towards singly-labeled prediction, only predicting labels for which the confidence is high. In the extreme, a high threshold constrains the teacher to predict empty label vectors. Interestingly, we find that lower threshold values result in higher ReaL-F1 score and better accuracy. In fact, the Real-F1 score benefits from lower $\rho$ than the accuracy. This is due to the fact that lower thresholds increase the number of predicted labels per image, which improves the recall in multi-label evaluations.

Multi-label MNIST. Many images in the real world datasets like WebVision or ImageNet contain a single object, which biases MILe towards predicting a small number of objects per

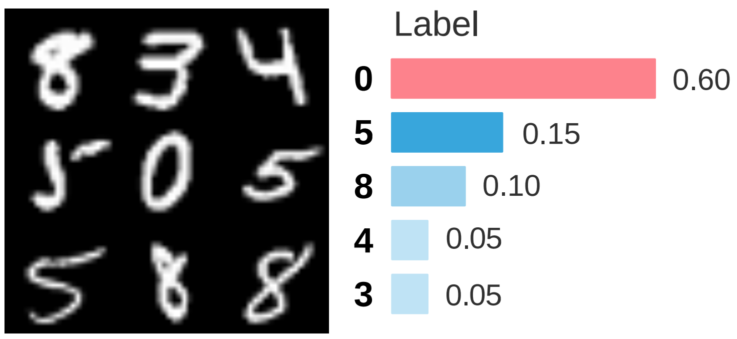


**Fig. 6.4. Multi-MNIST.** The center digit has a probability of 0.6 to be chosen as the label for the whole grid.

image. In order to explore the limits of MILe, we begin by designing a controlled experiment on a synthetic dataset where most samples contain multiple classes. Each sample consists of a $3 \times 3$ grid of randomly sampled MNIST digits [LeCun and Cortes, 2005]. For each grid, its single label corresponds to the center digit with probability 0.6 while the 8 remaining digits are sampled with probability 0.05 each (see Fig. 6.4). Note that, similar to the ImageNet, digits of the same class can repeat in the grid. However, the probability of having a $3 \times 3$ grid with the same digit repeated in each position is $10^{-9}$.

Results are shown in Table 6.5. We observe that MILe attains up to 12% better F1 score than the Softmax and Sigmoid baselines. It is worth noting that the improvement is most significant when thresholding the sigmoid output predictions to 0.25. Interestingly, for this experiment, we found the best threshold to produce multi-pseudo-labels from the teacher output to be ($\rho = 0.1$). Having a low threshold biases the student towards producing multi-label outputs. We find these results encouraging and we believe that better performance could be attained by improving the pseudo-multi-label generation strategy. We plan to explore these new strategies in future work.
Contribution of Self-Distillation and Iterated Learning. Here, we study of the effect of the multi-label distillation algorithm on the iterative procedure. We compare soft distillation (softmax + KL loss) with hard distillation (argmax + CE), and MILe (sigmoid + threshold + BCE) with and without iterated learning in Fig. 6.5. We compare the effect on two and many iterations. Hard labels outperform soft labels when training with many iterations. We provide an ablation of iterated learning with nosiy-student [Xie et al., 2020] distillation procedure depicted in Fig. B.7.3 of the supplementary material.

## 6.4. Related work

It is known that weakly-labeled datasets such as ImageNet contain label ambiguity [Stock and Cisse, 2018, Tsipras et al., 2020b, Shankar et al., 2020, Beyer et al., 2020, Yun et al., 2021, Barbu et al., 2019] and label noise [Van Horn et al., 2015, Recht et al., 2019b]. Label

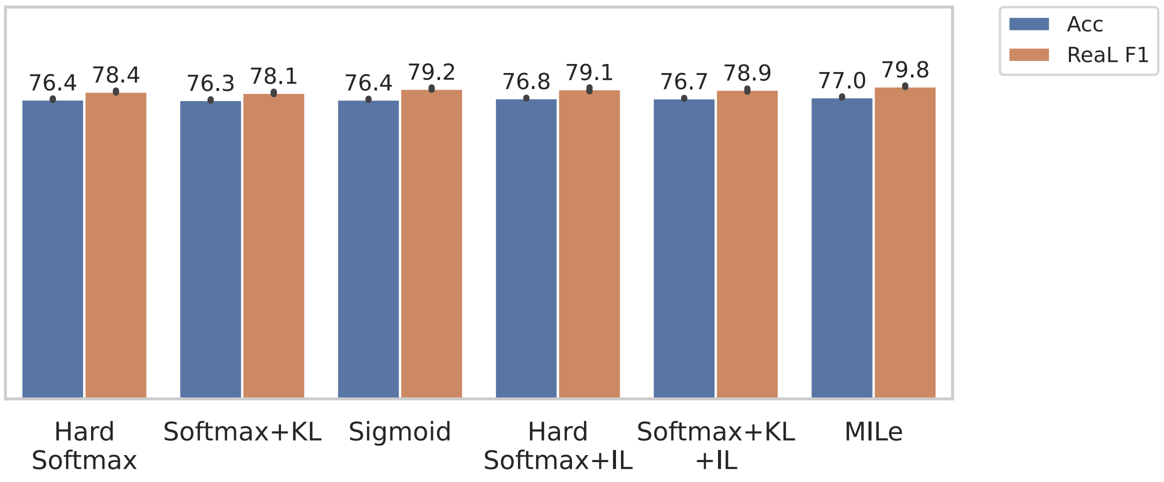


**Fig. 6.5.** Comparison between different distillation schedules and MILe. We report the Accuracy and ReaL-F1 score.

ambiguity refers to the cases where only one of the multiple possible labels was assigned to the image. In order to evaluate how label ambiguity affects ImageNet classifiers, Beyer et al. [2020] proposed ReaL, a curated version of the ImageNet validation set with multiple labels per image. They found that ImageNet classifiers tend to perform better on ReaL since it contains less label noise but they did not address the problem of inaccurate supervision during training where more than one correct class is present in the image. To deal with unfavorable training dynamics due to the mismatch between the multiplicity of object classes and the majority-aggregated single labels, Yun et al. [2021] proposed to relabel the ImageNet training set. They obtained pixel-wise labels by finetuning an ensemble of large models pretrained on a large external dataset Sun et al. [2017]. Although useful, undertaking such relabeling procedure for each dataset of interest is both laborious and unrealistic. In addition, it is not clear if the same relabeling approach could be used in larger, noisier databases such as WebVision [Li et al., 2017b], which contains 2.4M images downloaded from the internet and labels consisting of the queries used to download those images. In this work, we investigate the use of iterated learning on weak singly-labeled datasets as an alternative to relabeling in order to produce a multi-label output space. Different from existing methods, MILe uses neither external data nor additional relabeling procedures.

Knowledge Distillation. Knowledge distillation is a technique commonly used in model compression [Buciluǎ et al., 2006, Hinton et al., 2015, Ba and Caruana, 2013]. In the vanilla setting, a large deep neural network is used as a teacher to train a smaller student network from its logits. In addition to model compression, knowledge distillation has been used to improve the generalization of student networks reusing distilled students as teachers Furlanello et al. [2018] or distilling ensembles into a single model [Allen-Zhu and Li, 2020]. Gains have been observed even when the teacher and the student model are the same network, a regime

commonly known as self-distillation [Mobahi et al., 2020, Zhang et al., 2019, Allen-Zhu and Li, 2020]. Mobahi et al. [2020] further showed that iterative self-distillation induces a strong regularization effect, with effects that are different from early stopping. Self-distillation has also been used to improve the generalization and robustness of semi-supervised models. Xie et al. [2020] introduced noisy student for labeling unlabeled data during semi-supervised learning. While MILe also leverages teacher and student networks, it is fundamentally different from knowledge distillation approaches. The goal of knowledge distillation is to transmit all the knowledge of a teacher network to a student network. On the other hand, MILe trains a succession of short-lived teacher and student generations, thus creating an iterated learning bottleneck [Kirby, 2001], to construct a new multi-label representation of the images from single labels. This goal is also different from the goal of noisy student, which is to label unlabeled data, and which is trained three times until convergence.
Iterated Learning. The iterated learning hypothesis was first proposed by Kirby [2001, 2002] to explain language evolution via cultural transmission in humans. Languages need to be expressive and compressible to be effectively transmitted through generations. This learning bottleneck favors languages that are compositional as they can be easily and quickly learned by the offsprings and support generalization. Kirby et al. [2014] conducted human experiments and mathematical modeling, which showed that iterated transmission of unstructured language results in convergence to a compositional language. Since then, it has seen many successful applications, especially in the emergent communication literature [Guo et al., 2019, Ren et al., 2020, Cogswell et al., 2019, Dagan et al., 2020]. In these settings, the learning bottleneck is induced by limiting the data or learning time of the student, which helps it to converge to a compositional language that is easier to learn [Li and Bowling, 2019]. The approach starts by training a *teacher network* with a small number of updates on the training set. A *student network* is then trained to imitate the teacher based on pseudo-multi-labels inferred from the input samples. The student then replaces the teacher and the cycle repeats with a frequency modulated by a learning budget. Iterated learning has also been used in the preservation of linguistic structure in addition to its emergence by Lu et al. [2020a,b]. Furthermore, Vani et al. [2021] successfully applied it for emergent systematicity in VQA. To the best of our knowledge, this is the first application of the iterated learning framework in the visual domain.

## 6.5. Conclusion

We introduce multi-label iterated learning (MILe) to address the problem of label ambiguity and label noise in popular classification datasets such as ImageNet. MILe leverages iterated learning to build a rich supervisory signal from weak supervision. It relaxes the singly-labeled classification problem to multi-label binary classification and alternates the training of a teacher and a student network to build a multi-label description of an image from single labels. The teacher and the student are trained for few iterations in order to prevent them from overfitting the singly-labeled noisy predictions. MILe improves the performance of image classifiers for the singly-labeled and multi-label problems, domain generalization, semi-supervised learning, and continual learning on IIRC. A possible limitation, which is inherent to iterated learning [Lu et al., 2020a], is choosing the correct length of teacher ($k_t$) and student iterations ($k_s$). However, our ablation experiments suggest that the proposed procedure is beneficial for a wide range of $k_t$ and $k_s$ values (Sec. 6.3.3). MILe also depends on the threshold value $\rho$, which we use to produce pseudo-labels from the teacher's outputs. However, we found encouraging that low values of $\rho$ improve the performance of the classifiers, indicating that predicting multiple labels is beneficial. With respect to the computational cost, we found that the impact of MILe is lower than the validation phase of the models (see Sec. 6.2.2). Overall, we found that iterated learning improves the performance of models trained with weakly labeled data, helping them to overcome problems related to label ambiguity and noise. Broader impact and future work. Our approach is built on the hypothesis that the world is structured along objects and the fact that images result from the composition of those objects. We believe that our work could be applied to other tasks that build on the same assumptions such as object detection, segmentation, and multiple-instance learning. In these cases we hope approaches like MILe could open the door to leverage large amounts of webly supervised data to improve on these tasks.

# Chapter 7

# Prologue to Article 3

## 7.1. Article Details

**Haptics based Curiosity from Sparse Reward Tasks**. Sai Rajeswar*, Cyril Ibrahim*, Nitin Surya, Florain Golemo, David Vazquez, Aaron Courville, Pedro Pinheiro. Published and presented at International Conference on Robot Learning (CoRL), 2021.

*Personal Contribution* The idea of the project were primarily conceived by the author, and refined over discussions with Pedro Pinheiro and Aaron Courville. The author is responsible for running the experiments, writing parts of the paper, and coming up with an appropriate version of the dynamics prediction model used in the work. The author and Cyril Ibrahim were involved in writing all of the code. Pedro and Aaron Corville advised on the project and writing parts of the paper. Pedro was also involved in discussing results, suggesting experiments to run, and making the paper strong.

## 7.2. Context

The goal of this work was to study and help address the shortcomings of Reinforcement learning manipulation based control tasks where reward signal is mostly sparse and often non-existing. In contrast to RL agents, humans can learn behaviors without any external rewards, due to the intrinsic motivation that naturally drives them to be active and explore the environment [LAR, 2011, Legault, 2016]. The design of similar mechanisms for RL agents opens up possibilities for training and evaluating agents without external rewards, fostering more self-supervised strategies of learning.

## 7.3. Research Impact

The paper presents a cross-modal curiosity based self-supervised objective to encourage efficient exploration. This in turn helps in generating structured data that helps adapt to downstream tasks in a sample efficient manner. Ours is the one of the first attempts to leverage tactile sensing to learn efficient exploration strategies in conjunction with decision making. We are beginning to see other recent work building on HaC that leverages tactile modality for unsupervised RL problems [Xu et al., 2022]. While being competitive compared to existing intrinsic exploration approaches, we show that formulating such dense intrinsic rewards could help towards solving challenging control tasks that were previously unsolved.

**Note**: What follows is a slightly abridged version of the CoRL publication, with additional qualitative results that were not included in the published version due to space limitations.

# Chapter 8

# Article 3: Haptics based Curiosity from Sparse Reward Tasks

## Abstract

Robots in many real-world settings have access to force/torque sensors in their gripper and tactile sensing is often necessary for tasks that involve contact-rich motion. In this work, we leverage surprise from mismatches in haptics feedback to guide exploration in hard sparse-reward reinforcement learning tasks. Our approach, Haptics-based Curiosity (HaC), learns what visible objects interactions are supposed to “feel" like. We encourage exploration by rewarding interactions where the expectation and the experience do not match. We test our approach on a range of haptics-intensive robot arm tasks (e.g. pushing objects, opening doors), which we also release as part of this work. Across multiple experiments in a simulated setting, we demonstrate that our method is able to learn these difficult tasks through sparse reward and curiosity alone. We compare our cross-modal approach to single-modality (haptics- or vision-only) approaches as well as other curiosity-based methods and find that our method performs better and is more sample-efficient.

## 8.1. Introduction

Most successes in reinforcement learning (RL) come from games [Mnih et al., 2013, Silver et al., 2016] or scenarios where the reward is strongly shaped [Zhu et al., 2018b, Forestier et al., 2017]. In the former, the environment is often fully observable, and the reward is dense and well-defined. In the latter, a large amount of work is required to design useful reward functions. While it may be possible to hand-craft dense reward signals for many real-world tasks, we believe that it is a worthwhile endeavor to investigate learning methods that do not require dense rewards.

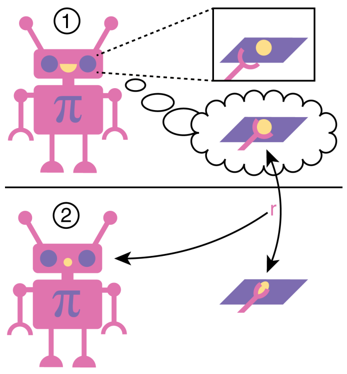


**Fig. 8.1. Touch-based Curiosity (HaC) Overview** *Top*: An agent perceives a scene visually and anticipates the force/torque (FT) sensation of interacting with an object. *Bottom*: The object interaction leads to an unexpected FT sensation, which gives a positive reward to the policy, leading to an exploration policy that is seeking interactions that are haptically surprising. The agent's experiences gained in this way are later relabeled to become task-specific.

Closely related to the sparse rewards problem is the issue of exploration. One reason that traditional RL agents struggle with sparse-reward problems is a lack of exploration. An agent may not obtain useful rewards without an intuitive exploration strategy when rewards are sparse. Exploration based on intrinsic curiosity comes naturally to many animals and infants (who start crawling and exploring the environment at around 9 months [Vernon et al., 2011] and oftentimes even before they can crawl by using their hands and mouth to touch and probe objects). Touch is a local experience and encodes accurate geometrical information while handling objects. Experimental studies in infants has suggested that tactile and visual sensory modalities play a central role in systematic learning of tasks related to object understanding, interaction and manipulation [E and CH, 2010;15(3, Connolly and Harris, 1971].

Ideally, we would like our RL agents to explore the environment in an analogous self-guided fashion to learn the dynamics and object properties, and use this knowledge to solve downstream tasks. Just as how humans utilize different sensory modalities to explore and understand the world around them, exploration in robots should be more embodied and related to a combination of vision and touch and potentially other sensor modalities. We believe that building autonomous agents that are self-driven and seek to explore via multi-modal interaction are crucial to address key issues in developmental robotics.

Recent works in RL have focused on a curiosity-driven exploration through prediction-based surprise [Burda et al., 2019a, Pathak et al., 2017a, Raileanu and Rocktäschel, 2020]. In this formulation, a forward dynamics model predicts the future, and if its prediction is incorrect when compared to the real future, the agent is surprised and is thus rewarded. This encourages the agent to look for novel states while improving its visual forward model in return. However, this formulation can be practically challenging to optimize since there are many states that are visually dissimilar but practically irrelevant (e.g. for a pushing task, moving a robotic end-effector without touching the object creates visual novelty but contributes little to task-related knowledge). One way to constrain this search space over curious behaviors is by involving another modality like haptics.

In this work, we demonstrate that a self-guided cross-modal exploration policy can help solve sparse-reward downstream tasks that existing methods without this curiosity struggle to solve. Our method uses cross-modal consistency (mismatch between visual and haptic signal) to guide this exploration. To use self-play knowledge in downstream tasks, we relabel past experiences, providing a dense reward signal that allows modern off-policy RL methods to solve the tasks. While there are many existing methods that use artificial curiosity/intrinsic motivation, the majority of these methods either rely on strong domain knowledge (e.g. labels of state dimensions in Forestier et al. [2017], a goal-picking strategy in Andrychowicz et al. [2017]) or are prone to get stuck in local optima when a single meaningless stimulus creates enough surprise to capture the attention of the agent (e.g. noisy-TV experiment from Burda et al. [2019b]). Other approaches depend on unrealistic assumptions and goal conditioning [Andrychowicz et al., 2017]. Our method presents a novel approach in the family of prediction-based models [Burda et al., 2019a, Huang et al., 2019b, Pathak et al., 2017a] and yields better performance on a wide range of robotic manipulation tasks than purely vision-based and haptics-based approaches [Burda et al., 2019a, Pathak et al., 2017a]. The tasks are chosen with careful consideration—they comprise of preliminary robotic manipulations such as grasping, pushing, and pulling. In this work, we present the following contributions:

- A new curiosity method to help solve sparse-reward tasks that use cross-modal consistency (predicting one modality from another) to guide exploration. We implement it in this work as vision and touch modalities, but the formulation of our method does not require any knowledge about the underlying modalities and can thus be applied to other settings.
- We create and maintain a manipulation benchmark of simulated tasks, *MiniTouch*, inspired by Chen et al. [2020a], Andrychowicz et al. [2017], where the robotic arm is equipped with a force/torque sensor. This allows evaluation of models' performance on different manipulation tasks that can leverage cross-modal learning.
- We validate the performance of our method on MiniTouch environment comprising of four downstream tasks. We compare purely vision-based curiosity approaches and standard off-policy RL algorithms. Our method improves both performance and sample efficiency.

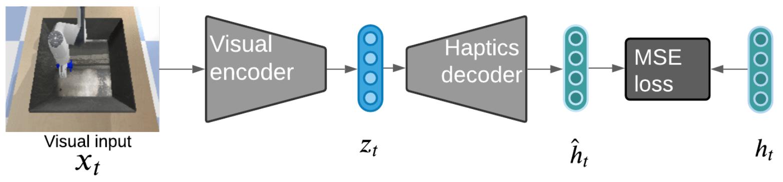
Project website: https://fgolemo.github.io/haptics-based-curiosity/

## 8.2. Related work

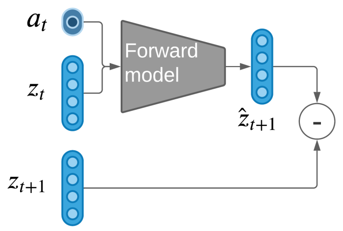
Intrinsic Motivation. Intrinsic motivation is an inherent spontaneous tendency to be curious or to seek something novel in order to further enhance one's skill and knowledge White [1959], Barto et al. [2004], Di Domenico and Ryan [2017]. This principle is shown to work well even in the absence of a well-defined goal or objective. In reinforcement learning, intrinsic motivation has been a well-researched topic [Schmidhuber, 1991, Oudeyer et al., 2007, Oudeyer and Kaplan, 2009, Lair et al., 2019, Marino et al., 2019, Savinov et al., 2018]. An intuitive way to perform intrinsic motivation is through the use of "novelty discovery". For example, incentivize the RL agent to visit unusual states or states with substantial information gain [Houthooft et al., 2016]. In its simplest form, this can be achieved with up-weighting mechanisms such as state visitation counts [Strehl and Littman, 2008]. Count-based methods have also been extended to high-dimensional state spaces [Bellemare et al., 2016, Ostrovski et al., 2017a]. Alternative forms of intrinsic motivation include disagreement [Sekar et al., 2020], empowerment [Klyubin et al., 2005, Gregor et al., 2016].

Exploratory intrinsic motivation can also be achieved through "curiosity" [Schmidhuber, 1991, Dubey and Griffiths, 2020]. In this setting, an agent is encouraged to visit states with high predictive errors [Raileanu and Rocktäschel, 2020, Burda et al., 2019a, Pathak et al., 2017a] by training a forward dynamics model that predicts the future state given the current state and action. Instead of making predictions in the raw visual space, Pathak et al. [2017a] mapped images to a feature space where relevant information is represented via an inverse dynamics model. Burda et al. [2019a] demonstrate that random features are sufficient for many popular RL game benchmarks. This approach may work well with tasks that require navigation to find a reward because each unseen position of the agent in the world leads to high intrinsic reward when unseen. However, in the case of manipulative tasks, we are less interested in the robot visiting all the possible states and more interested in states where the robot interacts with other objects. In this work, we leverage multimodal inputs that encourage the agent to find novel combinations of visual and force/torque modalities.

Self-Supervised Learning via Cross-modality. Exploiting multimodality to learn unsupervised representations dates back to at least 1993 [de Sa, 1993]. Multimodal signals are naturally suitable for self-supervised learning, as information from one modality can be used to supervise learning for another modality [Gao et al., 2018, Owens and Efros, 2018]. Different modalities typically carry different information, e.g., visual and touch sensory modalities emerge concurrently and often in an interrelated manner during contact-rich manipulation tasks [Blake et al., 2004]. Specifically, force/torque motor signal has always been a major component in the literature of perception and control [Kalakrishnan et al., 2011, Levine et al., 2016a, Liu et al., 2017].



**(a)** Haptics Control module



**(b)** Forward module

**Fig. 8.2. Haptics-based Curiosity Model.** (a) The input image $x_t$ at time $t$ is transformed into a 256-dimensional feature vector $z_t = enc(x_t)$ using a CNN encoder. The haptics decoder network predicts corresponding force/torque vector $\hat{h}_t = dec(z_t)$. The $L2$ norm between predicted haptics $\hat{h}_t$ and observed haptics $h_t$ is used as exploration reward. (b) To stabilize training, an additional network is used to predict the forward dynamics, and the difference between predicted next latent state $\hat{z}_{t+1}$ and actual next latent state $z_{t+1}$ is used as weighted additional term in the reward.

The most common ways to leverage multimodal signals to learn representations are through vision and language [Srivastava and Salakhutdinov, 2012, Gan et al., 2020]. Gao et al. [2016], Li et al. [2019b] demonstrated a unified approach to learning representations for prediction tasks using visual and touch data. Chen et al. [2021] learn world-models from multimodal data via a shared latent space. In robotics and interactive settings, the use of modalities such as tactile sensing [van Hoof et al., 2016, Calandra et al., 2018, Murali et al., 2018] is increasingly popular for grasping, manipulation and other externally-specified tasks [Pape et al., 2012, Lee et al., 2019a]. Lee et al. [2019b] showed the effectiveness of self-supervised training of tactile and visual representations by demonstrating its use on a peg insertion task.

While the mentioned approaches have used multiple sensory modalities for learning better representations, in this work we demonstrate its utility for allowing agents to explore. Similar to ours, Dean et al. [2020] use multimodal sensory association (i.e. audio and visual) to compute the intrinsic reward. Their curiosity-based formulation allows an agent to efficiently explore the environment in settings where audio and visual signals are governed by the same physical processes. In addition to the different nature of sensory signals, they use a discriminator that determines whether an observed multimodal pair is novel. This might not work in our case as touch is a more sparse signal and using a discriminator could lead to ambiguous outcomes.

## 8.3. Background

The goal of our method, Haptics-Based Curiosity (HaC), is to encourage the agent to interact with objects. HaC provides a reward signal for an RL agent to explore the state space of a task that involves interacting with objects. The exploration phase is independent of the downstream task, i.e., relying solely on visual and force/torque signals, without a reward signal from the downstream task.

Similar to how people spend more time exploring stimuli that are more incongruous [Connolly and Harris, 1971], HaC guides the agent to focus its experience on different novel cross-modal associations. We augment this intrinsic objective with a future visual state prediction objective similar to the one in Pathak et al. [2017a] to avoid getting stuck in undesired inactive configurations. Note that we sometimes refer to the future state prediction objective in the text as forward dynamics objective. In this work, we focus on the cross-modality between vision and touch, but the same idea could be applied to other pairs of sensory domains, such as vision and sound, or touch and acoustics.

### 8.3.1. Problem Formulation

The learning problem is formalized as a Markov decision process (MDP) defined by a tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho, \gamma\}$ of states, actions, transition probability, reward, initial state distribution, and discount factor. The goal is to find the optimal policy that maximizes the discounted sum of rewards, $\pi^* = \mathbb{E}_\pi[\sum_t^\infty \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)]$. In our case, each state $s_t$ in the trajectory is composed of both visual and corresponding touch features as detailed in the following section. We use soft actor-critic policy gradients (SAC) [Haarnoja et al., 2018] to train our policies, but in principle, our proposed approach is algorithm-agnostic. The policy $\pi$ is evaluated with an estimation of the soft Q-value:

$$Q(\mathbf{s}_t, \mathbf{a}_t) \triangleq r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{\mathbf{s}_{t+1}\sim p}[V(\mathbf{s}_{t+1})] , \tag{8.3.1}$$

where $V(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t\sim\pi}[Q(\mathbf{s}_t, \mathbf{a}_t) - \log \pi(\mathbf{a}_t|\mathbf{s}_t)]$ is the soft value function.

## 8.4. Curiosity based on Touch Prediction

Our core prediction framework consists of two modules: (i) a *haptics control* module, which learns to predict expected haptic signal from the visual input, and (ii) a *forward dynamics* model, which predicts the next latent state from the current latent state and the current action (see Figure 8.2). Let the state of the environment $\mathbf{s}_t = (z_t, h_t)$ at time $t$ be composed of a visual feature $z_t$ (the encoded visual input) and a haptic signal $h_t$. The touch prediction model consists of a convolutional encoder $z_t = enc(x_t)$ that transforms the image $x_t$ into a latent representation $z_t$ and a fully-connected decoder $\hat{h}_t = dec(z_t)$ that transforms the latent into a predicted haptic signal $\hat{h}_t$. The encoder-decoder is trained with an L2 reconstruction loss, i.e. for every image $x_t$ and force/torque sensor $h_t$:

$$L_{haptics} = \left\|\hat{h}_t - h_t\right\|_2 . \tag{8.4.1}$$

A high prediction error on a given image indicates that the agent has had few interactions like this. Therefore, to harness this “surprise" to guide exploration, we define the intrinsic reward at time $t$ during exploration to be proportional to this reconstruction loss. This essentially allows the agent policy to visit under-explored configurations of the state space by encouraging interactions where the system does not know what the target object “feels” like. In addition to efficient exploration, being aware of such incongruity via touch prediction aids learning local regularities in the visual input. This in turn could assist better generalization to unseen states. An overall pipeline of the framework is shown in Figure 8.2a.

## 8.4.1. Regularization Through Forward Dynamics Model

We found empirically (and we demonstrate in the experiments section below) that the surprise stemming from haptic novelty was not enough to cause object-centric interaction. We postulate that by incorporating visual surprise (i.e. the mismatch between predicted forward dynamics and observed dynamics) [Pathak et al., 2017a], we can create an agent that seeks out visual novelty as well as haptic one and thus leads to better state space coverage. To this end, our model is augmented with a forward dynamics model (see Figure-8.2b) $\hat{z}_{t+1} = fdm(z_t, a_t)$ that learns to map the latent state $z_t$ (obtained from the visual encoder $enc$) and action $a_t$ at time $t$ to the predicted latent state $\hat{z}_{t+1}$ at the next timestep. This model is also trained with L2 loss:

$$L_{fdm} = \|\hat{z}_{t+1} - z_{t+1}\|_2 = \|fdm(enc(x_t), a_t) - enc(x_{t+1})\|_2 \ . \tag{8.4.2}$$

The intrinsic reward is defined as the convex combination of the cross-modal prediction loss (Eq. 8.4.1) and the forward dynamics model loss (Eq. 8.4.2):

$$r_t = (1 - \lambda) \cdot L_{haptics} + \lambda \cdot L_{fdm} \ , \tag{8.4.3}$$

where $\lambda \in [0,1]$ is a balancing factor. The effect of the factor $\lambda$ on overall performance is studied in the ablation experiments described in Section 8.8.1.

## 8.4.2. Training

Learning is divided into two stages: (i) an *exploratory* step, where the agent performs free exploration following HaC, and (ii) an *adaptation* step, where the agent is tasked to solve a downstream problem, given a sparse reward.

During the exploratory step, each trajectory consists of pairs of image and force/torque features, $(z_1, h_1)$, $(z_2, h_2)$,..., $(z_n, h_n)$. These trajectories are used for two purposes: (i) updating the parameters of the prediction model to help shape the representations and (ii) updating the exploration policy based on the intrinsic reward $r_t$. Note that we use the touch features only to craft the intrinsic rewards and the input to RL agent consists of visual features alone to be comparable to the baselines. For vision-based curiosity models, Burda

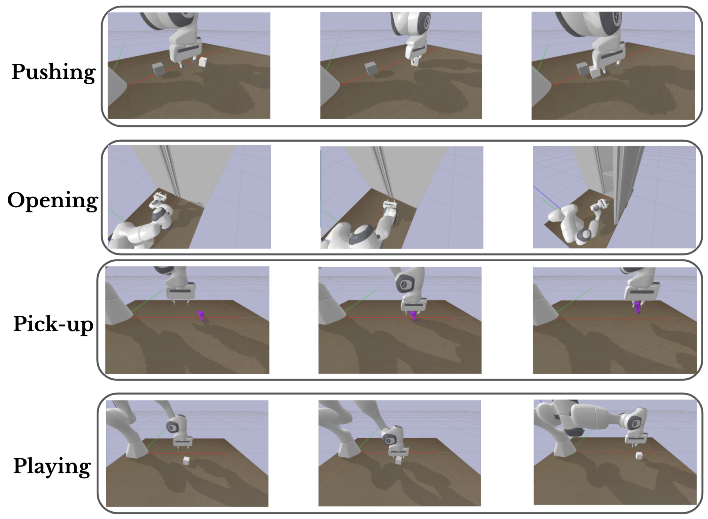


**Fig. 8.3. 'MiniTouch' benchmark tasks** include opening a door, pushing an object to a target, grasping and lifting an object and a toy task in which object interactions are counted.

et al. [2019a] observed that encoding visual features via a random network constitute a simple and effective strategy compared to learned features. In Section 8.8.1, we investigate the performance of our model in both scenarios, i.e., when the features are learned vs random. The overall optimization problem at this step consists of the policy learning (driven by intrinsic reward), the touch reconstruction loss (Eq. 8.4.1), and the forward dynamics loss (Eq. 8.4.2). During the downstream adaptation step, the parameters of the policy network, the Q network and the replay buffer are retained from the exploratory phase. The objective of the down-stream task is computed as:

$$\min_{\theta} \left[ -\mathbb{E}_{\pi} \left[ \sum_t r_t^e \right] \right] \ , \tag{8.4.4}$$

where $r_t^e$ in this phase is task-specific external sparse reward. In both steps, the objectives are optimized with Adam [Kingma and Ba, 2015].

## 8.5. MiniTouch Benchmark

**Implementation Details**: The encoder is a four-layered strided CNN followed by a fully connected network. We use LeakyReLU [Xu et al., 2015] as non-linear activation in all the layers. The decoder network is a two-layered MLP that maps 256-dimensional visual features to touch vectors. For a more detailed description of the networks (SAC policy network and HaC networks, including the forward model) and hyper-parameters, please refer to the supplementary material Section C.2.

## 8.6. Experimental Setting

Our experiments focus on a tabletop robot manipulation from raw image observations and raw force/torque sensory values, which we refer to as "touch vector". For our experiments, we use a 7-DoF Franka Emika Panda arm with a two-finger parallel gripper. Each of the fingers is equipped with a simulated force/torque sensor that measures the joint reaction force applied to it. We utilize PyBullet [Coumans and Bai, 2016–2019] to simulate the robot arm and haptic sensor.

### 8.6.1. MiniTouch Benchmark

Our proposed benchmark, MiniTouch, consists of four manipulation tasks: Playing, Pushing, Opening, Pick-up. Each of the tasks along with corresponding actions, observations, and rewards is described in detail in the Supplementary Material Section C.1 and further illustrated in Figure 8.3. MiniTouch is an active repository and we expect to update the benchmark with new tasks and datasets. The tasks are inspired by Yu et al. [2019] but are *not* based on a proprietary simulator, feature an arm that we have access to for real-world experimentation (in follow-up work), and where the arm is equipped with a haptic sensor.

## 8.7. Experiments and Results

### 8.7.1. Baseline Comparisons and Metrics

For the task evaluation, we study two versions of our model: (i) HaC-Pure considering the touch vector reconstruction intrinsic reward alone, and (ii) HaC, considering the full intrinsic reward (Eq. 8.4.3). We compare with several well-known intrinsic exploration baselines based on visual features:

- **SAC**: The unmodified Soft Actor-Critic algorithm from Haarnoja et al. [2018].
- **ICM**: SAC augmented with the state-of-the-art visual curiosity approach Intrinsic Curiosity Module (ICM) [Pathak et al., 2017b], which uses a visual prediction model to guide exploration.
- **Disagreement**: It uses model disagreement as objective for exploration [Pathak et al., 2019a, Sekar et al., 2020]. It leverages variance in the prediction of an ensemble of latent dynamics models as the reward.
- **RND**: Random Network Distillation [Burda et al., 2019b] utilizes a randomly initialized neural network to specify an intrinsic reward for visiting unexplored states in hard exploration problems.

Based on code from the official Franka Emika repo https://github.com/frankaemika/libfranka
https://github.com/ElementAI/MiniTouch

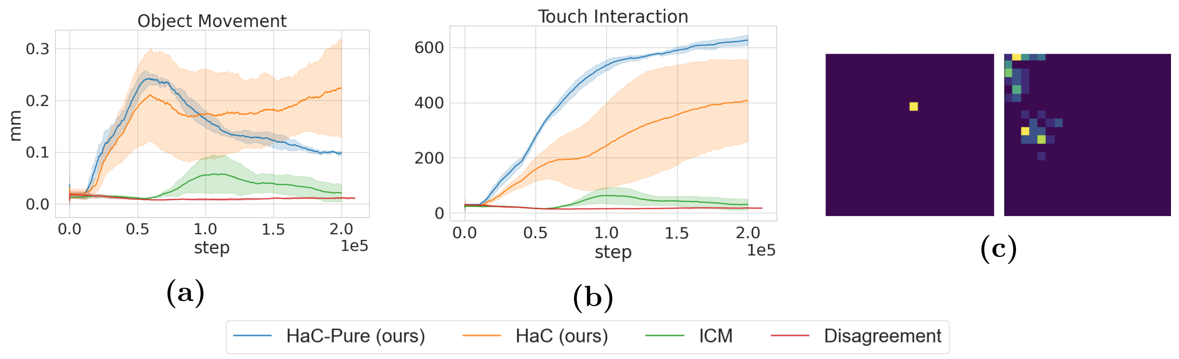


**Fig. 8.4. Object Interaction** To quantitatively evaluate an agent's object interaction, we consider both haptics interaction and object displacement. (a) shows the object movement evaluation (avg displacement in mm from the starting point) of our method compared with the baseline methods over training steps; (b) depicts the number of touch interactions, evaluated on the "playing" task. (c) Heat maps showing the object location at the beginning (*left*) and end (*right*) of the HaC training. The object locations are spread-out towards the end indicating interactive movement.

We built a Pytorch [Paszke et al., 2017] version of these baselines based on their open source code (details in supplementary material). We use the following metrics to evaluate our method and the baseline models:

**Exploration success:** measures the percentage of times that the agent attained the goal state in the *exploratory* phase, i.e. with no external reward. Higher is better.

**Success:** denotes the percentage of times that the agent attained the goal state during the *down-stream* task phase.

**Episode steps:** The number of steps required for each episode to succeed. This metric is an indicator of sample efficiency. The lesser the number of steps, the faster the agent's ability to succeed.

**Touch-interaction:** Amount of interaction the agent's fingers have with the underlying object. We measure this by computing the variance of force/torque sensory signal across the whole episode.

**Object movement:** The agent can resort to constantly engaging with the object unnecessarily to satisfy the objective. We, therefore, monitor the variance of door angle (for the Opening task) and the variance of object position (for the remaining tasks) over the course of training. A higher movement indicates diverse state space in addition to physical interaction.

## 8.8. Results and Discussion

HaC and baselines were trained on a Panda robot agent [Coumans and Bai, 2016–2019] for one million steps. In the *exploratory* phase of the training, we pre-train our method only with the curiosity-based intrinsic reward. We then progress to the *adaptation* phase. Also,

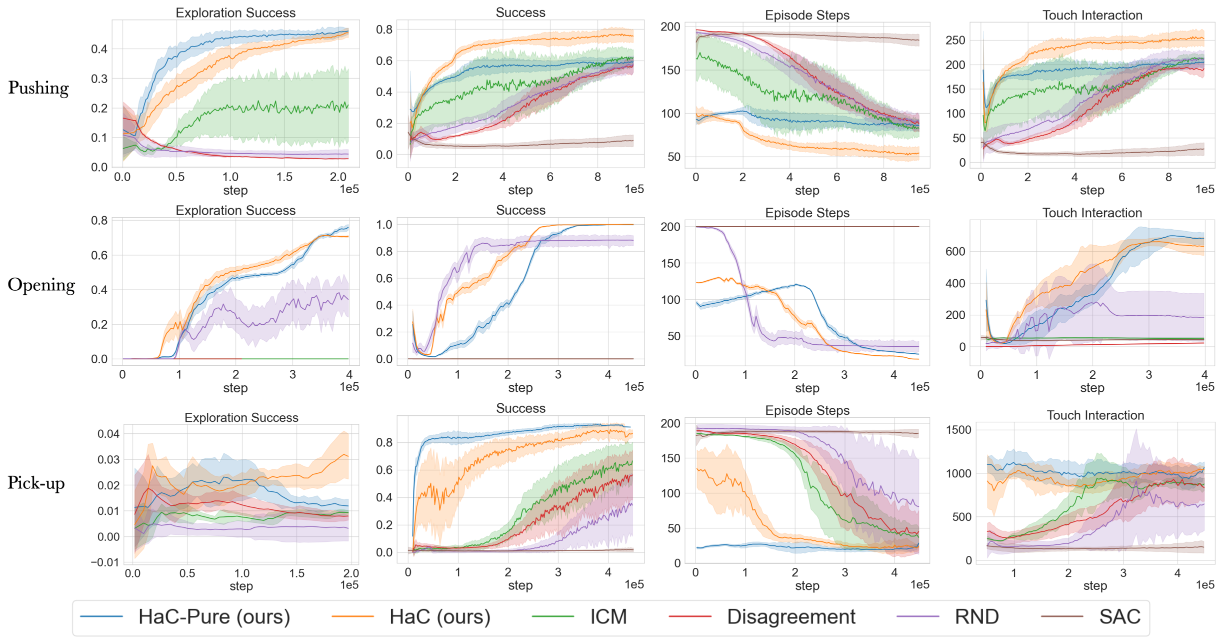


**Fig. 8.5. MiniTouch Evaluations**. Each row in the figure outlines the performance of the HaC variants and the baselines on a MiniTouch task. Each column marks performance on the four specified evaluation metrics over a number of training steps on the x-axis expressed in $1e^5$. The results are averaged across 5 random seeds and shaded areas represent mean $\pm$ one standard deviation while darker line represents the mean. In the majority of the tasks, HaC agents attain success in the *exploratory* phase with no external reward (see text). *Note*: We exclude the single object playing task as success in this task is equivalent to object interaction as depicted in Figure. 8.4.

note that across all tasks, HaC-Pure is based purely on cross-modal prediction, while HaC includes visual forward prediction reward in addition.

Figure 8.4a and 8.4b shows results on the basic task of playing with a single object. Since single object interaction does not have explicit goal states to evaluate, we instead measure the agent's ability to constantly engage and play with the underlying object. HaC-Pure displays four times better interaction with the underlying object when compared to SAC (see Figure 8.4b). Note from the plots that there is a trade-off when using HaC and HaC-Pure between constant interaction (i.e. touch interaction performance) and object movement dynamics. Collecting a variety of such interesting data during the *exploratory* phase helps the agent in terms of sample efficiency while solving the downstream tasks. Figure C.3.1 shows similar comparison for Open-door task. In the following section, we examine the role of the forward objective term on touch interaction and possible ways to encode agent's observation in our ablation studies.

We compare HaC and HaC-Pure with SAC and state-of-the-art vision-based curiosity baselines on the remaining downstream tasks in MiniTouch. From Figure 8.5 it is evident that HaC and HaC-Pure perform better than SAC in all the tasks and better than the

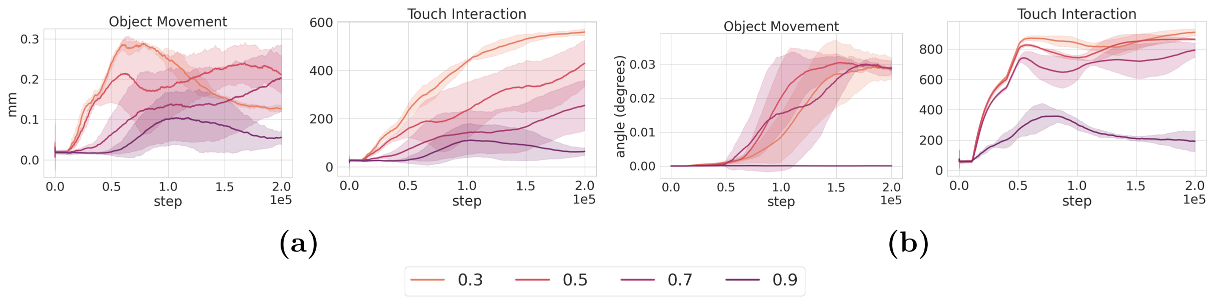


**Fig. 8.6. Forward Prediction weight.** Touch and Object movement evaluated on the (a) playing task and (b) Open-door task for different forward prediction weightings sampled $\sim [1, 0]$. Large weight (darker plot) favors object-movement towards the end of training, where as smaller weight improves touch-interaction. Middle value in the sweep range (*e.g.* 0.5) balances the trade-off. Note that movement is measured as distance(mm) for the Playing and angular distance for Open-door.

vision-based curiosity models in the majority of the tasks. Using SAC alone hinders the performance and is often unable to solve any of the three tasks. This is not surprising since the model is not motivated enough to collect diverse and useful data through interaction. We hypothesize that HaC-Pure without the visual prediction objective $L_{fdm}$ could potentially bias the solutions towards more physical interaction, which is not necessary for every task. For instance, constant interaction in the pushing task could be seen as a hindrance as it does not let the object slide easily towards the target. Observe that ICM performs better than HaC-Pure in the pushing task, however, HaC dominates in performance by about 15%. Recall that the goal is not just to succeed but to help attain success in a sample efficient manner in fewer steps. The results support our hypothesis that cross-modal curiosity enables an RL agent to succeed at an early stage in training and often without any external reward. Similarly, our model outperforms on the opening task without external reward. However, HaC initially has lower success compared to RND but surpasses RND towards the end. Although HaC and HaC-Pure attain similar success in the pick-up task towards the end of the training, it is compelling to note that HaC-Pure attains faster convergence. This is because the picking task requires constant touch interaction (where HaC-Pure has an advantage), as opposed to diverse object movement.

## 8.8.1. Ablations

**Forward objective and tuning Lambda** Visual forward prediction $L_{fdm}$ plays an important role when it is used in the right proportion. Our intrinsic reward is a weighted combination of cross-modal prediction and forward prediction as defined in Eq.8.4.3. Figure 8.6 illustrates the model behavior with different levels of emphasis on the forward loss term, with $\lambda$ uniformly sampled between 0 and 1. Higher weights indicate that the future prediction dominates force/torque prediction. This leads to more object movement but lesser robot's

| Metric | Pushing | | | Open Door | | | Pick-up | | | Playing | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HaC | ICM+*haptics* | ICM | HaC | ICM+*haptics* | ICM | HaC | ICM+*haptics* | ICM | HaC | ICM+*haptics* | ICM |
| **Exploration ↑** | **0.403** | 0.291 | 0.187 | **0.669** | 0.355 | 0.083 | **0.063** | 0.051 | 0.013 | - | - | - |
| **Success ↑** | **0.733** | 0.678 | 0.597 | **0.983** | 0.571 | 0.114 | **0.891** | 0.825 | 0.780 | - | - | - |
| **Episode steps ↓.** | **57.84** | 87.61 | 95.24 | **23.34** | 97.10 | 199.3 | **30.54** | 33.77 | 42.19 | - | - | - |
| **Touch-interaction ↑** | **247.79** | 210.11 | 202.66 | **600.1** | 287.97 | 43.56 | 980.7 | **984.2** | 952.3 | **388.15** | 267.021 | 63.31 |

**Table 8.1. Haptics-based future prediction** Table compares the mean evaluations for HaC and ICM+*haptics* on all the four tasks emphasizing the importance of cross-modal association(see text).

.

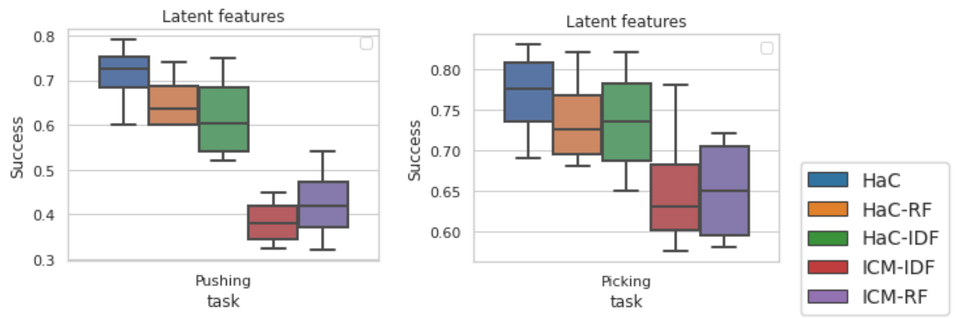


**Fig. 8.7.** Comparing different latent space performance on *pushing* and *grasping* tasks. Features learned via haptics prediction perform better than those learned using IDF or Random.

constant touch-interaction with the underlying objects. This is useful in tasks such as Pushing. A smaller $\lambda$ value leads to better inactive behavior which is handy in tasks such as grasping. In our experiments we choose an intermediate value (*e.g.* 0.5) that works best for all of our tasks.

**Haptics in future prediction.** The goal of this experiment is to strengthen the argument of cross-modal association. While conducting experiments, it is important to deduce information of one modality from another modality in a related manner than simply adding another modality on top of visual information. We created an additional baseline, ICM+*haptics*, where in addition to the visual prediction model we include the haptics-based future prediction model. We hypothesize ICM+*haptics* to perform better than ICM as it has additional information (haptics). The haptics-based future prediction model takes a touch vector as input and predicts the touch vector for the next time step. Table 8.1 compares HaC and ICM+*haptics* on MiniTouch tasks and we observe that ICM+*haptics* is better than ICM but compares below HaC. Table C.4.3 in the supplementary compares RND based baseline.

**Latent features for forward dynamics.** Choosing ideal embedding space for decoding the touch vector (Figure 8.2a) and for predicting future state (Figure. 8.2b) is important. Existing approaches rely on a pretext inverse dynamics (ID) task of predicting the agent's action given its current and next states [Pathak et al., 2017a]. Another simple yet strong

| Domain | Task | A3C | DDPG | PPO | SAC | MPO [Abdolmaleki et al., 2018] |
|---|---|---|---|---|---|---|
| Manipulator | Bring ball | $0.4 \pm 0.0$ | $0.6 \pm 0.1$ | $2.3 \pm 0.1$ | $1.5 \pm 0.3$ | $400 \pm 5.0$ |

**Table 8.2.** Performance on bring ball task of Manipulator domain using state space. scores outside the [800, 1000] range can be considered to be sub-optimal.

method is to use features from a random but fixed initialization of the encoder [Burda et al., 2019a]. In our work, we learn the features by leveraging the self-supervised pretext task of predicting one modality from the other. Figure 8.7 compares (1) encoder, i.e. learned through cross-modal prediction(HaC) (2) random feature encoder(HaC-RF) (3) encoder learned through ID task (HaC-IDF). In each case, the decoder network is optimized through touch prediction. We observe that the random features variant is stable and effective on both HaC and ICM models.

## 8.9. Dense Distance notion as Intrinsic Rewards

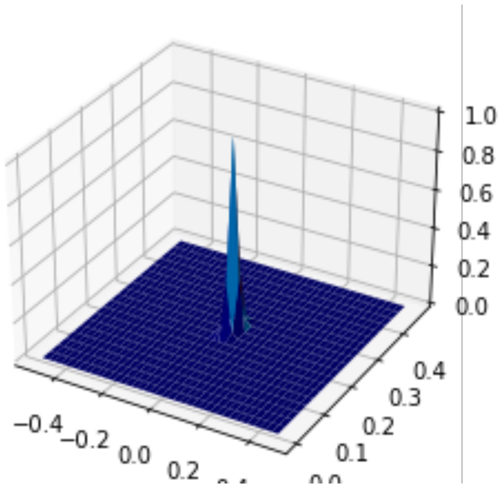
Manipulation domain in the DeepMind Control suite [Tassa et al., 2018] has been a challenging test bed of continuous control tasks. Most sophisticated RL agents that are trained in a fully supervised manner struggle to perform well on the contol tasks [Abdolmaleki et al., 2018]. A predominant hurdle causing the sub-optimal performance is crucially related to the sparse nature of the task reward function. Before delving more into the details, the following subsection outlines different tasks present in the manipulator domain.

### 8.9.1. Manipulator Domain and Tasks

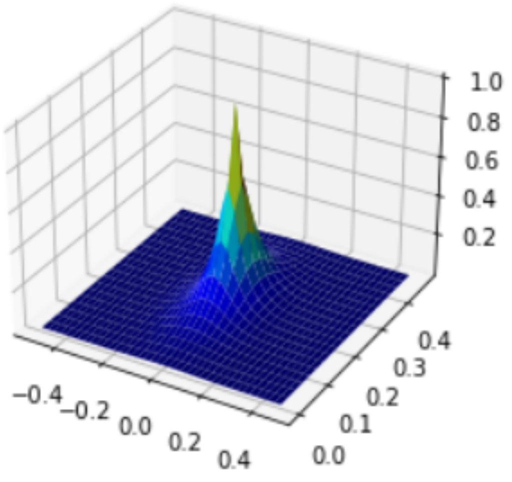
A Planar Manipulator domain is rewarded for bringing an object to a target location. It is composed following control tasks

- **Bring ball:** pick and bring the ball to a target location
- **Bring peg:** pick and bring the peg to the target peg in a relative alignment
- **Insert ball:**pick and place the ball inside the basket
- **insert peg:** pick and insert the peg into a slot

The above tasks are arranged in the incrasing order of difficulty. As one can imagine, these set of tasks are relatively harder exploration problems. For instance, solving the bring ball task requires the agent to first find and grasp the object at one part of the state space, and then bring it to a target slot in another part of the state space. Efficient exploration is the key to make progrss in such a setting. The performance of various agents on the easier bring ball task is shown in Table. 8.2. Clearly, most algorithms yeild sub-optimal performance on learning the task. A careful visualization of the sparse reward function for the bring ball task is shown in Figure. 8.8a. We hypothesize that a dense reward metric could help provide stronger gradient signal to make progress.



**(a)**



**(b)**

**Fig. 8.8.** (a)Figure depicting the sparse reward function for the manipulator task as a heat map over the $x - y$ co-ordinates of a 2D grid. (b) Shows the dense intrinsic reward.

To begin, we evaluate our method on the bring ball task. Although HaC exhibited reasonable success on the Minitouch benchmark tasks, it does not achieve optimal performance on the Manipulator domain of DeepMind control suite benchmark. We find that the approach do not always provide the same benefits for the task generalization in bring ball task for instance. This indicates that transfer remains a challenge and may require different approach in addition to curiosity based formulation for more challenging exploration setting. Towards this, we find that augmenting curiosity based exploration with a dense distance notion between the object and target location helps boost the performance. Figure. 8.8b shows a smoother version of the reward function compared to Figure. 8.8a. The behaviour policy essentially maximizes this intrinsic reward instead of the sparse extrinsic reward. Figure 8.9 depicts the improved performance when using dense intrinsic reward alongside HaC

While the qualitative performance on all the three tasks are provided in Figure 8.10, the quantitative results are show in the Figure 8.11 plots. As anticipated, the bring ball task is relatively easier to solve compared to the insert ball and bring peg tasks. The results are show for the state based inputs. For the case of pixel inputs, the tasks is even more challenging. Since our intrinsic reward computes a dense distance notion between the object and target, we therefore need to extract the object features from pixels. In this setting, we need to identify the object location form pixels to calculate the intrinsic dense reward. There have been many vision based self-supervised models that aim to obtain object key-points from pixels, *e.g.* from the videos of people moving etc. Transporter model [Kulkarni et al., 2019] infers persistent key-points more geometric state spaces rather than just pixels. Their formulation leverages a prediction problem, i.e. given two frames x an x' ( where x' differs from x by objects that moved over the frame gap), the model is asked to reconstruct x'

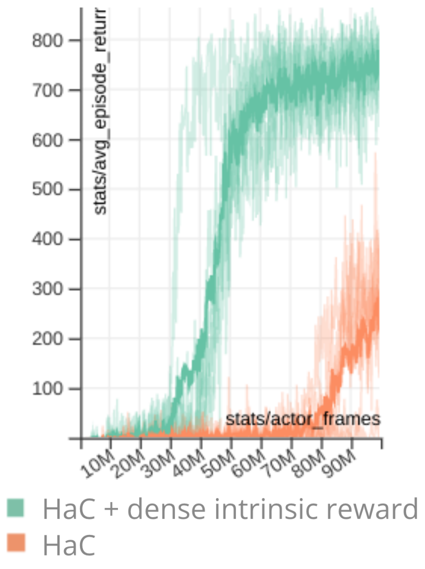
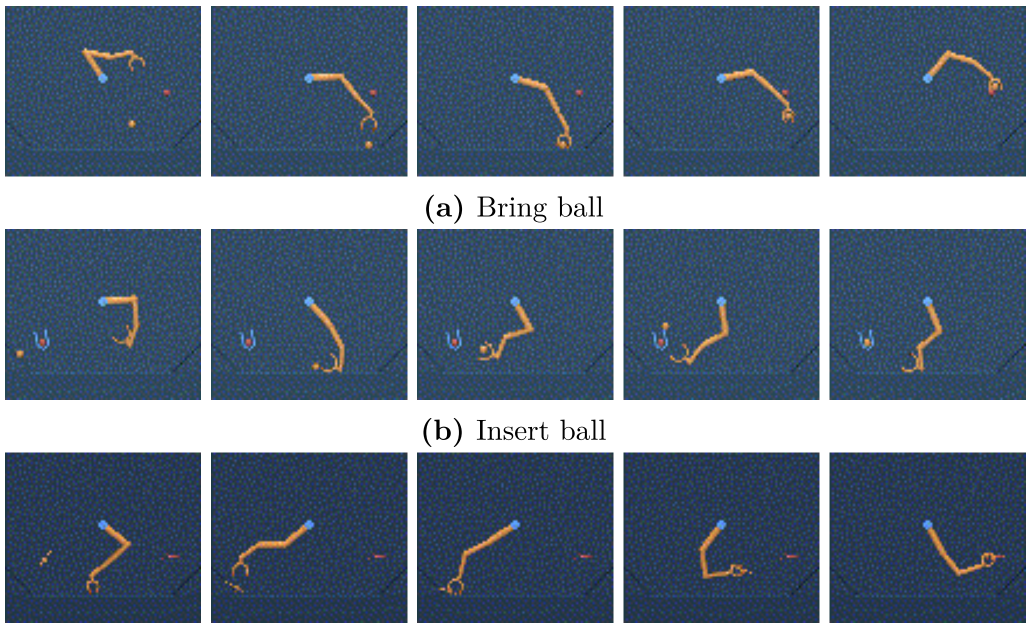


**Fig. 8.9.** Performance with and without the dense intrinsic reward on bring-ball task.



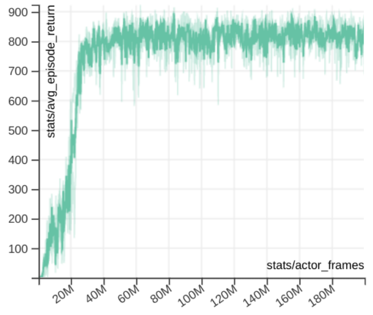
**(a)** Bring ball

**(b)** Insert ball

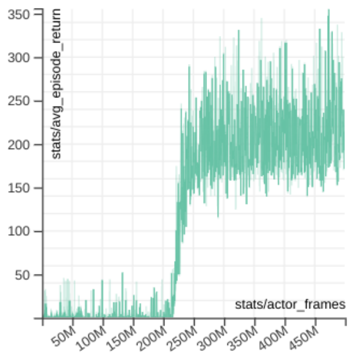
**(c)** Bring peg

**Fig. 8.10. Qualitative Evaluations**. Each row corresponds to the three manipulator tasks and each columns depicts few steps in to the episode of an agent trained using the intrinsic reward.

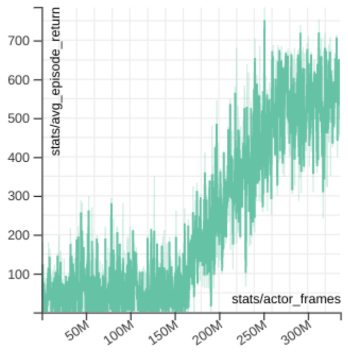
through the inductive bias of learning the dynamic key-points. In our work, we train the transporter model on a random policy, and use the trained model to infer the object location to compute our intrinsic distance reward. The resulting appraoch was able to solve the task albeit high sample complexity as shown in Figure 8.12.



**(a)** Bring ball



**(b)** Bring Peg



**(c)** Insert Ball

**Fig. 8.11.** Quantitative evaluations on the three manipulatpr tasks using state based input. Note that the bring-peg and insert-ball tasks are order of the magnitude challenging than the bring-ball task.

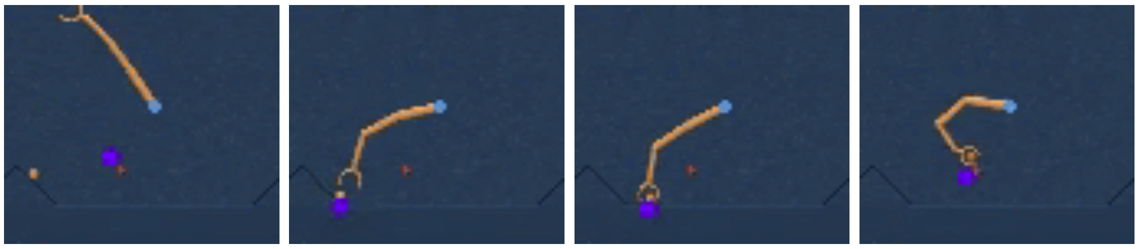


**Fig. 8.12. Qualitative Evaluations**. Model leverages object centric representation model to learn the object keypoint. The inferred object location is used to compute the intrinsic reward inorder to solve the bring ball task.

## 8.10. Concluding remarks

We formulated Haptics-based Curiosity (HaC) aimed at encouraging exploration via haptic interaction with the environment. We demonstrated in this work, that by involving additional modalities, the performance of curiosity-based systems on downstream tasks can be increased. We observed increased interaction with target objects, and presented evidence that HaC learns to solve the MiniTouch benchmark tasks in an efficient manner while vanilla RL algorithms and vision-based curiosity formulations struggled. Force/torque sensing is widespread in the lab and industrial robots but while there are plenty of robotic benchmarks, we believe that tactile feedback is an under-explored modality and by releasing our benchmark, we hope to enable future research in this exciting area.

# Chapter 9

# Prologue to Article 4

## 9.1. Article Details

**Unsupervised Model-based Pre-training for Data-efficient Reinforcement Learning from Pixels**. Sai Rajeswar*, Pietro Mazzaglia*, Tim Verbelen, Alexandre Piché,Bart Dhoedt, Aaron Courville, Alexandre Lacoste. This work will be presented at Decision Awareness in Reinforcement Learning workshop, ICML 2022. Also, it is submitted to International Conference on Robot Learning (CoRL), 2022. (First two authors contributed equally)

*Contribution* This project is a follow-up to the previous article aimed at building generalist RL agents. The efforts took shape from the observations highlighted in the unsupervised RL benchmark. It suggests that none of the tested unsupervised RL algorithms completely solve the benchmark and that a large gap in performance was found between using state-based inputs versus high-dimensional pixel-based inputs. The core idea of the model-based approach is conceived by the author of the dissertation. This was refined over regular meetings with Alexandre Lacoste and Pietro Mazzaglia. The author and Pietro jointly prepared the manuscript.

## 9.2. Context

Inspired from how natural agents acquire skills without supervision and efficiently apply to a variety of tasks, unsupervised reinforcement learning proposes to collect data through self-supervised interaction to accelerate task-specific adaptation. However, it is debatable if the current unsupervised approaches can adapt and generalize fast when the input observations are high-dimensional Images. In the previous chapter, for example, we observed that the DM Control suite tasks takes hundreds of millions of frames to reach optimal performance. In this project, we advance the field by closing the performance gap in the Unsupervised

Reinforcement Learning Benchmark, a collection of control tasks to be solved in a data-efficient manner, after interacting with the environment in a self-supervised way.

## 9.3. Research Impact

Intrinsic control and world models has been envisioned by LeCun [2022] as a crucial combination for building Autonomous machine intelligence. In our approach, we intrinsically explore and plan via an actor-critic framework and a world model (with a GRU) learnt using the perception from a posterior encoder. Many recent works demonstrated the use of model learning for autonomous visual control [Wu et al., 2022, Seo et al., 2022]. We, on the other hand, aim at building generalist agents that can adapt to tasks in a fast data-efficient manner. The work gives rise to an approach that solves the unsupervised RL benchmark [Laskin et al., 2021] by achieving **98.5±3.7%** of a supervised RL agent's overall performance. We empirically show that algorithms that learn a world model through self-supervised exploration can significantly improve their performance, compared to model-free approaches. Furthermore, we propose novel evaluation strategies to be adopted by the community that assess the quality of unsupervised model-based exploration approaches for the first time.

**Note**: This paper is presented as-is, with minor cosmetic changes to adhere to the Universite de Montreal thesis template. A modified version of the pre-print is currently under review at International Conference on Robot Learning (CoRL), 2022.

# Chapter 10

# Article 4: Unsupervised Model-based Pre-training for Data-efficient Reinforcement Learning from Pixels

## Abstract

Controlling robots from raw sensory data is an arduous task, especially using vision as the main sensory input. Reinforcement learning (RL) algorithms can learn complex behaviors but require large amounts of interactions between the agent and the environment. To alleviate this, unsupervised RL proposes to employ self-supervised interaction and learning, to adapt faster to future tasks. Yet whether current unsupervised strategies improve generalization capabilities is still unclear, more so when the input observations are high-dimensional. In this work, we present an unsupervised model-based strategy that enables data-efficient adaptation in visual control environments. Our approach shows improved performance on the Unsupervised RL Benchmark, where it matches the results of supervised methods by using 20x less task-specific data. Moreover, it shows significant improvement, when tested on the Real-Word RL benchmark, showing that the method is robust and its advantages are likely to translate to real robotic setups. We extensively evaluate our work, comparing several exploration methods and improving fine-tuning by studying the interaction between the model components. We further investigate the limits of the learned model and of the unsupervised methods, to gain insights into how these influence the decision process, shedding light on new research directions.

## 10.1. Introduction

Modern successes of Reinforcement Learning (RL) have shown promising results for robotics applications Levine et al. [2016b], OpenAI et al. [2019], Kalashnikov et al. [2018], Lu et al. [2021], Lee et al. [2021]. However, training an agent for each task individually

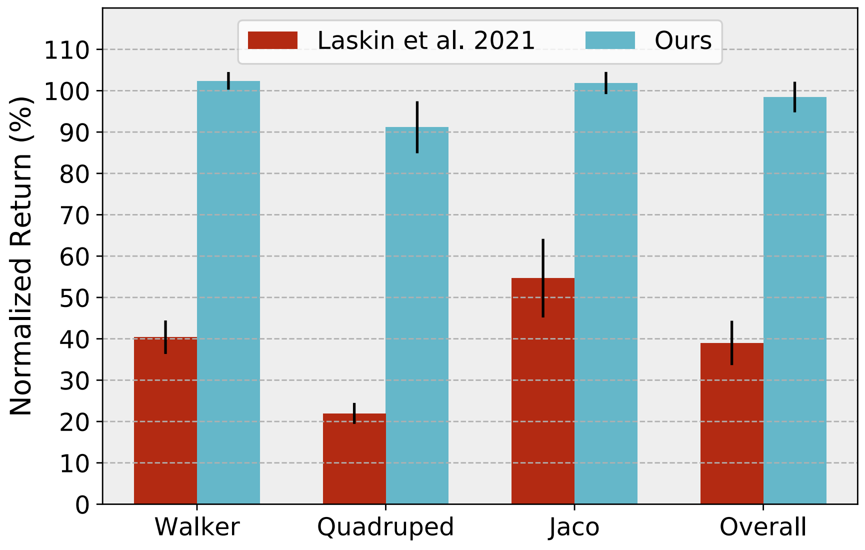


**Fig. 10.1. Progress on the URL benchmark from pixels.** Comparison of the overall best performing approach from the URLB paper, i.e. Disagreement [Pathak et al., 2019b] (**39.0±5.4%**), with our best performing approach (**98.5±3.7%**), fine-tuning for 100k steps pre-trained models that have been trained in an unsupervised way for 2M steps. Returns are normalized using the scores of supervised RL agents (details in Appendix).

requires a large amount of task-specific environment interactions, incurring huge redundancy and prolonged human supervision. On the other hand, training agents that can generalize quickly on more than a single task is desirable towards building intelligent autonomous systems. Developing algorithms that can efficiently adapt and generalize to new tasks has hence become an active area of research in the RL community.

In computer vision and natural language processing, unsupervised learning strategies have enabled learning models without supervision to reduce sample complexity on downstream tasks [Chen et al., 2020b, Radford et al., 2019]. In a similar fashion, unsupervised RL (URL) aims to learn about the environment using self-supervised exploration without the need for a reward function [Pathak et al., 2017b, Burda et al., 2019a, Bellemare et al., 2016]. The learned modules are then adapted to downstream tasks with the aim of reducing the amount of interactions required with the environment by orders of magnitude. This paradigm opens opportunities in robotics [Chebotar et al., 2021, Sharma et al., 2020], as one can consider pre-training in simulation or using a distributed collection of robots. Next, assuming a successful URL algorithm, the robot can learn new tasks with limited supervision or even in a zero-shot fashion, if the reward function is properly communicated to the agent.

Recently, the Unsupervised RL Benchmark (URLB) [Laskin et al., 2021] established a common protocol to compare self-supervised algorithms across several robotics tasks from the DM Control Suite [Tassa et al., 2018]. In the benchmark, an agent is allowed a task-agnostic pre-training stage, where it can interact with the environment in an unsupervised manner, followed by a fine-tuning stage where, given a limited budget of interactions with the environment, the agent should quickly adapt for a specific task. However, the results obtained by Laskin et al. [2021] suggest that no unsupervised RL algorithm was close to solve the benchmark, as agents pre-trained with unsupervised RL did not perform far better than randomly initializated agents. The limitations of the approaches emerged further when using high-dimensional camera inputs instead of low-dimensional sensor inputs, making it hard to believe that current approaches could scale to more realistic environments, where the use of vision is essential to provide the robots awareness about the workspace and/or their surroundings.

In this work, we present an unsupervised model-based RL approach that significantly improves data-efficiency when fine-tuned in a low-data regime, by planning and adapting task-specific policies on the synthetic data generated by the unsupervised pre-trained model. Our approach performs significantly better than previous approaches on the URL benchmark from pixels (Figure 10.1), nearly achieving the asymptotic performance of supervised RL agents, trained with 20x more task-specific data, and bridging the gap with low-dimensional sensor inputs [Laskin et al., 2021]. To test robustness to the challenging intricacies of realistic setups, we test our approach on vision-based variants of tasks from the Real-World RL benchmark [Dulac-Arnold et al., 2020a, 2019]. We show that leveraging unsupervised model-based pre-training significantly improves performance during the adaptation phase, even when the target environment where the agent is deployed presents model perturbations, noise and delays in the system.

Our efforts are aimed at improving and scaling unsupervised RL algorithms to operate efficiently in visual robotic control settings. Our contributions can be summarized as: *(i)* the design of a class of unsupervised model-based RL approaches, which enable fast adaptation after an unsupervised pre-training stage, *(ii)* a study of the interplays between the pre-trained modules that allow to improve sample efficiency during the fine-tuning stage, *(iii)* an empirical investigation on whether the improvements brought by our method would translate to more realistic robotics setups, *(iv)* an analysis of the models learned through unsupervised interaction with the environment, aimed at understanding what aspects could be improved to facilitate fast adaptation.

We demonstrate through experimentation that, following our approach, it is possible to bridge the performance gap between state-based and pixel-based inputs, and to achieve the asymptotic performance of supervised RL agents (Figure 10.1). An extensive empirical evaluation, supported by more than 2k experiments, among main results, analysis and ablations, was used to refine the components of our method. We hope that our large-scale evaluation will inform future research towards developing and deploying pre-trained agents that can be adapted with considerably less data to real-world robotics tasks, as it has happened with vision Parisi et al. [2022] and language [Ahn et al., 2022] unsupervised pre-trained models.

## 10.2. Preliminaries

The RL setting can be formalized as a Markov Decision Process (MDP), denoted with the tuple $\{\mathcal{S}, \mathcal{A}, T, R, \gamma\}$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $T$ is the state transition dynamics, $R$ is the reward function, and $\gamma$ is a discount factor. The objective of an RL agent is to maximize the expected discounted sum of rewards over time for a given task, also called return, and indicated as $G_t = \sum_{k=t+1}^{T} \gamma^{(k-t-1)} r_k$. In continuous action settings, one popular approach to predict the most rewarding actions is to combine a model that learns to output the best action given a certain state, referred to as the actor model, and a model that learns to estimate the expected value of the actor's actions over time, given a certain state, referred to as the critic model. Actor-critic algorithms can be combined with the expressiveness of neural network models to solve complex continuous control tasks [Haarnoja et al., 2018, Lillicrap et al., 2016, Schulman et al., 2017].

In this work, we investigate the problem of fast adaptation for a downstream task, after a phase of unsupervised training and interaction with the environment. We adopt the URL benchmark, which consists of three control domains, *Walker*, *Quadruped* and *Jaco*, and twelve tasks, four per each domain. Consistently with URLB [Laskin et al., 2021], our experimental procedure is made of two phases: a pre-training (PT) phase, where the agent can interact with a task-agnostic version of the environment for up to $2M$ steps, and a fine-tuning phase (FT), where the agent interacts with the same environment, being provided a task to solve and a limited budget of $100k$ steps. During the PT phase, rewards are removed from the environment. Sensible information about the environment can be obtained by exploring the domain-dependent dynamics, which will remain unchanged in the downstream tasks. During FT, the agent receives task-specific rewards when interacting with the environment. As the agent has no prior knowledge of the task, it should both understand the task and solve it efficiently, in the limited budget.

Crucially, we focus on the pixel-based setup of URLB, where the environment is perceived by the agent only through images. In this setting, the performance of several exploration strategies, combined with a state-of-the-art model-free approach [Yarats et al., 2022], were shown to lack behind the asymptotic performance of an RL agent trained on the downstream task, as reported in [Laskin et al., 2021] and Figure 10.1. We believe one of the causes of this is that model-free RL algorithms cannot successfully leverage the information observed about the environment dynamics during PT, as they rely uniquely on actor and critic's predictions. To overcome this limitation, we ground our work on a model-based RL agent, whose learned model should allow preserving important information about the environment.

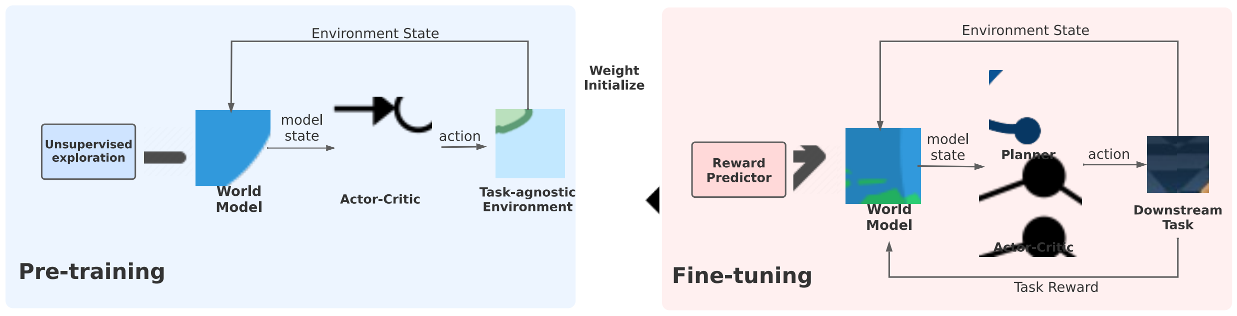


**Fig. 10.2. Approach Overview.** The unsupervised benchmark consists of pre-training (PT) and fine-tuning (FT) stages. During pre-training, the agent interacts with the environment through an unsupervised RL strategy, maximizing an intrinsic reward function, and concurrently training a world model on the data collected. During fine-tuning, the agent exploits the world-model learned to efficiently plan and adapt for different downstream tasks, where it receives rewards from the environment to maximize.

## 10.3. Approach

In order to perform well on the URL benchmark, it is important that an agent: (i) meaningfully interacts with the environment during the PT phase, to discover useful transitions; (ii) successfully reuses the modules learned during PT for fast adaptation; and (iii) efficiently employs the FT phase to understand and master the downstream task. In this section, we expand on how we addressed these challenges, giving rise to an approach that nearly solves the benchmark by achieving **98.5±4.7%** of a supervised RL agent's overall performance (Figure 10.1). An overview of the end-to-end approach is illustrated in Figure 10.2 and a detailed algorithm is presented in Appendix D.3 for reference.

### 10.3.1. Model-based Agent

We build our model-based agent upon DreamerV2 [Hafner et al., 2021], whose agent attempts to learn a world model [Ha and Schmidhuber, 2018, Hafner et al., 2019b, 2021] that allows predicting the outcomes of future actions in the environment. The environment dynamics is captured into a latent space $\mathcal{Z}$, which allows a compact representation of the high-dimensional inputs of the agent. The world model consists of the following components:

$$\begin{aligned}
&\text{Encoder:} && e_t = f_\phi(s_t),\\
&\text{Dynamics:} && p_\phi(z_t|z_{t-1}, a_{t-1}),\\
&\text{Posterior:} && q_\phi(z_t|z_{t-1}, a_{t-1}, e_t),\\
&\text{Image Decoder:} && p_\phi(s_t|z_t),\\
&\text{Reward Predictor:} && p_\phi(r_t|z_t).
\end{aligned}$$

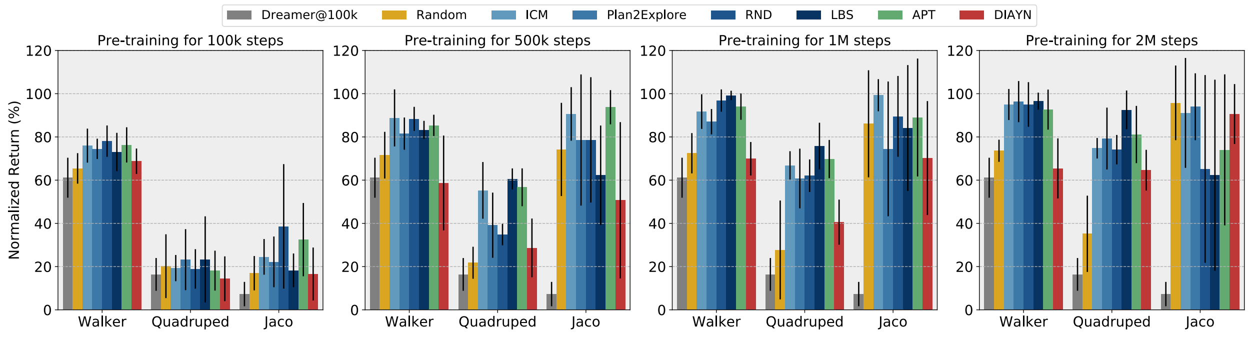


**Fig. 10.3. Model-based URLB.** We combined multiple unsupervised RL approaches with the model-based Dreamer agent and present the performance across the different domains of URLB, after 100k steps of fine-tuning. Each plot presents result for snapshots taken after a certain number of pre-training, 100k, 500k, 1M and 2M steps, from left to right.

The model states $z_t$ have both a deterministic component, modeled using the recurrent state of a GRU [Chung et al., 2014], and a (discrete) stochastic component. The encoder and decoder are convolutional neural networks (CNNs) and the remaining components are multi-layer perceptrons (MLPs). The world model is trained end-to-end by optimizing an evidence lower bound (ELBO) on the log-likelihood of the data collected in the environment [Hafner et al., 2019b,a].

In order to plan actions, the agent learns latent actor and critic networks:

$$\text{Actor:} \quad \pi_\theta(a_t|z_t), \qquad\qquad \text{Critic:} \quad v_\psi(z_t).$$

The actor is used to generate actions, given the model state, while the critic estimates the expected return for a certain model state, when following the actor's actions. Both components are trained online within the world model, by imagining the model state outcomes of the actions produced by the actor, using the model dynamics. Rewards for imagined trajectories are provided by the reward predictor and combined with the critic predictions to produce a GAE-$\lambda$ estimate of the returns [Schulman et al., 2016]. The actor maximizes such an estimate of the returns, backpropagating its gradients through the model dynamics.

For the encoder and the decoder networks, we used the same architecture as in Hafner et al. [2021]. The hyperparameters for the agent, which we keep fixed across all domains and tasks, can be found in Appendix D.4.

### 10.3.2. Unsupervised Pre-training

In the PT stage, different unsupervised RL strategies can be used to explore the environment and train the components of the agent. The resulting networks are then used to initialize respective components in the agent deployed for the downstream task, aiming to reduce sample complexity during FT.

As we employ a model-based agent, we use the experience collected to train the agent's world model, along with the actor and the critic networks. We note that, during PT, the reward predictor of the world model is either unused or used to predict intrinsic rewards, according to the unsupervised approach employed, as reward information should not be available to the agent. During FT, the reward predictor is trained to predict the downstream task rewards.

Unsupervised RL methods can be grouped in three categories: knowledge-based, data-based and competence-based [Schmidhuber, 2010, Laskin et al., 2021]. We study multiple approaches, focusing primarily on knowledge-based methods as these combine well with the model-based nature of our agent, and implement LBS [Mazzaglia et al., 2021], ICM [Pathak et al., 2017b], RND [Burda et al., 2019b], and Plan2Explore [Sekar et al., 2020]. As a data-based approach, we choose APT [Liu and Abbeel, 2021b], and as a competence-based approach, we use DIAYN [Eysenbach et al., 2019]. Finally, we add a Random action baseline, as a maximum entropy approach [Haarnoja et al., 2018]. Details on these methods and how we combined them with our model-based agent are discussed in Appendix D.2.

### 10.3.3. Fine-tuning for Downstream Tasks

During the unsupervised PT phase, the agent collects experience from the environment that is used to train several components: a task-agnostic world model (without the reward predictor), an actor and a critic network. Moving to the FT phase, the pre-trained weights of these components can be copied into a new instance of the model, aiming to leverage previous experience for faster adaptation.

Since the domain dynamics stays the same between the PT and FT phases, initializing the world model with the pre-trained one should facilitate adaptation. However, the reward is changing from *pseudo-reward* to task reward when changing from the PT phase to FT phase. Hence, it is not clear if pre-training of the actor and critic can help for the downstream task. To shed light on this question, we conduct experiments in Section 10.4 to determine if it useful to transfer the actor and the critic. Unless specified, for our *Default* FT model, which we refer to as (w/ model, w/ actor, w/o critic), we copy the weights of the pre-trained world model and actor but initialize the critic from scratch.

### 10.3.4. Model-based Planning

As we train a latent world model, we can exploit model-based planning to adapt with limited additional environment interaction. When an accurate model of the environment is available, traditional model-based control approaches, such as Model Predictive Control (MPC) [Williams et al., 2015, Chua et al., 2018, Richards, 2005], can be used to plan the agent's action. Nonetheless, using an actor and a critic has several advantages, such as amortizing the cost of planning by caching previously computed (sub)optimal actions and amortizing the cost of computing long-term returns from a certain state, without the need to imagine outcomes that are far in the future. We found it useful to adopt a hybrid planning strategy, which exploits the actor and critic's predictions as well as an evolutionary sampling strategy based on the Cross-Entropy Method (CEM; Rubinstein and Kroese [2004]).

## 10.4. Experiments and Analysis

In all the experiments, the results show average normalized returns with error bars showing the standard deviation. To normalize results in a comparable way for all tasks, we train a fully-supervised agent with 2M steps per each task. We use the mean performance of this agent, which we refer to as "oracle", as the reference scores to normalize results in the plots (details in Appendix D.1). For all experiments, results are presented with at least three random seeds.

### 10.4.1. Model-based URLB.

The results of the different exploration approaches are shown in Figure 10.3. Results are presented by taking snapshots of the agent at different times during training, i.e. 100k, 500k, 1M and 2M steps, and fine-tuning the pre-trained policies and models for 100k steps. As opposed to the model-free experiments in URLB, where they fine-tuned the actor and the critic, we found that leveraging a pre-trained world model during fine-tuning dramatically improves the performance. Simply using random actions for unsupervised exploration already increases performance compared to a supervised agent trained from scratch for 100k steps (*Dreamer@100k*). This is in contrast to the results in [Laskin et al., 2021], where the improvements when exploiting pre-training were less significant.

Unsupervised exploration strategies generally lead to higher performance in complex tasks, with performance that increases over time in the Walker and Quadruped tasks. We note that the performance in the Jaco domain is less consistent over time and tends to have higher variance. We hypothesize that this is due to a greater discrepancy between pseudo-reward and task-reward for this environment. DIAYN underperforms compared to the other methods, providing some support to the observation in [Laskin et al., 2021], claiming that current competence-based approaches tend to perform worse than the rest.

**Exploiting Pre-Trained Modules.** The results of the ablation studies on fine-tuning different sets of pre-trained modules are presented in Figure 10.4, averaging across all unsupervised RL methods. Overall, the default configuration, which reuses the weights of the world model and the actor performs best. Initializing the agent with the pre-trained actor is particularly useful in the Walker and Quadruped domains, but it is harmful in the Jaco tasks. A possible explanation for this could be that the exploration actor that is transferred from pre-training might have a precise explorative goal when brought to the fine-tuning stage, which may be particularly far from the target state of the task. In the Walker and Quadruped tasks, where the rewards are denser, exploring multiple states, even far from the downstream task behavior, provides useful reward information to exploit for the task. On the other hand, for the Jaco sparser tasks, if the exploration is initialized to reach a point that is too far from the reaching target, it might become arduous to encounter useful rewarding states within the reduced budget of FT. This hypothesis is supported by the fact that the random action methods and the DIAYN approach, which are believed to explore less than the others (thus, staying closer to the initial agent's position), perform well, and in some cases even better, than the other approaches (Figure 10.3).

Initializing the critic with the pre-trained one has little impact in Walker and Quadruped but has been problematic in the Jaco domain, likely because of the more sparse rewards of the Jaco tasks, which are very different from the dense pseudo-rewards used to pre-train the critic. For this reason, we default to not reusing the critic's weights. Finally, the world model is confirmed to be the most valuable component to initialize, as, independently from the other components, initializing the world model brings a significant improvement in performance. Detailed results per each method are available in Appendix D.5.

**Leveraging Planning.** In order to better exploit the pre-trained model during FT, while also leveraging the advantages of using an RL-like learning mechanism, we employ the recently proposed TD-MPC approach [Hansen et al., 2022], a hybrid planning strategy that combines MPC and temporal difference actor-critic learning. TD-MPC claims improved performance, thanks to the combination of short-term planning using MPC and long-term predictions,

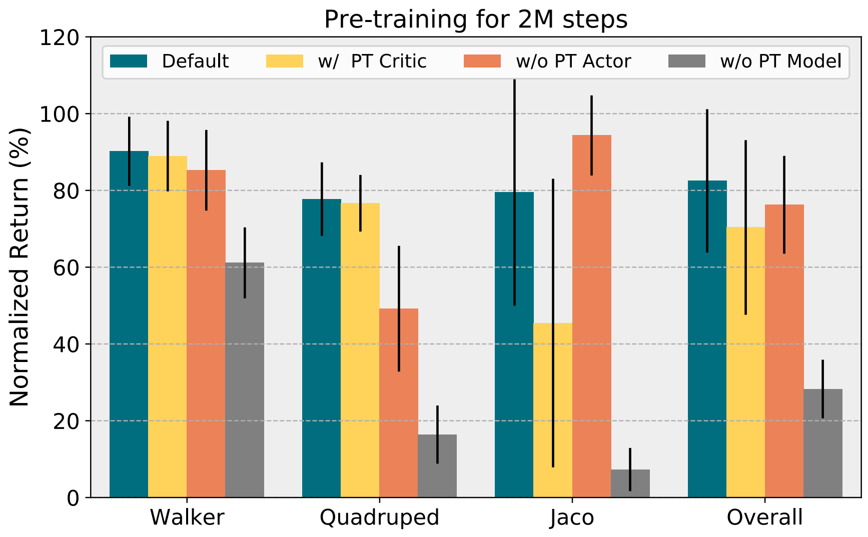


**Fig. 10.4. Exploiting Pre-Trained Modules.** Comparison of the results when fine-tuning different pre-trained components of the agent. Results are averaged across all unsupervised RL methods (2M steps pre-training).

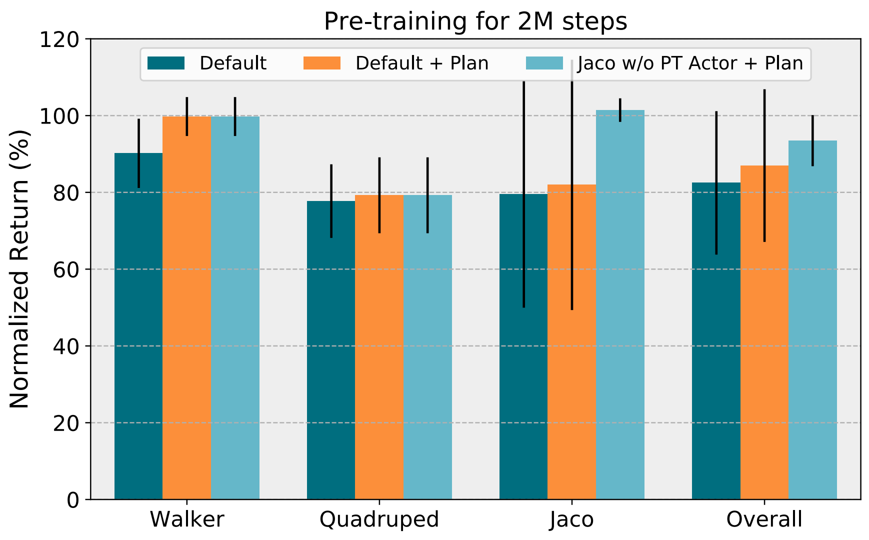


**Fig. 10.5. Leveraging Planning.** Improved results on the benchmark, obtained by combining actor-critic and MPC-like planning. Results are averaged across all unsupervised RL methods (2M steps pre-training).

using the actor-critic architecture. We implement TD-MPC on top of our models and compare the performance with the standard Dreamer actor-critic learning in Figure 10.5. We observe that TD-MPC slightly improves the performance in all domains. Combining the improved planning strategy with the previous insight of not initializing the weights of the actor in the Jaco domain, we obtain our overall best performance (*TD-MPC (Jaco w/o PT actor)*), with the LBS-based model being the overall best performing method (**98.5±3.7%**; Figure 10.1). Detailed results for each method are available in Appendix.

## 10.4.2. Real-World Reinforcement Learning Benchmark

Algorithms developed in simulation struggle to transfer to real-world systems due to a series of implicit assumptions that are rarely satisfied in real environments. The RWRL benchmark [Dulac-Arnold et al., 2020a] considers several challenges that are common in real-world systems and implements them on top of DM Control Suite tasks. Approaches that address the challenges proposed in the benchmark are more likely ready to be deployed for real world problems, such as robotics settings.

We employ vision-based variants of the *Walker Walk* and *Quadruped Walk* tasks from the RWRL benchmark. Compared to the URLB setting, these tasks introduce system delays, stochasticity, and perturbations of the robot's model and sensors, which are applied with three degrees of intensity to the original environment, i.e. 'easy', 'medium' and 'hard' settings (details in Appendix D.6). We employ the RWRL benchmark to answer the questions: *(i)* is unsupervised PT beneficial when transferring to a more realistic target environment for downstream tasks? *(ii)* does unsupervised exploration brings and advantage compared to

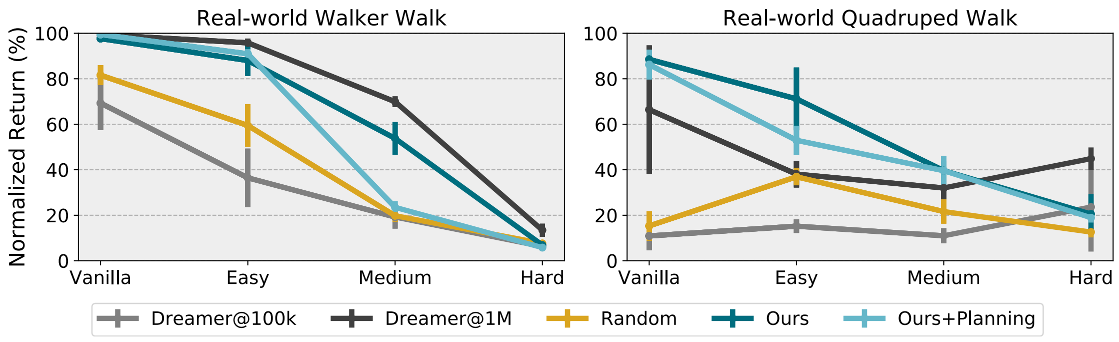


**Fig. 10.6. Results on RWRL.** We compare our method to random exploration and training from scratch on the tasks from the RWRL benchmark. Models are pre-trained on the vanilla version of the environment for 2M steps and fine-tuned for 100k steps on the perturbated tasks from RWRL. Normalization scores are provided in Appendix.

random data PT? *(iii)* can hybrid planning be employed to improve performance, when the environment is stochastic/non-stationary?

We present results of our method, with and without the hybrid planner, and compare to random exploration and training from scratch (Dreamer@100k), in Figure 10.6. Crucially, the PT models are trained in the standard task-agnostic version of the environments from the DM Control Suite, so that the results highlight the extent to which models trained in ideal conditions generalize to real-world settings, when fine-tuned in a low-data regime.

Overall, we found that PT offers an advantage over training from scratch, despite all the variations in the environment, although the advantage decreases with increasing intensity of the perturbations. Our approach, pre-trained using LBS during unsupervised PT, strongly outperforms the random exploration baseline in the 'easy' and 'medium' settings, showing that learning a better model yields better FT performance, even when the dynamics of the downstream task is affected by misspecifications and noisy factors. Finally, in contrast with the finding on URLB, using a typical RL actor-critic works better than an hybrid planner. We believe this is because the model's predictions are less certain and precise in this setting and thus cannot inform the short-term MPC planner accurately.

## 10.5. Extended Analysis

First, we showed how we managed to bridge the performance gap in the URL benchmark by leveraging model-based unsupervised RL, improving the exploitation of the different agent's components during FT, and leveraging planning to improve data efficiency. Now, we focus on providing an analysis of the world model learned during the unsupervised pre-training stage, aiming to gain insights on which aspects improve decision awareness in the URLB setting, i.e. how can we best exploit the two stages of training to improve the agent's decision making.

To shed light on this, we analyze discrepancies in the model's dynamics and in the reward prediction process that is leveraged during the fine-tuning planning.

## 10.5.1. Zero-shot Performance.

How useful is the model learned during the unsupervised pre-training stage? To gain insights into this matter, we perform some additional tests where we provide the FT agent with a pre-trained reward predictor, that we train on the data collected during PT, separately from the agent. Given such a reward predictor, it should be possible to achieve high performance on the downstream tasks by simply planning within the model, e.g. performing MPC in a zero-shot setting. This assumes that the model correctly learned the dynamics of the environment and explored rewarding transitions that are relevant to the downstream task. In Figure 10.7, we compare the results of performing MPC in a zero-shot setting with the performance of an MPC agent that is allowed the typical 100k steps for fine-tuning. As for the MPC method, we employ Model Predictive Path Integral control (MPPI) [Williams et al., 2015]. Because MPC is particularly expensive to test, we just perform this experiment on top of the models trained with the Plan2Explore URL approach. We also plot the performance of a non-pre-trained model and of using an actor-critic planning strategy (also provided with the reward predictor since the beginning of fine-tuning), for comparison.

We observe that the performance of MPC (zero-shot) is generally weak. While it overall performs better than the non-pre-trained model, simply applying MPC leveraging only the

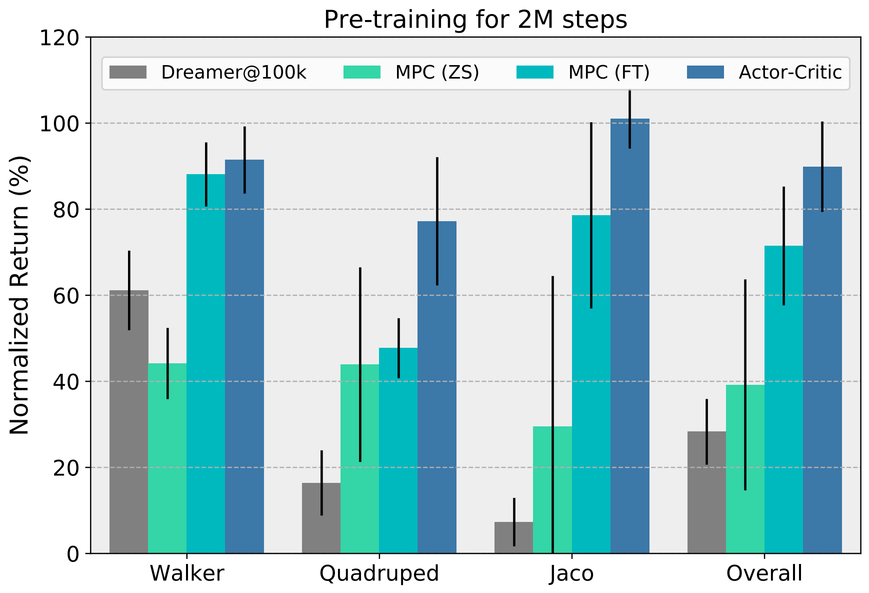


**Fig. 10.7. Model Predictive Control.** Exploiting a pre-trained reward predictor to test whether is there a gap between zero-shot (ZS) and fine-tuned (FT) MPC performance. Results refer to the Plan2Explore pre-trained agent (2M steps pre-training).

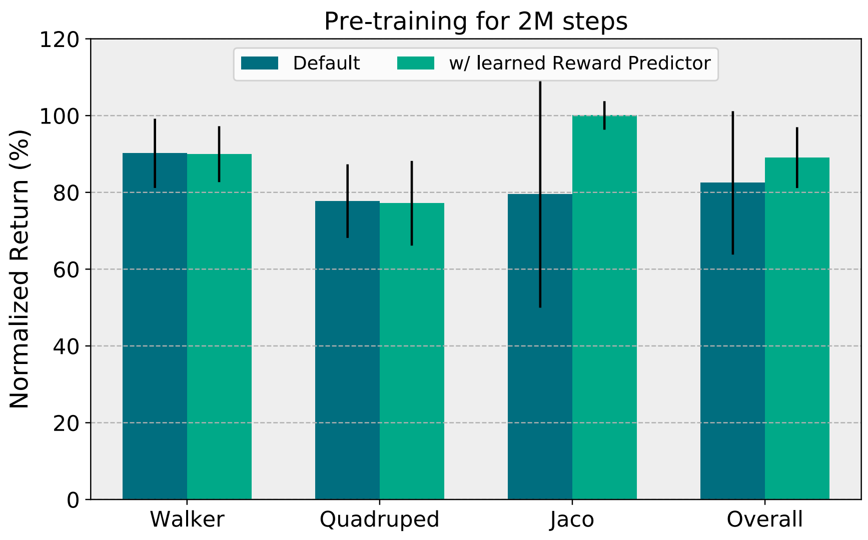


**Fig. 10.8. Model-based agent reward predictor ablation.** Evaluating performance when providing a task-specific reward predictor trained on the transitions collected during the unsupervised pre-training stage. Results are averaged across all unsupervised RL methods (2M steps pre-training).

pre-trained modules and the reward predictor trained on the PT stage data is not sufficient to guarantee satisfactory performance. The fact that exploiting the fine-tuning stage using the same MPC approach generally boosts performance demonstrates that the model has a major benefit from the fine-tuning stage. Still, the performance of MPC generally lacks behind the actor-critic performance, suggesting that, especially in a higher-dimensional action space such as the Quadruped one, amortizing the cost of planning with actor-critic seems crucial to achieve higher performance.

### 10.5.2. Learning the Reward Predictor.

Given that the agent strongly benefits from the 100k fine-tuning steps, we are interested in quantifying how much of this improvement is related to the necessity of learning a good reward function for the downstream task. In Figure 10.8, we measure the gap in performance between pre-trained agents that have no knowledge of the reward function at the beginning of fine-tuning and agents whose reward predictor is initialized from a reward predictor learned on top of the unsupervised pre-training data. Interestingly, the performance gap is overall small and irrelevant in the Quadruped and Walker domains. In the Jaco tasks, which have sparser reward functions, an a priori knowledge of the downstream task at the beginning of FT strongly improves performance.

According to our results, it is important that during the 100k steps of fine-tuning the actor is able to quickly obtain information about the downstream task. This might be easier for dense reward tasks, such as the Walker and Quadruped ones, but trickier in sparse settings

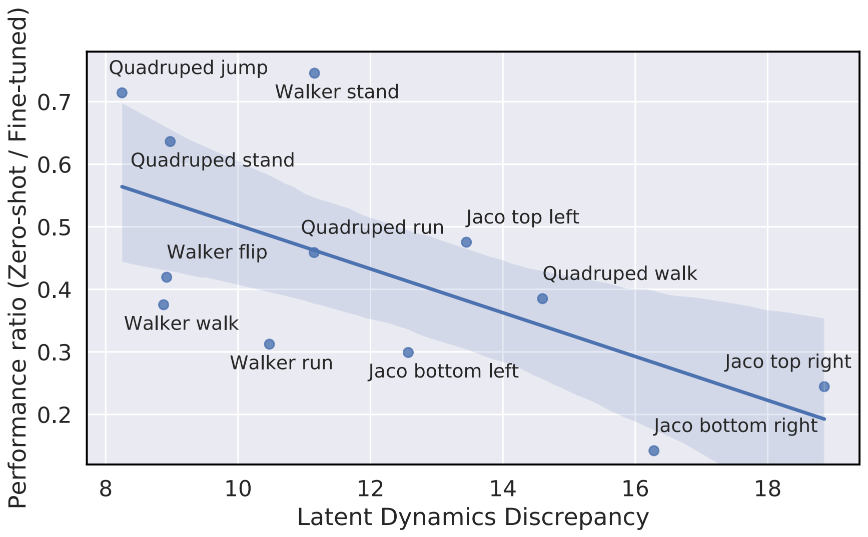


**Fig. 10.9. Correlation between latent dynamics discrepancy and performance.** Results refer to the Plan2Explore pre-trained agent. Most difficult tasks are highlighted.

like Jaco. As the initialization of the actor-critic modules also showed to be a compelling issue in our ablation study on the pre-trained modules exploitation (Figure 10.4), finding more efficient ways to pre-train/fine-tune the actor-critic modules could be an impactful research direction that will facilitate the adoption of unsupervised pre-training for RL.

### 10.5.3. Latent Dynamics Discrepancy.

A useful measure to assess the uncertainty or inaccuracy of a given model's dynamics is the model's misspecification, generally measured as the difference between the dynamics predictions and the real environment dynamics. When this metric is available, it is also possible to build robust RL strategies, that take the dynamics uncertainty into account while searching for the optimal behavior [Talvitie, 2018]. Dealing with pixel-based inputs, we observe the dynamics of the environment through high-dimensional images, which hinders the possibility to evaluate such a metric, as distances in pixel space can be misleading.

In our approach, we use a model-based RL agent that learns a model of the environment in a compact latent space $\mathcal{Z}$. In order to quantify the “misspecification" of the learned latent dynamics, we propose a new metric, which we call *Latent Dynamics Discrepancy*, that suits the setup of URLB. We aim to quantify the distance between the predictions of the pre-trained model and the same model after fine-tuning on a downstream task. However, as the decoder of the world model gets updated during fine-tuning, the latent space mapping between model states $z$ and environment states $s$ might drift. For this reason, we ran fine-tuning experiments where the agent's decoder weights are frozen, so that the decoder cannot be updated and the model can only improve the posterior and the dynamics. This ensures that the mapping

| Pre-training for 2M environment steps | | | | |
|---|---|---|---|---|
| | ICM | LBS | P2E | RND |
| Pearson Correlation | -0.54 | -0.60 | -0.34 | -0.03 |
| p-value | 0.07 | **0.04** | 0.28 | 0.91 |

**Table 10.1. Correlation between performance and intrinsic rewards.** Each column shows the Pearson correlation index and the p-value between fine-tuned performance across the URLB tasks and the intrinsic rewards computed on some oracle episodes.

$\mathcal{Z} \to \mathcal{S}$ remains unchanged and allows to compare the dynamics model after fine-tuning with the one before fine-tuning. In order to measure the distance between the distribution output by the dynamics network, we chose the symmetrical Jensen-Shannon divergence:

$$\mathbb{E}_{(z_t,a_t)}\Big[D_{\text{JS}}[p_{\text{FT}}(z_{t+1}|z_t,a_t)\|p_{\text{PT}}(z_{t+1}|z_t,a_t)]\Big], \tag{10.5.1}$$

where the expectation is taken over the previous model states $z_t$ sampled from the fine-tuned posterior $q_{\text{FT}}(z_t)$, actions $a_{t-1}$ sampled from an oracle actor $\pi^*(a_t|z_t)$, so that we evaluate the metric on optimal trajectories, whose environment's state distribution corresponds to the stationary distribution induced by the actor $s_t \sim d^{\pi^*}(s_t)$. We used 30 trajectories per task in our evaluation.

In Figure 10.9, we plot the correlation between our metric and the performance ratio between a zero-shot model and a fine-tuned model, where Plan2Explore was used for the 2M steps pre-training phase. We observed a strong negative Pearson correlation ($-0.62$), with a p-value of $0.03 < 0.05$, asserting that we must reject the null hypothesis, i.e. there exists a correlation between the two factors. This means that major updates in the model dynamics during fine-tuning played an important role in improving the agent's performance, compared to the pre-trained model and zero-shot performance. Future research may attempt to reduce such link, by either improving the model's learning process, so that the pre-trained dynamics could have greater accuracy, or the data collection process, proposing URL methods that directly aid to reduce such uncertainty.

### 10.5.4. Unsupervised Rewards and Performance.

How well can the unsupervised RL approaches we employed help improving adaptation further? To answer this question, we analyze the correlation between the normalized performance of the different agents and the intrinsic rewards they provide for optimal trajectories obtained by an oracle agent. A strong negative correlation between the two factors would indicate that the agent will be more interested in seeing the optimal trajectories when its performance is low on the task. We summarize the results of our analysis in Table 10.1.

We observe that there is negative correlation between Plan2Explore (P2E), ICM, LBS's performance and their intrinsic rewards, while we found ∼0 correlation for RND. In particular, the correlation for LBS, which overall performed best in the benchmark, has a statistical significance, as its p-value is $< 0.05$. Given such correlation, we believe the intrinsic rewards of LBS might be one of the causes of its outstanding performance. As LBS searches for transitions of the environment that are difficult to predict for its dynamics, the model likely learns those transitions more accurately, facilitating planning during the fine-tuning stage and eventually leading to higher performance (particularly in the most difficult domain, Quadruped). It is important that future work would consider learning a less ambiguous dynamics during the unsupervised RL phase, which can be efficiently leveraged by the agent for fine-tuning.

## 10.6. Related work

Our work lies at the intersection of unsupervised RL, model-based RL, and representation learning for RL. We discuss below the relevant literature in these three fields.

### 10.6.1. Model Based RL

In continuous control, model-based RL combined with powerful search methods has led to impressive results on a wide variety of tasks such as Atari [Schrittwieser et al., 2020] and continuous control [Hafner et al., 2019a]. In this work, we used the MPC approach MPPI, which is based on the Cross-Entropy Method (CEM Rubinstein and Kroese [2004]). These methods perform trajectory optimization by fitting a multivariate Gaussian distribution to the imagined future actions allowing them to search the space efficiently. Alternative search methods such as Monte Carlo Tree search [Coulom, 2006] and Sequential Monte Carlo planning [Piché et al., 2018] could also have been used without much change to the algorithm. Given that we do not have the reward information during pre-training, we base our model on the Dreamer architecture which reconstructs the future frames (and rewards, when available) to learn a transition model. This has the advantage of being simple and not requiring the task specification a priori. Task and reward awareness is necessary to learn a model to be self-consistent in latent space [Schrittwieser et al., 2020, Grimm et al., 2020].

### 10.6.2. Unsupervised RL

Research in unsupervised RL spans many fields, from computational accounts of useful intrinsic motivations [Barto et al., 2004] to empirical evidence for certain intrinsic costs in humans [Kool et al., 2013]. Such intrinsic behavior learning could aid an RL agent to adapt across tasks posed by the environment in a sample-efficient manner. Although it is unclear as to the exact nature and origin of good intrinsic reward functions, there have been extensive studies recently that streamlined the way one can conveniently categorize such approaches. Oudeyer and Kaplan [2008] classified intrinsic motivation algorithms into three different kinds - knowledge-based, competence-based and data-based models. Knowledge-based models rely on an prediction-error signal to build pseudo-rewards [Pathak et al., 2017b, 2019b, Burda et al., 2019a, Mazzaglia et al., 2021, Burda et al., 2019b, Rajeswar et al., 2021]. Competence models aim at learning a set of diverse and repeatable policies through information-theoretic objectives [Mohamed and Rezende, 2015]. This is achieved by maximizing the mutual information between the trajectory or states and latent skill variables [Eysenbach et al., 2019, Gregor et al., 2016, Liu and Abbeel, 2021a, Frank et al., 2014]. Data-based methods try to increase the diversity of the dataset, often times through explicit maximum entropy objectives or count-based objectives. Bellemare et al. [2016], Ostrovski et al. [2017b], Liu and Abbeel [2021b], Yarats et al. [2019]. An unsupervised RL approach that is closely related to our method is Plan2Explore Sekar et al. [2020], which combines Disagreement [Pathak et al., 2019b], a knowledge-based exploration approach, with the first iteration of the Dreamer agent [Hafner et al., 2019a]. In their work, they also presented few-shot adaptation experiments. We improve upon their method in several ways: combining various unsupervised RL strategies with the DreamerV2 agent [Hafner et al., 2021], exploiting the pre-trained components of the agent for fast adaptation, and employing a hybrid planning strategy to improve data-efficiency.

### 10.6.3. Representation Learning

When inputs are high-dimensional images, it is beneficial to learn compact state representations of the inputs. Much progress in unsupervised representation learning for RL has been influenced by developments in vision-based unsupervised learning [Chen et al., 2020b, Kingma and Welling, 2013]. More recently, a number of works have investigated representation learning for RL [Srinivas et al., 2020, Laskin et al., Yarats et al., 2021]. In our work, we focus on representation learning with autoencoders [Hafner et al., 2019b, Yarats et al., 2019]. Specifically, we leverage DreamerV2 [Hafner et al., 2021] for efficient exploration and fast adaptation. Predicting ahead in learned latent space facilitates both long-term predictions and allows to efficiently predict thousands of compact state sequences in parallel.

## 10.7. Conclusion

In order to accelerate development and deployment of learning agents for real-world robotics tasks, it is crucial that the employed algorithms can adapt in a data-efficient way for multiple tasks. Our unsupervised model-based RL approach represents a step forward for training agents that can adapt faster, as shown by the near-optimal performance obtained in URLB and by the robustness showed in the more realistic tasks from the RWRL benchmark. We also highlighted several limitations of the method, mostly concerned about the quality of the learned pre-trained model, which we aim to address in the future, as we move towards applying our method in a real robotics setup.

Given that our model-based framework relies on the predictions of a learned reward function and, in turn, from the value predictions of the critic, we aim to find faster ways to adapt those functions, given the pre-trained experience. One idea would be to leverage the flexibility of successor representations and features [Barreto, 2018], which allow learning a task-agnostic estimate of the features expected under the actions of a certain actor. These predictions could then be exploited for transfer and adaptation. If we could learn useful successor features during pre-training, we might use them to solve downstream tasks faster during fine-tuning.

## 10.8. Limitations

Leveraging unsupervised interaction with the environment to pre-train the agent components has the potential to speed-up adaptation. However, the unsupervised behavior of a robot could cause harm to the environment and itself. Safety constraints or safety-aware algorithms should be introduced in our system in order to let self-supervised agents learn autonomously in real world.

We aimed at improving the action selection process during fine-tuning, by employing a hybrid planner that adopts both a classical MPC-based planner and an actor-critic architecture. However, our strategy overlooks the uncertainty of the model. In order to account for this, Bayes-adaptive RL strategies could be attempted [Mehta et al., 2022].

# Chapter 11

# Conclusion

## 11.1. Summary of Contributions

Perceptual structure can arise as a natural consequence of statistical regularities in the world. The motivating theme for this dissertation was to highlight the key elements missing in the current machine learning models to achieve the ability of natural agents in understanding the underlying structure of the world. We identified supervision as the critical bottleneck in developing ML systems that can draw inferences and make decisions in high dimensions. And we showed that, by using self-supervision, the performance of an ML model can be increased in several tasks, such as vision, object modeling, and robotics control. To offset the rich dense supervision requirements of current learning systems, we discussed the use of Probabilistic generative models and Iterated Learning for problems in the computer vision domain and Intrinsic control for Reinforcement Learning problems.

We now summarize the contributions within each of the four articles, while also examining their current limitations.

- **Unsupervised 3D Aware Representations from a Single Image.** Comprehending the image rendering pipeline as a differentiable function allowed us to learn generative models of 3D structures and recover these structures from vastly available 2D images via probabilistic inference. This is a valuable research direction as obtaining ground-truth 3D data supervision is expensive and often times impossible. Pix2Shape blends decades of advances in graphics rendering and the capabilities of deep generative models. One existing limitation with the approach is in scaling to the real-world images and scenes. This is due to an additional emphasis on the geometric consistencies imposed through physically based rendering. We believe that such stress could cripple the potential of underlying structured generative modeling. In the Future work section, we elaborate on scaling such models to arbitrary scenes in the wild.
- **Compositional Reasoning in a Weakly Supervised Setting.** Natural images possessing a single object is not a realistic assumption even in the case of curated

datasets like ImageNet. MILe attempts to tackle the problem by proposing a mechanism that gives us a multi-label descriptions of images. The approach presented two ways of aggregating compositions from single labeled data. Firstly, it uses a sigmoid to classify the images. Next, it leverages iterated learning framework to provide multiple labels as targerts to the sigmoid objective. A possible limitation, which is inherent to iterated learning is choosing an appropriate length of the teacher and imitator iterations, however, our proposed procedure is robust for a wide range of iterations.

- **Intrinsic Exploration to Overcome the need for Dense Supervision.** Sparse reward supervision complicates the temporal credit assignment problem significantly and negatively impacts the overall reinforcement learning process. Learning useful skills without supervision may help address challenges in exploration in such environments. The proposed approach leverages a multi-modal knowledge prediction pretext task to help attain meaningful behaviors. Such reward-free exploratory skills can aid the agent to adapt to various downstream tasks in a sample efficient manner. However, it is currently limited to a domain of tasks and not extended for other variety of tasks, which could be explored in the future work. Another limitation lies in assuming the availability of touch sensors or related sensory information in the agents.
- **Data-Efficient Unsupervised Reinforcement Learning.** Structured exploration has been promising at training generalist fast adapting agents. However, the existing approaches have only yielded sub-optimal results on a standardized setup, especially on high-dimensional pixel inputs. The proposed approach uses model-based exploration, during pre-training, to learn a knowledgeable world model, and adopts a hybrid planner to fine-tune efficiently, achieving comparable results to task-specific baselines, while using 20x less data. In addition, the model analysis sheds light on different aspects of the learned model that were key to improving the performance. An existing limitation is that if the environment introduces too intense perturbations, our model-based agent struggles, to the extent that using a planner even decreases performance. Developing more resilient models that can be trained in an unsupervised fashion and used for data-efficient planning will be the focus of future studies.

## 11.2. Discussion and Future Research

At the outset, we highlighted the learning abilities of natural agents and questioned key elements missing in current artificial agents. Although the thesis focuses on learning structure and structured behaviour from supervision constrained scenarios, human ability to reason, explore and adapt is far beyond any of these. We have the capability to perceive hierarchies, continua, and causal relations, among others. We believe that the other potential missing elements could include self-supervision from a temporal signal, causal structure discovery, and new inductive biases to help bridge the learning gap. We highlight some of these potential ideas in this section. I conclude by laying out some relevant problems to further enable scientific discovery through advancements in supervision-constrained machine learning.

### 11.2.1. Unsupervised Reinforcement Learning

Existing Reinforcement learning (RL) algorithms are competent but are mostly specialized (*e.g.* DQN, PPO, MuZero, etc). For any conceivable task, we have individual agents that are specifically trained for a particular task and each such agent typically requires millions of reward interactions to train. However, what is desired is to move towards generalist or adaptive agents, that when pretrained in an unsupervised manner can adapt to different but related downstream tasks. Natural agents, such as humans, often develop intrinsic desires and learn dynamic models of the world by simply interacting with the environment. Such unsupervised task-agnostic learning can scale down the reliance on unreasonable supervision [Jaderberg et al., 2017, Laskin et al., 2021]. We discussed such intrinsic control based approaches in Chapter 8 and Chapter 10. Currently, there is much needed progress to be made on fast adaptation for a downstream task, after a phase of unsupervised training and interaction with the environment. A careful analysis, as in Chapter 8 and Chapter 10 along with a theoretical investigation on the synergies of unsupervised RL approaches on high-dimensional inputs could be attempted in the future.

Different approaches to learning could also be studied for this particular training setting, where the agent experiences training through two different stages: pre-training and fine-tuning. Meta-learning, or learning to learn strategies for RL could help design a new approach to tackle the problem end-to-end [Finn et al., 2017, Gupta et al., 2018]. This approach can be further extended to address another challenge of existing unsupervised RL algorithms, which is building useful representations for learning incremental skills for a changing world. Current objectives for unsupervised RL make a major assumption about the environment stationarity. We aim to study this by expanding on the recently introduced URL benchmark [Laskin et al., 2021] with challenging exploration domains and realistic variations [Hansen et al., 2021].

### 11.2.2. Unsupervised 3D Aware Representations for Perception

Perceptual models that can understand and model geometrical or structural aspects associated with the 3D world are central to building and effectively training autonomous AI agents. Probabilistic generative models offer a promising approach to automate the reasoning and understanding of underlying data distribution. One active area of research we discussed in Chapter 4 leverages such generative models for structure learning tasks *e.g.* 3D from 2D. There has been growing interest in integrating neural rendering approaches to learning unsupervised 3D aware representations for novel view synthesis. However, these methods either lack 3D consistency [Ramirez et al., 2021] or are restricted to a single category of dataset [Cai et al., 2022]. We believe that tighter integration of structured generative scheme from Chapter 4 with recent volume rendering techniques (*e.g.* NeRF [Mildenhall et al., 2020]) could help effectively reason about any object category without being constrained to synthetic benchmarks or well-crafted and annotation-intensive datasets.

### 11.2.3. Iterated Learning and Multi-modal Transfer

Self-training has been a successful tool in obtaining improved generalization performance across a wide range of tasks [Araslanov and Roth, 2021, Xie et al., 2020]. Although predominantly useful in learning efficient compact models, the framework has enormous potential for extracting useful knowledge. Self-training with bottleneck learning (such as the procedure described in Chapter 6) can be generally applicable where there exists hidden information beyond available ground truth labels, and can significantly improve the generalization ability of the model (*e.g.* in a weakly supervised setting). I end by attempting to highlight one such avenue. Recently, large-scale multi-modal pre-training (VLP) models have shown promising results on various downstream multimodal alignment tasks, e.g., image-text retrieval and image classification [Radford et al., 2021, Jia et al., 2021]. Despite their remarkable abilities, the embedded knowledge has been restrictive and has not been exploited in generative multi-modal tasks (such as image captioning or VQA) or other semantically meaningful tasks such as object detection and segmentation. A desirable endeavor could be to distill useful knowledge while leveraging compositional inductive biases to be applicable to generative modeling schemes and spatial reasoning tasks.

## 11.3. Broader Impact

Artificial agents that can autonomously develop cognition and intentions are at the center of this thesis. We expect this research topic to be widely investigated in the future, as we aim to build agents that require little or no human intervention. Applications of this research could impact numerous fields, such as robotics, where robots could learn to sense and act by themselves, and AI-driven service systems, where the system could learn to better interact with its users. However, as we start employing unsupervised learning mechanisms several questions arise about their safety and ethical issues.

One major limitation of unsupervised learning strategies is that we have little knowledge and control of the model that is learned. This limits the explainability of the agent and of its intentions, making these models difficult to trust in real-life contexts. Unsupervised interaction with the environment could also represent a threat. For instance, the actions that a robot could try in its environment, in order to enrich its experience could cause damage to things and people around it. For these reasons, we believe future applications of unsupervised learning and reinforcement learning should be supported by means to make such applications safer for their users, in order to make the impact of this research even broader.

# References

Chapter 5 - intrinsic motivation and positive development. In R. M. Lerner, J. V. Lerner, and J. B. Benson, editors, *Positive Youth Development*, Advances in Child Development and Behavior. JAI, 2011. 103

M. Abdelsalam, M. Faramarzi, S. Sodhani, and S. Chandar. Iirc: Incremental implicitly-refined classification. *CVPR*, 2021. 89, 181, 182

A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. A. Riedmiller. Maximum a posteriori policy optimisation. *CoRR*, abs/1806.06920, 2018. 118

M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, and M. Yan. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022. 128

Z. Allen-Zhu and Y. Li. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*, 2020. 100, 101

A. Almahairi, S. Rajeswar, and A. Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. *International Conference on Machine Learning (ICML)*, 2018. 38

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba. Hindsight experience replay. In *NeurIPS*, 2017. 107

R. M. Angulo-kinzler. Exploration and selection of intralimb coordination patterns in 3-month-old infants. *Journal of Motor Behavior*, 33(4):363–376, 2001. 34

N. Araslanov and S. Roth. Self-supervised augmentation consistency for adapting semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 146

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. *International Conference on Machine Learning (ICML)*, 2017. 73

M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 19, 184, 185

D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR. org, 2017. 96

L. J. Ba and R. Caruana. Do deep nets really need to be deep? *arXiv preprint arXiv:1312.6184*, 2013. 100

A. Barbu, D. Mayo, J. Alverio, W. Luo, C. Wang, D. Gutfreund, J. B. Tenenbaum, and B. Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *NeurIPS*, 2019. 88, 99

H. Barlow. Unsupervised Learning. *Neural Computation*, 1(3):295–311, 1989. 33

A. Barreto. Transfer in reinforcement learning with successor features and generalised policy improvement. In *ICML*, 2018. 142

A. G. Barto, S. Singh, and N. Chentanez. Intrinsically motivated learning of hierarchical collections of skills. In *International Conference on Development and Learning(ICDL)*, 2004. 58, 108, 140

S. Beery, G. Van Horn, and P. Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473, 2018. 184

E. Bekele and W. Lawson. The deeper, the better: Analysis of person attributes recognition. In *International Conference on Automatic Face Gesture Recognition*, 2019. 186

M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29, 2016. 59, 108, 126, 141

R. Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 38(8):716–719, 1952. 55, 57

Y. Bengio, A. Courville, and P. Vincent. Representation learning: a review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8), August 2013. doi: 10.1109/tpami.2013.50. 51

L. Beyer, O. J. Hénaff, A. Kolesnikov, X. Zhai, and A. v. d. Oord. Are we done with imagenet? *arXiv preprint arXiv:2006.07159*, 2020. 35, 88, 89, 93, 94, 95, 96, 99, 100, 184

D. Billman and J. Knutson. Unsupervised concept learning and value systematicitiy: A complex whole aids learning the parts. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 2(22):458–475, 1996. 33

R. Blake, K. V. Sobel, and T. W. James. Neural synergy between kinetic vision and touch. *Psychological Science*, 15:397–402, 2004. 108

C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD*, pages 535–541, 2006. 100

Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros. Largescale study of curiosity-driven learning. *ICLR*, 2019a. 106, 107, 108, 111, 118, 126, 141

Y. Burda, H. Edwards, A. J. Storkey, and O. Klimov. Exploration by random network distillation. *ICLR*, 2019b. 107, 113, 131, 141, 199

H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019. 66

S. Cai, A. Obukhov, D. Dai, and L. V. Gool. Pix2nerf: Unsupervised conditional $\pi$-gan for single image to neural radiance fields translation. In *arXiv*, 2022. 146

R. Calandra, A. Owens, D. Jayaraman, J. Lin, W. Yuan, J. Malik, E. H. Adelson, and S. Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 2018. 109

M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 96

J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *CVPR*, 2017. 87

E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. Pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5799–5809, June 2021. 64

A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *arXiv*, 2015. 66, 69, 75

S. Chaudhuri, E. Kalogerakis, L. Guibas, and V. Koltun. Probabilistic reasoning for assembly-based 3d modeling. In *ACM SIGGRAPH*, 2011. 69

Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. Julian, C. Finn, et al. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021. 126

A. S. Chen, H. Nam, S. Nair, and C. Finn. Batch exploration with examples for scalable robotic reinforcement learning. *arXiv*, 2020a. 107

K. Chen, Y. Lee, and H. Soh. Multi-modal mutual information (mummi) training for robust self-supervised deep reinforcement learning. *ICRA*, 2021. 109

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020b. 18, 96, 126, 141

T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint:2006.10029*, 2020c. 96, 97, 182

W. Chen, J. Gao, H. Ling, E. J. Smith, J. Lehtinen, A. Jacobson, and S. Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. *CoRR*, abs/1908.01210, 2019. 66, 69, 70

X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020d. 96

C. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. *ArXiv*, 2016. 66, 69

K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, 2018. 131

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Workshop on Deep Learning, 2014*, 2014. 130

M. Cogswell, J. Lu, S. Lee, D. Parikh, and D. Batra. Emergence of compositional language with deep generational transmission. *arXiv preprint arXiv:1904.09067*, 2019. 101

M. R. Connolly and L. Harris. Effects of stimulus incongruity on children's curiosity as measured by looking time and expression change. *Psychonomic Science*, 1971. 33, 106, 110

R. Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006. 57, 140

E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning, 2016–2019. 113, 114, 189, 190

A. Courville, G. Desjardins, J. Bergstra, and Y. Bengio. The spike-and-slab rbm and extensions to discrete and sparse data distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(9), 2014. 49

G. Dagan, D. Hupkes, and E. Bruni. Co-evolution of language and agents in referential games. *arXiv preprint arXiv:2001.03361*, 2020. 101

P. de Haan, D. Jayaraman, and S. Levine. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

V. R. de Sa. Learning classification with unlabeled data. In *NeurIPS*, 1993. 108

V. Dean, S. Tulsiani, and A. Gupta. See, hear, explore: Curiosity via audio-visual association. In *NeurIPS*, 2020. 109

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 35, 36, 87

S. I. Di Domenico and R. M. Ryan. The emerging neuroscience of intrinsic motivation: A new frontier in self-determination research. *Frontiers in Human Neuroscience*, 2017. 108

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31st International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 647–655, 2014. 33, 48

J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016. 53, 72

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. 34

R. Dubey and T. L. Griffiths. Understanding exploration in humans and machines by formalizing the function of curiosity. *Current Opinion in Behavioral Sciences*, 2020. 108

G. Dulac-Arnold, D. Mankowitz, and T. Hester. Challenges of real-world reinforcement learning, 2019. 34, 127

G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. An empirical investigation of the challenges of real-world reinforcement learning, 2020a. 127, 134, 203

G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. An empirical investigation of the challenges of real-world reinforcement learning, 2020b. 34

V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016. 53, 67, 70, 72, 73

A. E and R. CH. The importance of touch in development. *Paediatr Child Health*, 2010;15(3). 106

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019. 131, 141, 200

C. A. Fields. The principle of persistence, leibniz's law, and the computational task of object re-identification. *Human Development*, 56, 2013. 34

C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. 145

S. Forestier, Y. Mollard, and P. Oudeyer. Intrinsically motivated goal exploration processes with automatic curriculum learning. *CoRR*, 2017. 105, 107

M. Frank, J. Leitner, M. Stollenga, A. Förster, and J. Schmidhuber. Curiosity driven reinforcement learning for motion planning on humanoids. *Frontiers in Neurorobotics*, 7, 2014. 59, 141

K. P. F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. 48

T. Furlanello, Z. Lipton, M. Tschannen, L. Itti, and A. Anandkumar. Born again neural networks. In *ICML*, 2018. 100

M. Gadelha, S. Maji, and R. Wang. 3d shape induction from 2d views of multiple objects. *CoRR*, abs/1612.05872, 2016. 69

Z. Gan, Y.-C. Chen, L. Li, C. Zhu, Y. Cheng, and J. J. Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020. 109

R. Gao, R. Feris, and K. Grauman. Learning to separate object sounds by watching unlabeled video. *ECCV*, 2018. 108

Y. Gao, L. A. Hendricks, K. J. Kuchenbecker, and T. Darrell. Deep learning for tactile understanding from visual and haptic data. In *ICRA*, 2016. 109

M. Gasse, D. GRASSET, G. Gaudron, and P.-Y. Oudeyer. Causal reinforcement learning using observational and interventional data, 2022.

R. Geirhos, C. R. M. Temme, J. Rauber, H. H. Schütt, M. Bethge, and F. A. Wichmann. Generalisation in humans and deep neural networks, 2018. 34

R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. *European Conference of Computer Vision (ECCV)*, 2016. 66, 69

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014a. 52

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems (NIPS)*, 2014b. 67, 72

P. Goyal, M. Caron, B. Lefaudeux, M. Xu, P. Wang, V. Pai, M. Singh, V. Liptchinsky, I. Misra, A. Joulin, et al. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*, 2021. 96

K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *CoRR*, 2016. 108, 141

J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 96

C. Grimm, A. Barreto, S. Singh, and D. Silver. The value equivalence principle for model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 5541–5552, 2020. 140

I. Gulrajani and D. Lopez-Paz. In search of lost domain generalization. *NeurIPS*, 2020. 185

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *Advances in Neural Information Processing Systems (NIPS)*, 2017. 73

S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. Scott, and D. Huang. Curriculumnet: Weakly supervised learning from large-scale web images. In *ECCV*, 2018. 95

S. Guo, Y. Ren, S. Havrylov, S. Frank, I. Titov, and K. Smith. The emergence of compositional languages for numeric concepts through iterated learning in neural agents, 2019. 101

A. Gupta, B. Eysenbach, C. Finn, and S. Levine. Unsupervised meta-learning for reinforcement learning. *CoRR*, 2018. 145

D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 129

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018. 34, 57, 110, 113, 128, 131, 191

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *null*, 2006. 173

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019a. 57, 130, 140, 141

D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565, 2019b. 129, 130, 141

D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. 33, 129, 130, 141, 202

N. Hansen, H. Su, and X. Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. In *Conference on Neural Information Processing Systems*, 2021. 145

N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control. 2022. 133, 198

F. Hausdorff. *Grundzüge der Mengenlehre*. Chelsea Pub. Co, New York, 1949. ISBN 9780828400619. 76

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 81, 87, 93

K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask r-cnn. *ICCV*, 2017. 87

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019a. 96

T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, pages 558–567, 2019b. 94

P. Henderson and V. Ferrari. Learning to generate and reconstruct 3d meshes with only 2d supervision. *CoRR*, abs/1807.09259, 2018. 66, 69, 70

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017. 53

T. Hey, S. Tansley, and K. Tolle. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009. 33

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597. 48

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 100

G. E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*. 2012. 49

G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. *Distributed Representations*, page 77–109. 1986. 43

R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. D. Turck, and P. Abbeel. Curiosity-driven exploration in deep reinforcement learning via bayesian neural networks. *CoRR*, 2016. 108

J. Huang, Y. Zhou, T. A. Funkhouser, and L. J. Guibas. Framenet: Learning local canonical frames of 3d surfaces from a single RGB image. *CoRR*, abs/1903.12305, 2019a. 69

S. H. Huang, M. Zambelli, J. Kay, M. F. Martins, Y. Tassa, P. M. Pilarski, and R. Hadsell. Learning gentle object manipulation with curiosity-driven deep reinforcement learning. *CoRR*, 2019b. 107

D. Hubel and T. Wiesel. Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962. 45

E. Insafutdinov and A. Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. *CoRR*, abs/1810.09381, 2018. 70

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning (ICML)*, 2015. 73, 172

A. Iscen, J. Valmadre, A. Arnab, and C. Schmid. Learning with neighbor consistency for noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 86

M. Ishmael, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, and A. Courville. Mutual information neural estimation. *International Conference on Machine Learning (ICML)*, 2018. 38, 74, 173

M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *5th*

*International Conference on Learning Representations, ICLR*, 2017. 145

C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. 2021. 146

C. M. Jiang, D. Wang, J. Huang, P. Marcus, and M. Nießner. Convolutional neural networks on non-uniform geometrical signals using euclidean spectral transformation. *CoRR*, abs/1901.02070, 2019. 69

L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018. 95

Y. Jiang, D. Ji, Z. Han, and M. Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 62

J. T. Kajiya. The rendering equation. In *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1986. 59, 171

M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal. Learning force control policies for compliant manipulation. In *IROS*, 2011. 108

D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *ArXiv*, abs/1806.10293, 2018. 125

E. Kalogerakis, S. Chaudhuri, D. Koller, and V. Koltun. A probabilistic model for component-based shape synthesis. *ACM Transactions in Graphics*, 31(4):55:1–55:11, 2012. 69

A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *European Conference on computer Vision (ECCV)*, 2018. 69

A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-specific object reconstruction from a single image. *CoRR*, abs/1411.6069, 2014. 66

T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018. 52

H. Kato and T. Harada. Learning view priors for single-view 3d reconstruction. *CoRR*, abs/1811.10719, 2018. 69

H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. *CoRR*, abs/1711.07566, 2017. 70

H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 61

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *ICLR*, 2015. 112

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 141

D. P. Kingma and M. Welling. Stochastic Gradient VB and the Variational Auto-Encoder. *2nd International Conference on Learning Representationsm (ICLR)*, pages 1–14, 2014. 51

S. Kirby. Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation, 5(2):102–110*, 2001. 36, 89, 91, 101

S. Kirby. Natural language from artificial life. *Artificial life, 8(2): 185–215*, 2002. 89, 101

S. Kirby, T. Griffiths, and K. Smith. Iterated learning and the evolution of language. *Current opinion in neurobiology, 28:108–114*, 2014. 89, 101

A. S. Klyubin, D. Polani, and D. L. Nehaniv. Empowerment: A universal agent-centric measure of control. *In 2005 IEEE Congress on Evolutionary Computation*, 2005. 108

L. Kobbelt and M. Botsch. A survey of point-based techniques in computer graphics. *Computers & Graphics*, 28(6):801–814, 2004. 70

M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath. Causalgan: Learning causal implicit generative models with adversarial training. *CoRR*, abs/1709.02023, 2017.

G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015. 81

W. Kool, J. McGuire, G. Wang, and M. Botvinick. Neural and behavioral evidence for an intrinsic cost of self-control. *PloS one*, 8, 2013. 58, 140

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 34, 87

D. Krueger, E. Caballero, J. Jacobsen, A. Zhang, J. Binas, R. L. Priol, and A. C. Courville. Out-of-distribution generalization via risk extrapolation (rex). *CoRR*, 2020. 184, 185

T. D. Kulkarni, W. Whitney, P. Kohli, and J. B. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 69

T. D. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih. Unsupervised learning of object keypoints for perception and control. *CoRR*, abs/1906.11883, 2019. 119

N. Lair, C. Colas, R. P. J.-M. Dussoux, P. F. Dominey, and P.-Y. Oudeyer. Language grounding through social interactions and curiosity-driven multi-goal learni. *arXiv*, 2019. 108

B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017. doi: 10.1017/S0140525X16001837. 59

M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas. Reinforcement learming with augmented data. arXiv:2004.14990. 141

M. Laskin, D. Yarats, H. Liu, K. Lee, A. Zhan, K. Lu, C. Cang, L. Pinto, and P. Abbeel. URLB: Unsupervised reinforcement learning benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

19, 27, 124, 126, 127, 128, 131, 132, 145, 197, 198

Y. LeCun. A path towards autonomous machine intelligence. In *OpenReview*, 2022. 124

Y. LeCun and C. Cortes. The mnist database of handwritten digits. 2005. 99

Y. LeCun, S. Chopra, R. Hadsell, F. J. Huang, and et al. A tutorial on energy-based learning. In *PREDICTING STRUCTURED DATA*. MIT Press, 2006. 49

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. 42

A. X. Lee, C. Devin, Y. Zhou, T. Lampe, K. Bousmalis, J. T. Springenberg, A. Byravan, A. Abdolmaleki, N. Gileadi, D. Khosid, C. Fantacci, J. E. Chen, A. S. Raju, R. Jeong, M. Neunert, A. Laurens, S. Saliceti, F. Casarini, M. A. Riedmiller, R. Hadsell, and F. Nori. Beyond pick-and-place: Tackling robotic stacking of diverse shapes. *ArXiv*, abs/2110.06192, 2021. 125

J.-T. Lee, D. Bollegala, and S. Luo. "touching to see" and "seeing to feel": Robotic cross-modal sensory data generation for visual-tactile perception. *ICRA*, 2019a. 109

K.-H. Lee, X. He, L. Zhang, and L. Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *CVPR*, 2018. 18, 95

M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. *ICRA*, 2019b. 109

L. Legault. *Intrinsic and Extrinsic Motivation*. Springer International Publishing, 2016. 103

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, 2016a. 34, 108

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 2016b. 125

C. Li, H. Liu, C. Chen, Y. Pu, L. Chen, R. Henao, and L. Carin. Alice: Towards understanding adversarial learning for joint distribution matching. In *Advances in Neural Information Processing Systems (NIPS)*, 2017a. 17, 74, 83

F. Li and M. Bowling. Ease-of-teaching and language structure from emergent communication. In *NeurIPS*, 2019. 101

J. Li, C. Xiong, and S. C. Hoi. Mopro: Webly supervised learning with momentum prototypes. *ICLR*, 2021. 95

T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6), 2018. 62

W. Li, L. Wang, W. Li, E. Agustsson, and L. Gool. Webvision database: Visual learning and understanding from web data. *ArXiv*, 2017b. 86, 89, 93, 100

X. Li, W. Wang, X. Hu, and J. Yang. Selective kernel networks. In *CVPR*, pages 510–519, 2019a. 97

Y. Li, J.-Y. Zhu, R. Tedrake, and A. Torralba. Connecting touch and vision via cross-modal prediction. In *CVPR*, 2019b. 109

Z. Li, T. Dekel, F. Cole, R. Tucker, N. Snavely, C. Liu, and W. T. Freeman. Learning the depths of moving people by watching frozen people. *CoRR*, abs/1904.11111, 2019c. 66

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR*, 2016. 57, 128

U. Lindenberger and M. Lövdén. Brain plasticity in human lifespan development: The exploration–selection–refinement model. *Annual Review of Developmental Psychology*, 1: 197–222, 2019. 34

P. Lippe, S. Magliacane, S. Löwe, Y. M. Asano, T. Cohen, and E. Gavves. CITRIS: Causal Identifiability from Temporal Intervened Sequences. In *Proceedings of the 39th International Conference on Machine Learning, ICML 2022*, 2022. URL https://arxiv.org/abs/2202.03169.

G.-H. Liu, A. Siravuru, S. Prabhakar, M. Veloso, and G. Kantor. Learning end-to-end multimodal sensor policies for autonomous navigation. In *CoRL*, 2017. 108

H. Liu and P. Abbeel. Aps: Active pretraining with successor features. In *Proceedings of the 38th International Conference on Machine Learning*, pages 6736–6747, 2021a. 141

H. Liu and P. Abbeel. Unsupervised active pre-training for reinforcement learning. *ICLR*, 2021b. 131, 141, 200

S. Liu, W. Chen, T. Li, and H. Li. Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction. *CoRR*, abs/1901.05567, 2019a. 70

S. Liu, T. Li, W. Chen, and H. Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, 2019b. 61

S. Liu, J. Niles-Weed, N. Razavian, and C. Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *NeurIPS*, 2020. 17, 91, 94, 96

W. Liu, Z. Liu, H. Wang, L. Paull, B. Schölkopf, and A. Weller. Iterative teaching by label synthesis, 2021. 91

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 93, 185, 186

S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4), July 2019. 61

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 88

M. M. Loper and M. J. Black. OpenDR: An approximate differentiable renderer. In *ECCV 2014*, volume 8695, pages 154–169. 61, 69, 70

Y. Lu, S. Singhal, F. Strub, A. Courville, and O. Pietquin. Countering language drift with seeded iterated learning. In *ICML*, 2020a. 91, 101, 102, 185

Y. Lu, S. Singhal, F. Strub, O. Pietquin, and A. Courville. Supervised seeded iterated learning for interactive language learning. In *EMNLP*, 2020b. 101

Y. Lu, K. Hausman, Y. Chebotar, M. Yan, E. Jang, A. Herzog, T. Xiao, A. Irpan, M. Khansari, D. Kalashnikov, and S. Levine. AW-opt: Learning robotic skills with imitation andreinforcement at scale. In *5th Annual Conference on Robot Learning (CoRL)*, 2021. 125

K. Marino, A. Gupta, R. Fergus, and A. Szlam. Hierarchical rl using an ensemble of proprioceptive periodic policies. *ICLR*, 2019. 108

M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020. 181

P. Mazzaglia, O. Çatal, T. Verbelen, and B. Dhoedt. Curiosity-driven exploration via latent bayesian surprise. *ArXiv*, abs/2104.07495, 2021. 131, 141, 200

W. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 5, 115–133*, 1943. 42

V. Mehta, B. Paria, J. Schneider, W. Neiswanger, and S. Ermon. An experimental design perspective on model-based reinforcement learning. In *International Conference on Learning Representations*, 2022. 142

T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernockỳ. Empirical evaluation and combination of advanced language modeling techniques. In *INTERSPEECH*, 2011. 172

B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *CoRR*, abs/2003.08934, 2020. 146

M. Mirza and S. Osindero. Conditional generative adversarial nets. *Arxiv*, 2014. 79

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, 2013. 105

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 56

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *ICML*, 2016. 191

H. Mobahi, M. Farajtabar, and P. L. Bartlett. Self-distillation amplifies regularization in hilbert space, 2020. 101

S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, page 2125–2133. MIT Press, 2015. 58, 141

A. Murali, Y. Li, D. Gandhi, and A. Gupta. Learning to grasp without seeing. *International Symposium on Experimental Robotics*, 2018. 109

A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 57

T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang. Hologan: Unsupervised learning of 3d representations from natural images. *Arxiv*, 2019. 64, 69

C. Niu, J. Li, and K. Xu. Im2struct: Recovering 3d shape structure from a single RGB image. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 176

D. Novotný, D. Larlus, and A. Vedaldi. Learning 3d object categories by looking around them. *CoRR*, abs/1705.03951, 2017. 69

D. Novotný, N. Ravi, B. Graham, N. Neverova, and A. Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. *ArXiv*, abs/1909.02533, 2019. 66

OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. A. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. M. Zhang. Solving rubik's cube with a robot hand. *ArXiv*, abs/1910.07113, 2019. 125

G. Ostrovski, M. G. Bellemare, A. van den Oord, and R. Munos. Count-based exploration with neural density models. *ICML*, 2017a. 108

G. Ostrovski, M. G. Bellemare, A. van den Oord, and R. Munos. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 2017b. 141

P.-Y. Oudeyer and F. Kaplan. How can we define intrinsic motivation ? In *the 8th International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*. Lund University Cognitive Studies, Lund:LUCS, Brighton, 2008. 58, 140

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 2009. 108

P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 2007. 108

A. Owens and A. A. Efros. Audio-visual scene analysis with self-supervised multisensory features. *ECCV*, 2018. 108

L. Pape, C. M. Oddo, M. Controzzi, C. Cipriani, A. Förster, M. C. Carrozza, and J. Schmidhuber. Learning tactile skills through curious exploration. *Frontiers in Neurorobotics*, 6:6, 2012. 109

S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta. The unsurprising effectiveness of pre-trained vision models for control, 2022. 128

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch, 2017. 114

D. Pathak, P. Agrawal, A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. *ICML*, 2017a. 106, 107, 108, 110, 111, 117

D. Pathak, P. Agrawal, A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. *ICML*, 2017b. 113, 126, 131, 141, 199

D. Pathak, D. Gandhi, and A. Gupta. Self-supervised exploration via disagreement. In *ICML*, 2019a. 113

D. Pathak, D. Gandhi, and A. Gupta. Self-supervised exploration via disagreement. In *ICML*, 2019b. 24, 126, 141, 198, 199

E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. *arXiv preprint arXiv:1709.07871*, 2017. 73

H. Pfister, M. Zwicker, J. van Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Annual Conference on Computer Graphics and Interactive Techniques*, 2000. 71

A. Piché, V. Thomas, C. Ibrahim, Y. Bengio, and C. Pal. Probabilistic planning with sequential monte carlo methods. In *International Conference on Learning Representations*, 2018. 57, 140

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations (ICLR)*, 2015. 172

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019. 126

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 89, 146

R. Raileanu and T. Rocktäschel. RIDE: Rewarding impact-driven exploration for procedurallygenerated environments. *ICLR*, 2020. 106, 108

S. Rajeswar, M. Ishmael Belghazi, O. Mastropietro, J. mitrovic, N. Rostamzadeh, and A. Courville. Hierarchical adversarially learned inference. *ICML workshop*, 2018. 38

S. Rajeswar, F. Mannan, F. Golemo, D. Vazquez, D. Nowrouzezahrai, and A. Courville. Pix2shape: Towards unsupervised learning of 3d scenes from images using a view-based representation. *International Journal of Computer Vision (IJCV)*, 7(5):889–904, 2020. 36

S. Rajeswar, C. Ibrahim, N. Surya, F. Golemo, D. Vazquez, A. Courville, and P. O. Pinheiro. Haptics-based curiosity for sparse-reward tasks. In *5th Annual Conference on Robot Learning*, 2021. 37, 141

S. Rajeswar, P. Mazzaglia, T. Verbelen, A. Piché, B. Dhoedt, A. Courville, and A. Lacoste. Unsupervised model-based pre-training for data-efficient reinforcement learning from pixels. In *Decision Awareness in Reinforcement Learning Workshop at ICML 2022*, 2022a. 37

S. Rajeswar, P. Rodriguez, S. Singhal, D. Vazquez, and A. Courville. Multi-label iterated learning for image classification with label ambiguity. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b. 36

P. Z. Ramirez, A. Tonioni, and F. Tombari. Unsupervised novel view synthesis from a single image. *arxiv*, 2021. 146

B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? *arXiv*, 2019a. 88

B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811*, 2019b. 99

D. A. Reed and J. Dongarra. Exascale computing and big data. *Commun. ACM*, 58(7):56–68, 2015. 33

Y. Ren, S. Guo, M. Labeau, S. B. Cohen, and S. Kirby. Compositional languages emerge in a neural iterated learning model. In *ICLR*, 2020. 101

D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, ICML'15, page 1530–1538. JMLR.org, 2015. 48

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14. JMLR.org, 2014. 51

D. J. Rezende, S. M. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 17, 26, 64, 67, 69, 70, 80, 81, 176, 177

A. G. Richards. *Robust constrained model predictive control.* PhD thesis, Massachusetts Institute of Technology, 2005. 131

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. 48

R. Y. Rubinstein and D. P. Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*, volume 133. Springer, 2004. 57, 132, 140

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988. 43

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, pages 211–252, 2015. 93, 94

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016. 53

T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: A pixelcnn implementation with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017. 50

A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, 3:210–229, jul 1959. 39

N. Savinov, A. Raichuk, R. Marinier, D. Vincent, M. Pollefeys, T. Lillicrap, and S. Gelly. Episodic curiosity through reachability. *aarXiv preprint arXiv:1810.02274*, 2018. 108

A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Transasctions on Pattern Anal. Mach. Intell. (PAMI)*, 31(5), 2009. 69

J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. *ICSAB: From animals to animats*, 1991. 108

J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, pages 230–247, 2010. 58, 131

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. doi: https://doi.org/10.1016/j.neunet.2014.09.003. 42

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. 140

F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. 2015. 186

J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016. 130

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017. 57, 128

R. Sekar, O. Rybkin, K. Daniilidis, P. Abbeel, D. Hafner, and D. Pathak. Planning to explore via self-supervised world models. In *ICML*, 2020. 108, 113, 131, 141, 199

Y. Seo, D. Hafner, H. Liu, F. Liu, S. James, K. Lee, and P. Abbeel. Masked world models for visual control, 2022. 124

M. Shah, K. Viswanathan, C.-T. Lu, A. Fuxman, Z. Li, A. Timofeev, C. Jia, and C. Sun. Inferring context from pixels for multimodal image classification. In *CIKM*. ACM, 2019. ISBN 9781450369763. 95

V. Shankar, R. Roelofs, H. Mania, A. Fang, B. Recht, and L. Schmidt. Evaluating machine accuracy on imagenet. In *ICML*, 2020. 88, 96, 99

A. Sharma, M. Ahn, S. Levine, V. Kumar, K. Hausman, and S. Gu. Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. *arXiv preprint arXiv:2004.12974*, 2020. 126

R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171 (3972):701–703, 1971. 177

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. V. D. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot. Mastering the game of go with deep neural networks and tree search. *nature*, 2016. 105

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017. 33

H. Singh, V. Hnizdo, A. Demchuk, and N. Misra. Nearest neighbor estimates of entropy. *American Journal of Mathematical and Management Sciences*, 23, 02 2003. doi: 10.1080/01966324.2003.10737616. 200

S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg. Intrinsically motivated reinforcement learning: An evolutionary perspective. *IEEE Transactions on Autonomous Mental Development*, 2 (2):70–82, 2010. 59

A. A. Soltani, H. Huang, J. Wu, T. D. Kulkarni, and J. B. Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. *Computer Vision and Pattern Recognition (CVPR)*, 2017. 69

E. S. Spelke and K. D. Kinzler. Core knowledge. *Developmental Science*, 10(1):89–96, 2007. 59

J. Speth and E. M. Hand. Automated label noise identification for facial attribute recognition. In *CVPR Workshops*, pages 25–28, 2019. 185

J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *ICLR*, 2015. 46

A. Srinivas, M. Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119*, 2020. arXiv:2004.04136. 141

N. Srivastava and R. R. Salakhutdinov. Multimodal learning with deep boltzmann machines. *NeurIPS*, 2012. 109

P. Stock and M. Cisse. Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases. In *ECCV*, pages 498–512, 2018. 88, 96, 99

A. L. Strehl and M. L. Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 2008. 108

P. F. Sturm. Pinhole camera model. In *Computer Vision, A Reference Guide*, 2014. 60

S. Subramanian, S. Rajeswar, A. Sordoni, A. Trischler, A. C. Courville, and C. Pal. Towards text generation with adversarially learned neural outlines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 38

C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017. 100

A. A. Taha and A. Hanbury. An efficient algorithm for calculating the exact hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015. 176

E. Talvitie. Learning the reward function for a misspecified model. In J. Dy and A. Krause, editors, *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 4838–4847. PMLR, 10–15 Jul 2018. 138

Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. P. Lillicrap, and M. A. Riedmiller. Deepmind control suite. *CoRR*, abs/1801.00690, 2018. 118, 126

K. A. Thoroughman and J. A. Taylor. Rapid reshaping of human motor generalization. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 25(39):8948–8953, Sep 2005. ISSN 1529-2401. doi: 10.1523/JNEUROSCI.1771-05.2005. 16192385[pmid]. 34

D. Tsipras, S. Santurkar, L. Engstrom, A. Ilyas, and A. Madry. From imagenet to image classification: Contextualizing progress on benchmarks. In *ICML*, 2020a. 88

D. Tsipras, S. Santurkar, L. Engstrom, A. Ilyas, and A. Madry. From imagenet to image classification: Contextualizing progress on benchmarks. In *ICML*, 2020b. 34, 88, 96, 99

Y. Tu, L. Niu, D. Cheng, and L. Zhang. Protonet: Learning from web data with memory. *CVPR*, 2020. 95

S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. *CoRR*, abs/1612.00404, 2016. 69

S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *CoRR*, abs/1704.06254, 2017. 66

A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *SSW*, 2016. 50

A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. ICML'16. JMLR.org, 2016. 50

H. van Hoof, N. Chen, M. Karl, P. van der Smagt, and J. Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *IROS*, 2016. 109

G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 99

A. Vani, M. Schwarzer, Y. Lu, E. Dhekane, and A. Courville. Iterated learning for emergent systematicity in vqa. In *ICLR*, 2021. 101

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 48

D. Vernon, C. Hofsten, and L. Fadiga. A roadmap for cognitive development in humanoid robots. *978-3-642-16903-8*, 11, 2011. 106

R. White. Motivation reconsidered: the concept of competence. *Psychological review*, 1959. 108

O. Wiles and A. Zisserman. Silnet : Single- and multi-view reconstruction by learning from silhouettes. *CoRR*, abs/1711.07888, 2017. 66

G. Williams, A. Aldrich, and E. Theodorou. Model Predictive Path Integral Control using Covariance Variable Importance Sampling. *arXiv e-prints*, art. arXiv:1509.01149, 2015. 131, 136

R. Woodcock, N. Mather, and K. McGrew. Woodcock johnson iii - tests of cognitive skills. *Riverside Pub*, 2001. 177

J. Wu, T. Xue, J. Lim, Y. Tian, J. Tenenbaum, A. Torralba, and W. Freeman. Single image 3d interpreter network. *ArXiv*, 2016a. 66, 69

J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *Advances in Neural Information Processing Systems (NIPS)*, 2016b. 66, 69

J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5d sketches. *CoRR*, abs/1711.03129, 2017. 66, 69

P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel. Daydreamer: World models for physical robot learning, 2022. 124

Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. *Computer Vision and Pattern Recognition (CVPR)*, 2015. 66, 69

Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le. Self-training with noisy student improves imagenet classification. In *CVPR*, pages 10687–10698, 2020. 99, 101, 146, 186

B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arxiv*, 2015. 112

J. Xu, S. Song, and M. Ciocarlie. Tandem: Learning joint exploration and decision making with tactile sensors, 2022. 104

X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 17, 66, 67, 69, 76, 79, 80, 81

J. Yang, L. Feng, W. Chen, X. Yan, H. Zheng, P. Luo, and W. Zhang. Webly supervised image classification with self-contained confidence. *ECCV*, 2020. 94, 95

D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving sample efficiency in model-free reinforcement learning from images. 2019. 141

D. Yarats, I. Kostrikov, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. 141

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022. 128

L. Yi, L. Shao, M. Savva, H. Huang, Y. Zhou, Q. Wang, B. Graham, M. Engelcke, R. Klokov, V. Lempitsky, Y. Gan, P. Wang, K. Liu, F. Yu, P. Shui, B. Hu, Y. Zhang, Y. Li, R. Bu, M. Sun, W. Wu, M. Jeong, J. Choi, C. Kim, A. Geetchandra, N. Murthy, B. Ramu, B. Manda, M. Ramanathan, G. Kumar, P. Preetham, S. Srivastava, S. Bhugra, B. Lall, C. Haene, S. Tulsiani, J. Malik, J. Lafer, R. Jones, S. Li, J. Lu, S. Jin, J. Yu, Q. Huang, E. Kalogerakis, S. Savarese, P. Hanrahan, T. Funkhouser, H. Su, and L. Guibas. Large-scale 3d shape reconstruction and segmentation from shapenet core55, 2017. 34

T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *CoRR*, 2019. 113, 189

S. Yun, S. J. Oh, B. Heo, D. Han, J. Choe, and S. Chun. Re-labeling imagenet: from single to multi-labels, from global to localized labels. In *CVPR*, 2021. 88, 89, 96, 99, 100

M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, 2014. 48

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. 96

L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *ICCV*, pages 3713–3722, 2019. 101

N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev. Panda: Pose aligned networks for deep attribute modeling. In *CVPR*, 2014. 186

X. Zhang, Z. Zhang, C. Zhang, J. Tenenbaum, B. Freeman, and J. Wu. Learning to reconstruct shapes from unseen classes. In *Advances in Neural Information Processing Systems (NIPS)*, 2018. 69

Z. Zhang, H. Zhang, S. O. Arik, H. Lee, and T. Pfister. Distilling effective supervision from severe label noise. In *CVPR*, 2020. 18, 95

H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 88

J.-Y. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. B. Tenenbaum, and W. T. Freeman. Visual object networks: Image generation with disentangled 3D representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018a. 69

Y. Zhu, Z. Wang, J. Merel, A. A. Rusu, T. Erez, S. Cabi, T. Saran, J. Kramár, R. Hadsell, N. de Freitas, and N. Heess. Reinforcement and imitation learning for diverse visuomotor skills. *CoRR*, 2018b. 105

# Appendix A

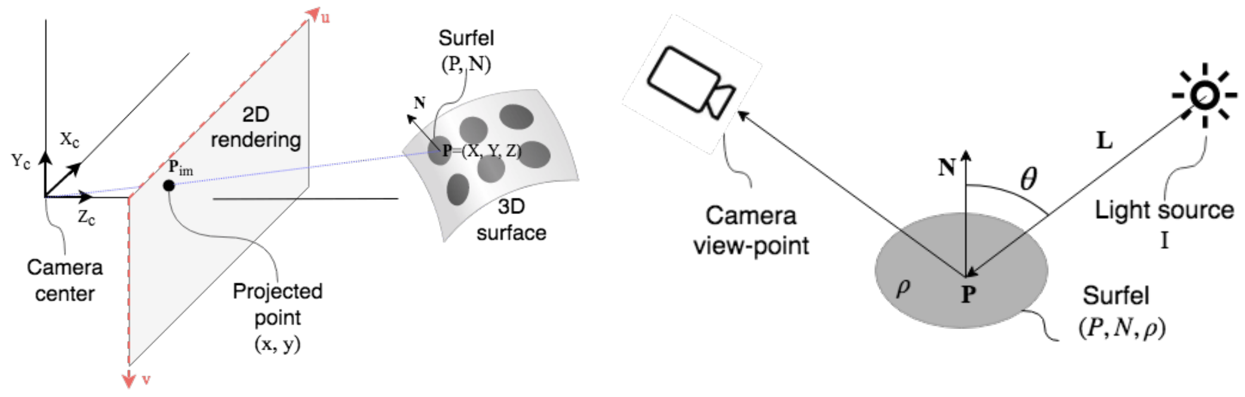
# Appendix for Chapter 4

## A.1. Rendering Details

The color of a surfel depends on the material reflectance, its position and orientation, as well as the ambient and point light source colors (See Figure A.1.1b). Given a surface point $P_i$, the color of its corresponding pixel $I_{rc}$ is given by the shading equation:

$$\begin{aligned} I_{rc} = \rho_i (L_a + \sum_j \frac{1}{k_l \|d_{ij}\| + k_q \|d_{ij}\|^2} L_j \\ \max\left(0, N_i^T d_{ij} / \|d_{ij}\|\right)), \end{aligned} \tag{A.1.1}$$

where $\rho_i$ is the surface reflectance, $L_a$ is the ambient light's color, $L_j$ is the $j^{\text{th}}$ positional light source's color, with $d_{ij} = L_j^{\text{pos}} - P_i$, or the direction vector from the scene point to the point light source, and $k_l$, $k_q$ being the linear and quadratic attenuation terms respectively. Equation A.1.1 is an approximation of rendering equation proposed in Kajiya [1986].



**(a)** Projection model **(b)** Shading model

**Fig. A.1.1. Differentiable 3D renderer.** (a) A surfel is defined by its position $P$, normal $N$, and reflectance $\rho$. Each surfel maps to an image pixel $P_{im}$. (b) The surfel's color depends on its reflectance $\rho$ and the angle $\theta$ between each light $I$ and the surfel's normal $N$.

## A.2. Architecture

Pix2Shape is composed of an encoder network (See Table A.2.1), a decoder network (See Table A.2.2), and a critic network (See Table A.2.3). Specifically, the decoder architecture is similar to the generator in DCGAN [Radford et al., 2015] but with LeakyReLU [Mikolov et al., 2011] as activation function and batch-normalization [Ioffe and Szegedy, 2015]. Also, we adjusted its depth and width to accommodate the high resolution images accordingly. In order to condition the camera position on the $z$ variable, we use conditional normalization in the alternate layers of the decoder. We train our model for 60K iterations with a batch size of 6 with images of resolution $128 \times 128 \times 3$.

| Layer | Output size | Kernel | Str. | BNorm | Activation |
|---|---|---|---|---|---|
| In $[x, c]$ | $128 \times 128 \times 3$ | | | | |
| Conv. | $64 \times 64 \times 85$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $32 \times 32 \times 170$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $16 \times 16 \times 340$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $8 \times 8 \times 680$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $4 \times 4 \times 1360$ | $4 \times 4$ | 2 | No | LReLU |
| Conv. | $1 \times 1 \times 1$ | $4 \times 4$ | 1 | No | |

**Table A.2.1. Pix2Shape encoder architecture**

| Layer | Output size | Kernel | Str. | BNorm | Activation |
|---|---|---|---|---|---|
| In $[x, c]$ | $131 \times 1$ | | | | |
| Conv. | $4 \times 4 \times 1344$ | $4 \times 4$ | 1 | Yes | LReLU |
| Conv. | $8 \times 8 \times 627$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $16 \times 16 \times 336$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $32 \times 32 \times 168$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $64 \times 64 \times 84$ | $4 \times 4$ | 2 | Yes | LReLU |
| Conv. | $128 \times 128 \times nCh$ | $4 \times 4$ | 2 | Yes | |

**Table A.2.2. Pix2Shape decoder architecture.**

| Layer | Output size | Kernel | Str. | BNorm | Activation |
|---|---|---|---|---|---|
| Input $[x, c]$ | $128 \times 128 \times 6$ | | | | |
| Conv. | $64 \times 64 \times 85$ | $4 \times 4$ | 2 | No | LReLU |
| Conv. | $32 \times 32 \times 170$ | $4 \times 4$ | 2 | No | LReLU |
| Conv. | $16 \times 16 \times 340$ | $4 \times 4$ | 2 | No | LReLU |
| Conv. | $8 \times 8 \times 680$ | $4 \times 4$ | 2 | No | LReLU |
| Conv. + [z] | $4 \times 4 \times 1360$ | $4 \times 4$ | 2 | No | LReLU |
| Convolution | $1 \times 1 \times 1$ | $4 \times 4$ | 1 | No | |

**Table A.2.3. Pix2Shape critic architecture.** Conditional version takes image, latent code $z$ and camera position $c$.

## A.3. Architecture for Semi-supervised experiments

Pixel2Surfels architecture remains similar to the previous one but with higher capacity on the decoder and critic. The most important difference is that for those experiments we do not condition the networks with the camera pose to be fair with the baselines. In addition to the three networks we have a statistics network (see Table A.3.4) that estimates and minimizes the mutual information between the two set of dimensions in the latent code using MINE [Ishmael et al., 2018]. Out of 128 dimensions for $z$ we use first 118 dimensions for represent scene-based information and rest to encode view based info.

| Layer | Output size | Kernel | Str. | BNorm | Act. |
|---|---|---|---|---|---|
| In $[z[:118], z[118:]]$ | $1 \times 1 \times 128$ | | | | |
| Conv. | $1 \times 1 \times 256$ | $1 \times 1$ | 1 | No | ELU |
| Conv. | $1 \times 1 \times 512$ | $1 \times 1$ | 1 | No | ELU |
| Conv. | $1 \times 1 \times 1$ | $1 \times 1$ | 2 | No | None |

**Table A.3.4. Pix2Shape statistics network architecture.**

The architecture of the baseline networks is shown in Figure A.3.2. During training we use the contrastive loss [Hadsell et al., 2006]:

$$\begin{aligned}\mathcal{L}_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2, y) &= (1-y)\frac{1}{2}(D_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2))^2 \\ &\quad + (y)\frac{1}{2}(max(0,m - D_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2)))^2 \\ D_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2) &= ||G_\theta(\boldsymbol{x}_1) - G_\theta(\boldsymbol{x}_2)||_2,\end{aligned} \tag{A.3.1}$$

where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are the input images, $y$ is either 0 (if the inputs are supposed to be the same) or 1 (if the images are supposed to be different), $G_\theta$ is each ResNet block, parameterized by $\theta$, and $m$ is the margin, which we set to 2.0. We apply the contrastive loss to the 2048 features that are generated by each ResNet block.

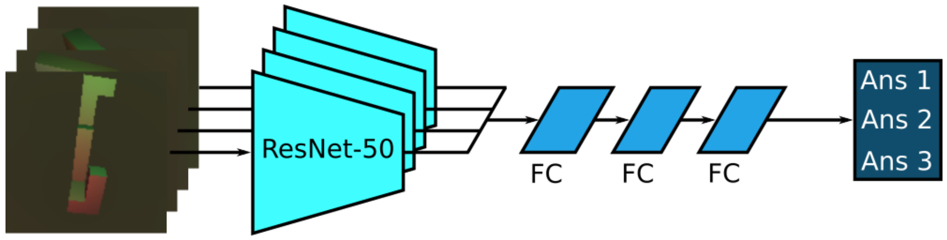
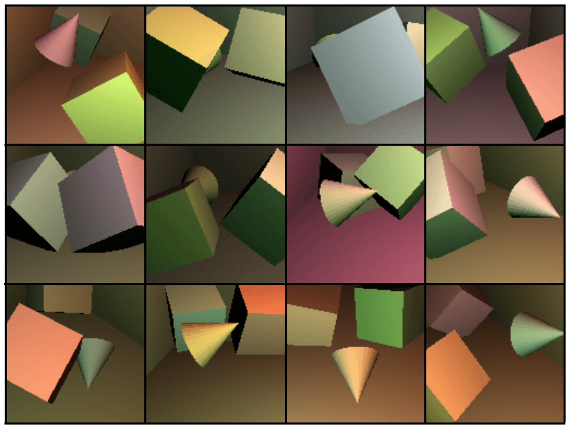
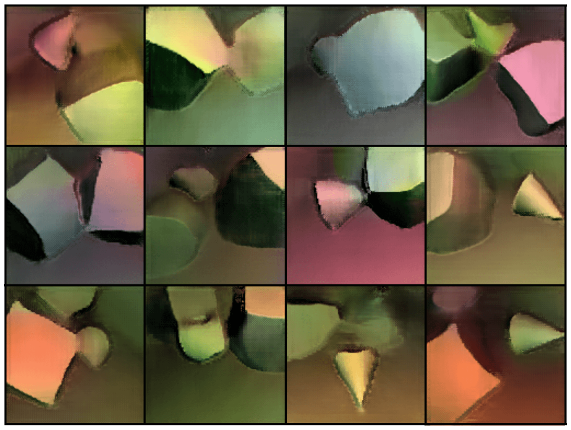


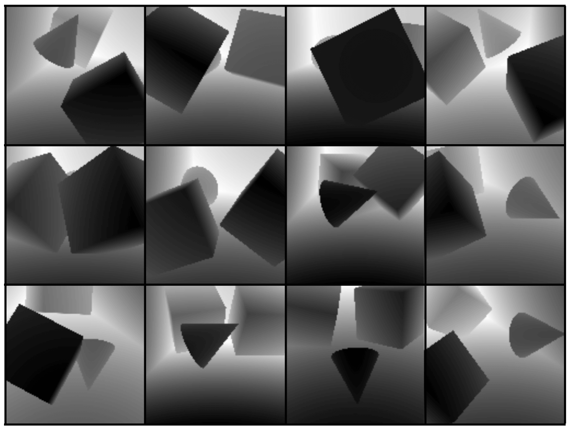
**Fig. A.3.2. 3D-IQTT baseline architecture.** The four ResNet-50 share the same weights and were slightly modified to support our image size. "FC" stands for fully-connected layer and the hidden node sizes are 2048, 512, and 256 respectively. The output of the network is encoded as one-hot vector.

## A.4. Material, Lights, and Camera Properties

Material. In our experiments, we use diffuse materials with uniform reflectance. The reflectance values are chosen arbitrarily and we use the same material properties for both the input and the generator side. Figure A.4.3 shows that it is possible to learn reflectance alongside learning the 3D structure of the scenes by requiring the model to predict the material coefficients along with the depth of the surfels. The color of the objects depend on both the lighting and the material properties. We do not delve into more details on this, as our objective is to model the structural/geometrical properties of the world with the model. This will be explored further in a later study.


**(a)** Color input images


**(b)** Reconstructed images


**(c)** Ground-truth depth

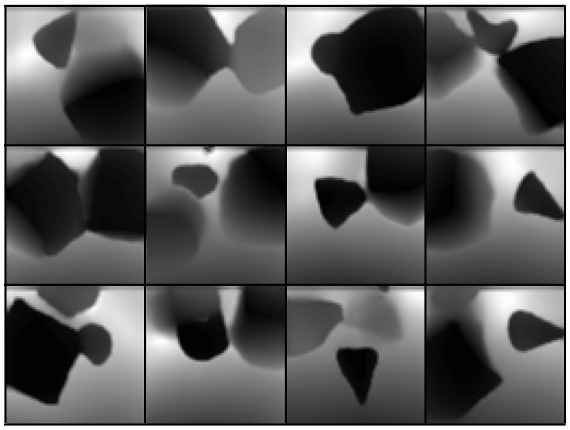

**(d)** Reconstructed depth

**Fig. A.4.3. Learning material along with structure.** The model learns the foreground and background colors separately.

Camera. The camera is specified by its position, viewing direction and vector indicating the orientation of the camera. The camera positions were uniform randomly sampled on a sphere for the 3D-IQTT task and on a spherical patch contained in the positive octant, for the rest of the experiments. The viewing direction was updated based on the camera position and

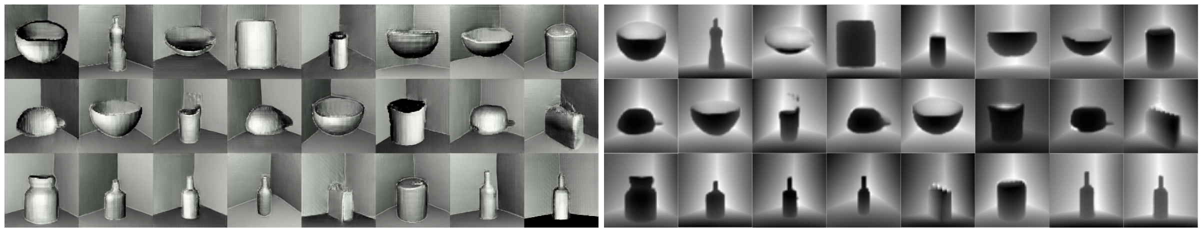
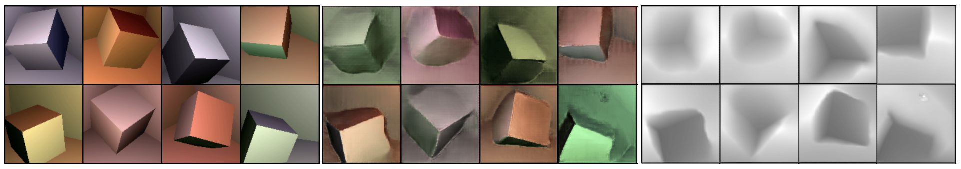


**Fig. A.4.4. Unconditional scene generation.** Generated samples from Pix2Shape model trained on ShapeNet scenes. **Left:** shaded images; **Right:** depth maps

the center of mass of the objects, so that the camera was always looking at a fixed point in the scene as its position changed. The focal length ranged between [18 mm and 25mm] in all the experiments and the field-of-view was fixed to 24mm. The camera properties were also shared between the input and the generator side. However, in the 3D-IQTT task we relaxed the assumption that we know the camera pose and instead estimate the view as a part of the learnt latent representation.
Lights. For the light sources, we experimented with single and multiple point-light sources, with the light colors chosen randomly. The light positions are uniformly sampled on a sphere for the 3D IQTT tasks, and uniformly on a spherical patch covering the positive octant for the other scenes. The same light colors and positions are used both for rendering the input and the generated images. The lights acted as a physical spot lights with the radiant energy attenuating quadratically with distance. As an ablation study we relaxed this assumption of having perfect knowledge of lights by using random position and random color lights. Those experiments show that the light information is not needed by our model to learn the 3D structure of the data. However, as we use random lights on the generator side, the shading of the reconstructions is in different color than in the input as shown in Figure A.4.5.

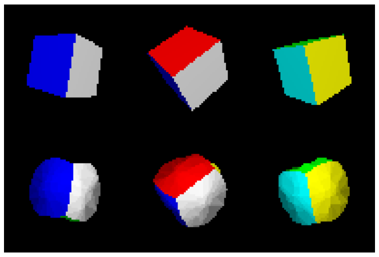
## A.5. Unconditional ShapeNet Generation

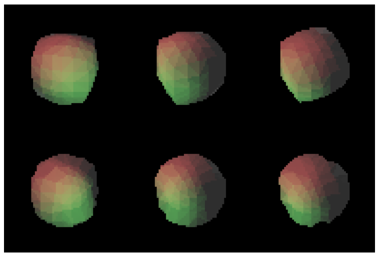
We trained Pix2Shape on scenes composed of ShapeNet objects from six categories (i.e., bowls, bottles, mugs, cans, caps and bags). Figure A.4.4 shows qualitative results on unconditional generation. Since no class information is provided to the model, the latent variable captures both the object category and its configuration.



**(a)** Input images **(b)** Reconstructions **(c)** Recovered depth

**Fig. A.4.5. Random lights configuration.**


(a) Reproduction of original results.


(b) Qualitative results on isolated and centered cube-like shape without background.

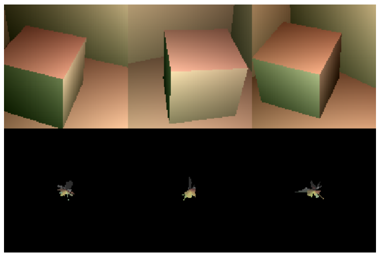

(c) Degenerative results on on full scene.

**Fig. A.5.6. Reproduction of Rezende et al. [2016] and qualitative results.** Top row: Samples of input images; bottom row: corresponding reconstructed images. We found that with a single centered object, the method was able to correctly reproduce the shape and orientation. However, when the object was not centered, too complex, or there was a background present, the method failed to estimate the shape.

## A.6. Evaluation of 3D Reconstructions

For evaluating 3D reconstructions, we use the Hausdorff distance [Taha and Hanbury, 2015] as a measure of similarity between two shapes as in Niu et al. [2018]. Given two point sets, $A$ and $B$, the Hausdorff distance is,

$$\max\left\{\max D_H^+(A,B), \max D_H^+(B,A)\right\},$$

where $D_H^+$ is an asymmetric Hausdorff distance between two point sets. E.g., $\max D_H^+(A,B) = \max D(a,B)$, for all $a \in A$, or the largest Euclidean distance $D(\cdot)$, from a set of points in $A$ to $B$, and a similar definition for the reverse case $\max D_H^+(B,A)$.

## A.7. Ablation study on depth supervision

As an ablation study, we repeated the experiment that demonstrates the view extrapolation capabilities of our model with depth superrvision. Table A.7.5 depicts the quantitative evaluations on reconstruction if the scenes from unobserved angles.

| | **Shape scenes** | | | | **Multiple-shape scenes** | | | |
|---|---|---|---|---|---|---|---|---|
| | 5° | 35° | 55° | 80° | 5° | 35° | 55° | 80° |
| Hausdorff-F | 0.093 | 0.088 | 0.085 | 0.096 | 0.173 | 0.218 | 0.194 | 0.201 |
| Hausdorff-R | 0.081 | 0.100 | 0.108 | 0.112 | 0.221 | 0.243 | 0.238 | 0.254 |
| MSE-depth | 0.004 | 0.004 | 0.005 | 0.007 | 0.009 | 0.008 | 0.008 | 0.009 |

**Table A.7.5. View point reconstruction.** Quantitative evaluation of implicit 3D reconstruction for unseen views by extrapolating the view angle from 0°(original) to 80° with depth supervision.

## A.8. 3D Intelligence Quotient Task.

In their landmark work, Shepard and Metzler [1971] introduced the mental rotation task into the toolkit of cognitive assessment. The authors presented human subjects with reference images and answer images. The subjects had to quickly decide if the answer was either a 3D-rotated version or a mirrored version of the reference. The speed and accuracy with which people can solve this mental rotation task has since become a staple of IQ tests like the Woodcock-Johnson tests [Woodcock et al., 2001]. We took this as inspiration to design a quantitative evaluation metric (number of questions answered correctly) as opposed to the default qualitative analyses of generative models. We use the same kind of 3D objects but instead of confronting our model with pairs of images and only two possible answers, we include several distractor answers and the subject (human or computer) has to to pick which one out of the three possible answers is the 3D-rotated version of the reference object (See Figure 4.2).

## A.9. Details on Human Evaluations for 3D IQTT

We posted the questionnaire to our lab-wide mailing list, where 41 participants followed the call. The questionnaire had one calibration question where, if answered incorrectly, we pointed out the correct answer. For all successive answers, we did not give any participant the correct answers and each participant had to answer all 20 questions to complete the quiz.

We also asked participants for their age range, gender, education, and for comments. While many commented that the questions were hard, nobody gave us a clear reason to discard their response. All participants were at least high school graduates currently pursuing a Bachelor's degree. The majority of submissions (78%) were male, whereas the others were female or unspecified. Most of our participants (73.2%) were between 18 and 29 years old, the others between 30 and 39. The resulting test scores are normally distributed according to the Shapiro-Wilk test ($p < 0.05$) and significantly different from random choice according to 1-sample Student's t test ($p < 0.01$).

## A.10. Implementation of Rezende et al.

With the publication of Rezende et al. [2016], the authors did not publicly release any code and upon request did not offer any either. When implementing our own version, we attempted to reproduce their results first, which is depicted in Figure A.5.6a. Further, we attempted to use the method for the same qualitative reconstruction of the primitive-in-box dataset as shown in Figure 4.4. We found that this worked only with one main object and when there was no background (see Figure A.5.6b). When including the background, applying the same method lead to degenerate solutions (see Figure A.5.6c).

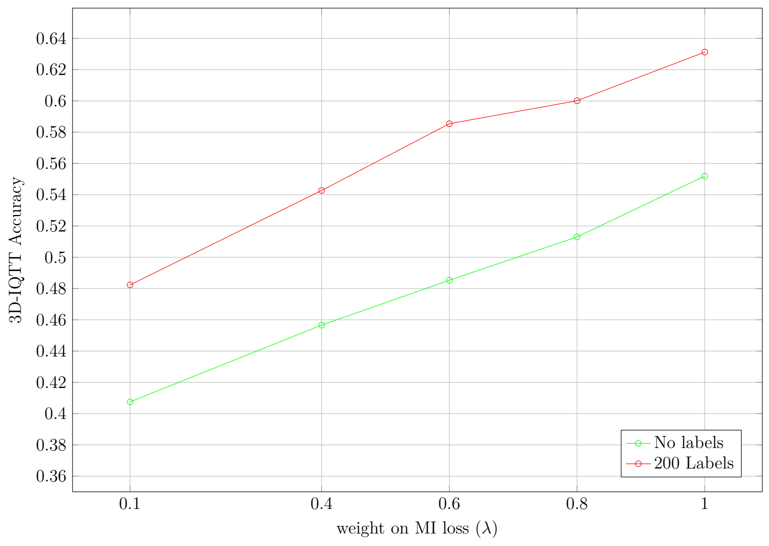


**Fig. A.11.7. Study of effect of mutual-information objective on 3D-IQTT performance.** Our model performance is correlated positively to the the weight on Mutual information term increases

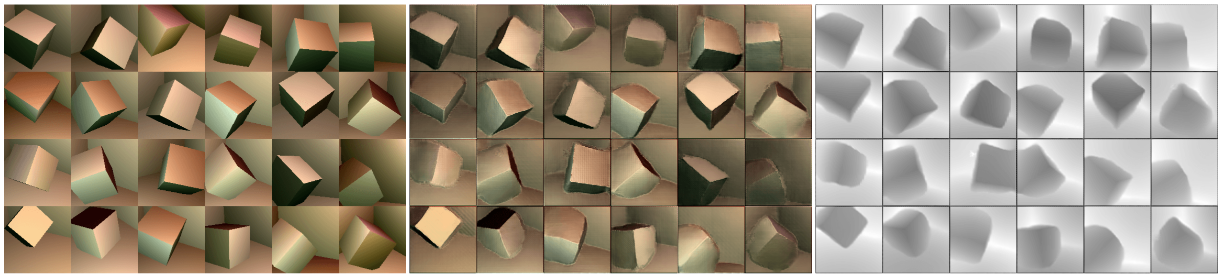
## A.11. Ablation study of the weighted Mutual-Info loss on 3D-IQTT

Consider the semi-supervised objective used in algorithm 1. In this section we do an ablation study on 3D-IQTT performance with the modified form of the equation where Mutual-information loss $I_\Theta(z_{scene}, z_{view})$ is weighted by a co-efficient $\lambda$. Plot in Figure A.11.7 indicates the importance of the MI term. Having a good estimate of the view point and disentangling the view information from geometry is indeed crucial to the performance of the IQ task.

$$L \leftarrow \mathcal{L}_{ALI} + \mathcal{L}_{recon} + I_\Theta(z_{scene}, z_{view})$$

## A.12. More Scene Reconstructions

Figure A.12.8 shows 3D reconstructions of scenes formed by boxes in a room. In Figure A.12.9 our model is asked to reconstruct the scenes of the first column and then render different views of the same scene. In this case we show the normal maps of those views. Figure A.12.10 shows the recovered shading, depth and normal images from reconstructions of complex scenes such as bedrooms and bunny.



(a) Input images (b) Reconstructed images (c) Reconstructed depth maps

**Fig. A.12.8. Scene reconstruction.** (a) Input images of rotated cubes into a room. (b) Pix2Shape reconstructions with its (c) associated depth maps.

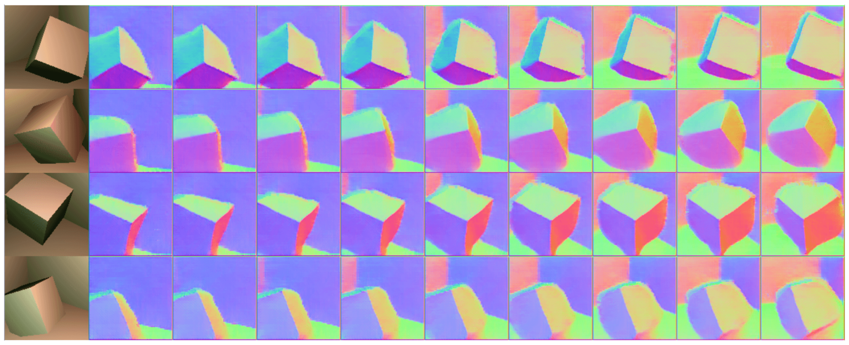
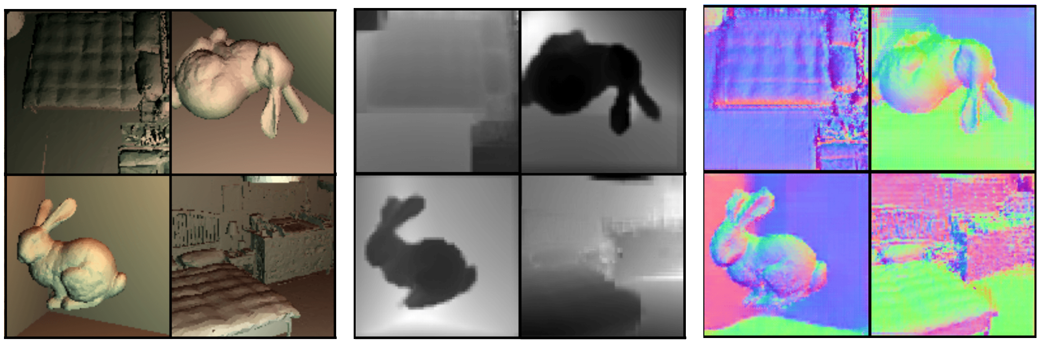


**Fig. A.12.9. Normal views reconstruction.** For each row, the first column is the input image and other columns are the extrapolated normal maps of that image from different views.



(a) Input images (b) Reconstructed depth (c) Reconstructed normal

**Fig. A.12.10. Reconstruction of complex scenes.** Reconstruction of bedroom scenes and bunny.

# Appendix B

# Appendix for Chapter 6

## B.1. Introduction

In Section B.2, we provide continual learning results on the IRCC benchmark [Abdelsalam et al., 2021]. In Section B.4 we investigate to which extent MILe is able to recover labels that were not present in the original dataset. In Section B.4 we provide additional details on the domain generalization experiment. In Section B.6, we provide additional results for multi-label classification on CelebA. In Section B.7, we test additional iterated learning schedules such as that of noisy student.

## B.2. IIRC benchmark

We explore whether MILe can incrementally learn an increasingly complex class hierarchy by teaching previously seen tasks to new generations. We experiment with Incremental Implicitly-Refined Classification (IIRC) [Abdelsalam et al., 2021], an extension to the class incremental learning setup [Masana et al., 2020] where the incoming batches of classes have two granularity levels, e.g. a coarse and a fine label. Labels are seen one at a time, and fine labels for a given coarse class are introduced after that coarser class is visited. The goal is to incorporate new finer-grained information into existing knowledge in a similar way as humans learn different breeds of dogs after learning the concept of dog.

### B.2.1. Metrics

As it can be seen in Fig. B.2.1, the two reported metrics are the precision-weighted Jaccard similarity and the mean precision-weighted Jaccard similarity.
Precision-weighted Jaccard Similarity. The Jaccard similarity (JS) refers to the intersection over union between model predictions $\hat{Y}_i$ and ground truth $Y_i$ for the $i$th sample:

$$JS = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap \hat{Y}_i|}{|Y_i \cup \hat{Y}_i|}, \tag{B.2.1}$$

The precision-weighted JS for task $k$ is the product between the JS and the precision for the samples belonging to that task:

$$R_{jk} = \frac{1}{n_k} \sum_{i=1}^{n_k} \frac{|Y_{ik} \cap \hat{Y}_{ik}|}{|Y_{ik} \cup \hat{Y}_{ik}|} \times \frac{|Y_{ik} \cap \hat{Y}_{ik}|}{\hat{Y}_{ik}}$$

where $(j \geq k), \hat{Y}_{ik}$ is the set of (model) predictions for the $i$th sample in the $k$th task, $Y_{ik}$ are the ground truth labels, and $n_k$ is number of samples in the task. $R_{jk}$ can be used as a proxy for the model's performance on the $k$th task as it trains on more tasks (i.e. as j increases).

Mean precision-weighted Jaccard similarity. We evaluate the overall performance of the model after training until the task $j$, as the average precision-weighted Jaccard similarity over all the classes that the model has encountered so far. Note that during this evaluation, the model has to predict all the correct labels for a given sample, even if the labels were seen across different tasks.

### B.2.2. Results.

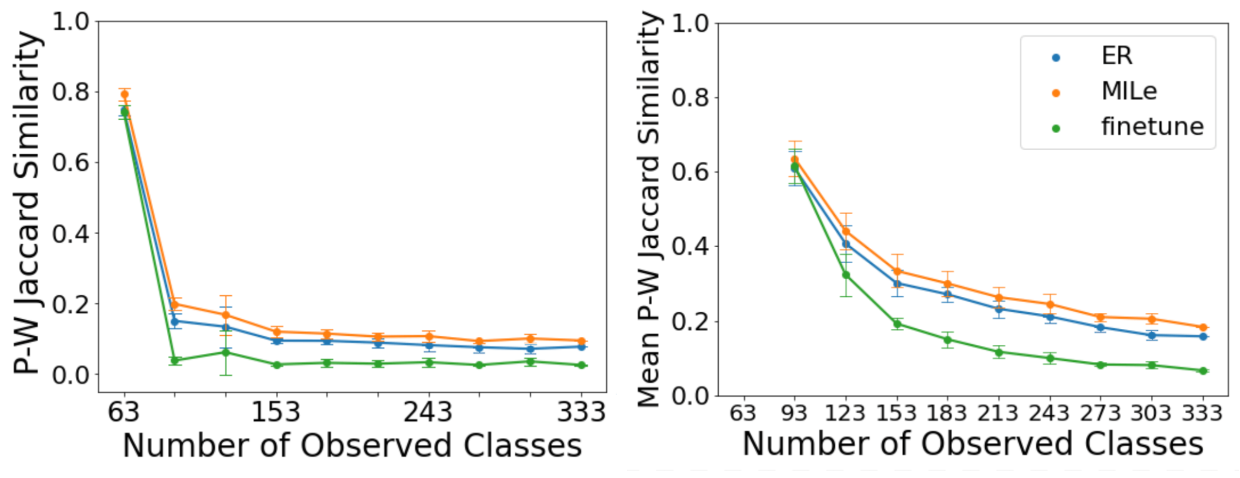
Following the procedure described by Abdelsalam et al. [2021], we train a ResNet-50 on ImageNet and a reduced ResNet-32 on CIFAR100. Also following Abdelsalam et al. [2021], we compare with an *experience replay* (ER) baseline and a *finetune* lower-bound. We report the model's overall performance after training until task $i$ as the precision-weighted Jaccard similarity between the model predictions and the ground-truth multi-labels over all classes encountered so far. We report IIRC-ImageNet-lite evaluation scores in Fig. B.2.1a and CIFAR in Fig. B.2.1b. In all cases, we find that iterative learning increases the performance with respect to the ER baseline by a constant factor. This suggests that MILe helps prevent forgetting previously seen labels by propagating them through the iterated learning procedure.

## B.3. Self-supervised implementation details

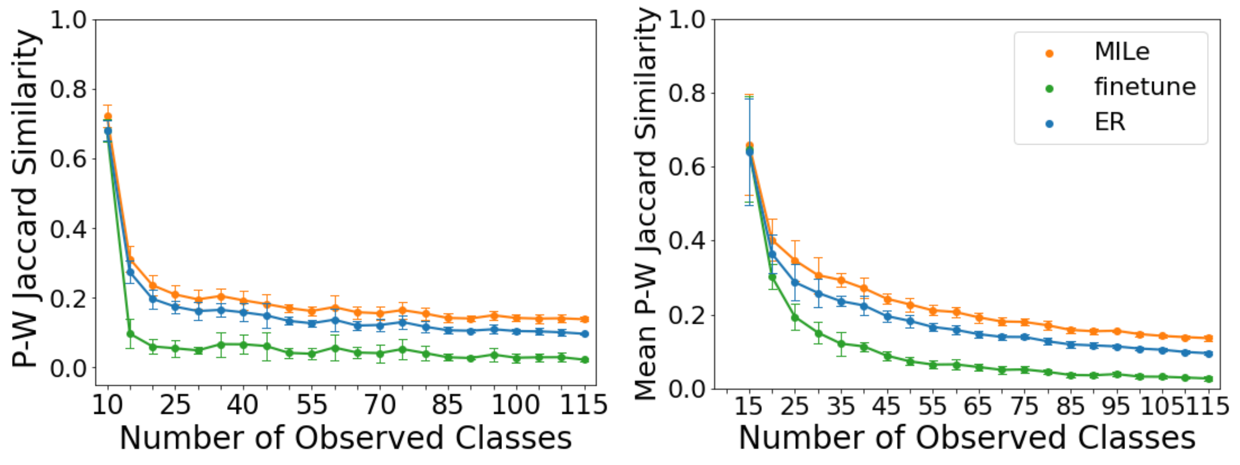
We obtained the performance of R50-2x+SK from Table 2 in the SimCLRv2 paper [Chen et al., 2020c]. As for MILe, we use our best model based on R50-1x+SK from Table 4, using the teacher to predict pseudo-labels on *the same* unlabeled set designed in the SimCLRv2 official codebase. The results for self-distilled SimCLRv2 with R50-1x+SK are obtained with the original ImageNet validation code and pre-trained model provided in the official SimCLRv2 repository.

## B.4. ReaL label recovery

The goal of MILe is to alleviate the problem of label ambiguity by recovering all the alternative labels for a given sample. We define alternative labels as those that were not



**(a)** IIRC-ImageNet



**(b)** IIRC-CIFAR10

**Fig. B.2.1. IIRC evaluation**. (a) Average performance on IIRC-ImageNet-lite. (b) Average performance on IIRC-CIFAR10. We run experiments on five different task configurations and report the mean and standard deviation. Left: average performance when the tasks are equally weighted irrespective of how many samples exist per task. Right: average performance over the number of samples. In this case, the first task has more weight since it is larger in the number of samples.

originally present in the ground truth. In this section, we evaluate how much of those alternative labels are recovered with MILe.

| Method | ResNet-50 | | ResNet-18 | |
|---|---|---|---|---|
| | 10% data | 100% data | 10% data | 100% data |
| Softmax | 0.2171 | 0.2679 | 0.1983 | 0.2648 |
| Sigmoid | 0.2310 | 0.2845 | 0.2047 | 0.2836 |
| MILe (ours) | **0.3042** | **0.3248** | **0.2187** | **0.2880** |

**Table B.4.1. Secondary label recovery.** Mean average precision over labels that appear in ReaL but not in the original ImageNet validation set.

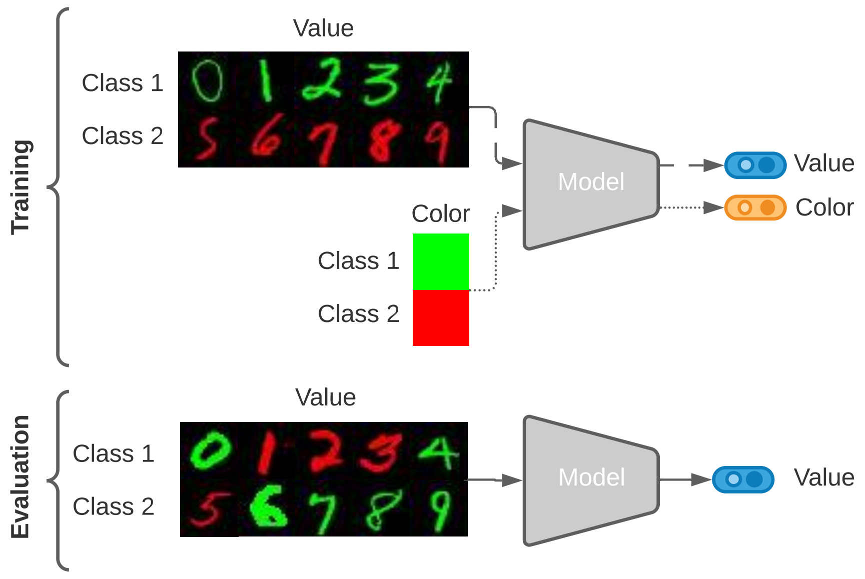


**Fig. B.4.2. ColoredMNIST+.** During training, the model is asked to classifier either digits or colors. Digits are highly correlated with their color, e.g. 0-4 tend to be green while 5-9 tend to be red. At test time, digits are less correlated with color.

| Method | CMNIST | CMNIST+ |
|---|---|---|
| ERM | 51.6± 0.1 | 51.1 ± 0.1 |
| IRM [Arjovsky et al., 2019] | 51.8± 0.1 | 51.2 ± 0.2 |
| REx [Krueger et al., 2020] | 51.6± 0.1 | 51.2 ± 0.2 |
| MILe (ours) | 51.8± 0.1 | **53.5** ± 0.6 |

**Table B.4.2. OOD generalization** on ColoredMNIST [Arjovsky et al., 2019] (CMNIST), which consists of predicting digits and ColoredMNIST+, which consists of color or digit prediction.

Table B.4.1 displays the mean average precision on the alternative labels present in ReaL [Beyer et al., 2020]. As it can be seen, MILe is able to recover up to 7% more labels than replacing softmax by sigmoid and binary cross entropy during training.

## B.5. Domain Generalization Experiments

A common problem of machine-learning models is that they tend to fail when presented with out-of-distribution data [Beery et al., 2018]. Arjovsky et al. [2019] claimed that this happens due to models relying on spurious correlations rather than the causal factors of the data. Thus, we investigate whether iterative learning can reduce the effect of spurious correlations by allowing the model to produce independent predictions of the two correlated factors. Following Arjovsky et al. [2019], we perform experiments on ColoredMNIST [Arjovsky et al., 2019], a version of MNIST where the color of the digits is spuriously correlated with

their value. The spurious correlation is removed at test time, i.e. colors are assigned randomly, to reveal whether models are affected by color. During training, data is sampled from two different image-label distributions or environments. In the first one, the correlation between digit and color is 90% and in the second is 80%. The correlation between the digit and color is 10% at test time. Since we want to explore the effect on generalization when the model is able to predict the digit and the color independently, we add a 33% chance of showing a blank image with no digit and only background color, where the background color is the label. This would be equivalent to a "beach" class in ImageNet. Note that this change does not remove the spurious correlations between the existing digits and their color. We call this benchmark ColoredMNIST+, see Fig. B.4.2. During training, iterated learning builds a multi-label representation of the digits, often including their color, leading to better disentanglement of the concepts "digits" and "color".
**Results.** We compare with invariant risk minimization (IRM) [Arjovsky et al., 2019] and risk extrapolation (REx) [Krueger et al., 2020] based on the DomainBed implementation [Gulrajani and Lopez-Paz, 2020]. These two approaches leverage differences between multiple environments, with different levels of correlation between digit and color, to become invariant to spurious attributes.We report results in Table B.4.2. MILe surpasses REx by 2 points. Interestingly, even though ERM and IRM are also required to solve the color classification task, only iterated learning is able to use it to improve performance. Although the color and digit prediction tasks are mutually exclusive, during iterated learning the teacher produces labels for both tasks simultaneously and thus the student learns to predict the color even for images that contain a digit. This helps the model to learn that these are two independent attributes, boosting its performance. In order to investigate how models perform outside of their original training distribution, Arjovsky et al. [2019] introduced ColoredMNIST, a dataset of digits presented in different colors. In order to create spurious correlations, the color of the digits is highly correlated with the value itself.

## B.6. Multi-label classification on CelebA

We provide results on CelebA [Liu et al., 2015], a multi-label dataset. CelebA is a large-scale dataset of facial attributes with more than 200K celebrity images, each with 40 attribute annotations that are known to be noisy [Speth and Hand, 2019]. We report results in Table B.6.3. Interestingly, despite the fact that CelebA is a multi-label dataset, we observe a $\sim 1\%$ improvement in F1 score when using the proposed iterative learning procedure. This along with per-class balanced accuracy in Table B.6.4 is in line with our hypothesis that the iterated learning bottleneck has a regularization effect that prevents the model from learning noisy labels [Lu et al., 2020a]. It is worth noting that MILe shows improved scores for the attributes that are difficult to classify such as *big-lips*, *arched-eyebrows* and *moustache*.

| Method | F1-score |
|---|---|
| CE-Sigmoid | 80.14 |
| ResNet-18(FPR) [Bekele and Lawson, 2019] | 77.55 |
| ResNet-34 (FPR) [Bekele and Lawson, 2019] | 79.96 |
| MILe (ours) | **81.40** |

**Table B.6.3.** Comparison on CelebA multi-attribute classification. Just as in ReaL ImageNet validation, we use F1-score (based on the intersection over union) measure to evaluate the methods.

| | *5 o Clock Shadow* | *Arched Eyebrows* | *Attractive* | *Bags Under Eyes* | *Bald* | *Bangs* | *Big Lips* | *Big Nose* | *Black Hair* | *Blond Hair* | *Blurry* | *Brown Hair* | *Bushy Eyebrows* | *Chubby* | *Double Chin* | *Eyeglasses* | *Goatee* | *Gray Hair* | *Heavy Makeup* | *High Cheekbones* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Triplet-kNN Schroff et al. [2015] | 66 | 73 | 83 | 63 | 75 | 81 | 55 | 68 | 82 | 81 | 43 | 76 | 68 | 64 | 60 | 82 | 73 | 72 | 88 | 86 |
| PANDA Zhang et al. [2014] | 76 | 77 | 85 | 67 | 74 | 92 | 56 | 72 | 84 | 91 | 50 | **85** | 74 | 65 | 64 | 88 | 84 | 79 | 95 | **89** |
| Anet Liu et al. [2015] | 81 | 76 | **87** | 70 | 73 | 90 | 57 | **78** | **90** | 90 | 56 | 83 | 82 | 70 | 68 | 95 | **86** | 85 | **96** | **89** |
| MILe | **85** | **83** | 82 | **74** | **82** | **92** | **65** | 74 | 88 | **91** | **76** | 79 | **83** | **72** | **72** | **98** | **86** | **86** | 93 | **89** |

| | *Male* | *Mouth Slightly Open* | *Mustache* | *Narrow Eyes* | *No Beard* | *Oval Face* | *Pale Skin* | *Pointy Nose* | *Receding Hairline* | *Rosy Cheeks* | *Sideburns* | *Smiling* | *Straight Hair* | *Wavy Hair* | *Wearing Earrings* | *Wearing Hat* | *Wearing Lipstick* | *Wearing Necklace* | *Wearing Necktie* | *Young* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Triplet-kNN Schroff et al. [2015] | 91 | 92 | 57 | 47 | 82 | 61 | 63 | 61 | 60 | 64 | 71 | 92 | 63 | 77 | 69 | 84 | 91 | 50 | 73 | 75 |
| PANDA Zhang et al. [2014] | **99** | 93 | 63 | 51 | 87 | 66 | 69 | 67 | 67 | 68 | 81 | **98** | 66 | 78 | 77 | 90 | **97** | 51 | **85** | 78 |
| Anet Liu et al. [2015] | **99** | **96** | 61 | 57 | 93 | **67** | **77** | 69 | 70 | **76** | 79 | 97 | 69 | 81 | 83 | 90 | 95 | **59** | 79 | **84** |
| MILe | **99** | 95 | **74** | **77** | **94** | 64 | 75 | **69** | **77** | 74 | **87** | 94 | **74** | **83** | **84** | **94** | 93 | 56 | 77 | 81 |

**Table B.6.4.** Mean per-class balanced accuracy in percentage points for each of the 40 face attributes on CelebA.

## B.7. Comparisons with Noisy Student Scheduling

Xie et al. [2020] introduced noisy student for labeling unlabeled data during semi-supervised learning. This is different from the goal of MILe, which is to construct a new multi-label representation of the images from single labels. Different from MILe, which trains a succession of short-lived teacher and student models, noisy student trains the model three times until convergence. This raises the question of how would MILe perform if it followed noisy student's iteration schedule instead of the one introduced in the main text.

In Fig. B.7.3 we compare the performance of the best MILe iteration schedule with the NS schedule. We found that MILe achieves the best performance in terms of the ReaL-F1 score.

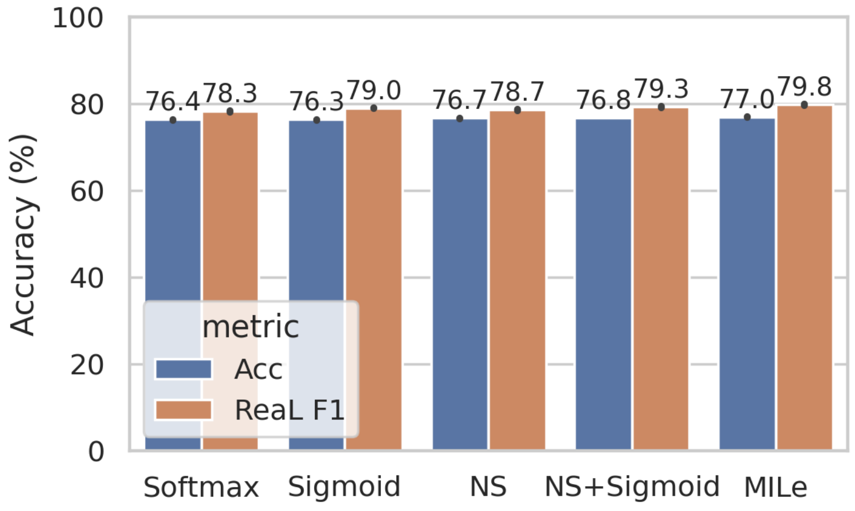


**Fig. B.7.3. Ablation study.** Comparison between different iteration schedules. (a) Comparison with noisy student (NS). (b)(c) Sweep over length of interactive learning phase $k_t$ and length of imitation phase $k_s$. We report the ReaL-F1 score for 10% (b) and 100% (c) data fraction.

# Appendix C

# Appendix for Chapter 8

In this Supplementary Material section, we provide additional details concerning various elements which could not be elaborated on in the main paper. We begin with a detailed description of the proposed MiniTouch benchmark. We outline the action space, state space, and reward structure for each of the tasks. This is followed by a closer look into the various aspects of the haptics-control module including architectures, experimental procedures, hyper-parameters, and additional results. We include an ablation experiment at the end that investigates the usefulness larger exploration phase while solving each of the tasks.

## C.1. MiniTouch Tasks

This section describes the cross-modal benchmark of simulated manipulation tasks, which we make use of and release as part of the paper. Unlike these prior simulation benchmarks, we particularly focus on providing a platform where one can use cross-modal information to solve diverse manipulation tasks. Existing benchmarks Yu et al. [2019] do not include touch modality. An overview of the tasks in MiniTouch is outlined in Figure 8.3. These tasks are built off Pybullet Coumans and Bai [2016–2019] physics engine and contain different scene setups for each of the four tasks. The details of each of these tasks are further expanded. All the four tasks are compiled together as a “MiniTouch" benchmark suitable for evaluating interaction-based algorithms.

**Playing:** This environment is intended as a toy task to evaluate interaction frequency and does not feature any reward beyond interaction count. A cube is placed in a random position on a table at each episode. The agent needs to localize and interact with the cube.

**Pushing:** In this task, the agent needs to push an object placed randomly on a table to a target (visually indicated as a gray cube). The object position is sampled uniformly in polar coordinates around the target object (i.e. angle 0 to 360 degrees, distance 10 to 20 centimeters). The end effector’s start position is sampled in the same way as the target position. In addition, the orientation of the gripper is fixed to be perpendicular to the ground all the time. The robot agent succeeds and receives a reward of +25 if the distance between the cube and the target object is less than 7 centimeters. A new episode starts if the agent succeeds. The environment also restarts if the cube is placed or pushed beyond a predefined bounding box comprising of acceptable positions on the table (*i.e.* positions that can be reached by the robot hand).

**Opening:** A cabinet with a door is randomly placed in reach of the agent. The goal is to find the door handle and open the door. The gripper orientation is fixed to point its fingers towards the door, parallel to the ground. For this task, the fingers are discretized to be open or closed. In addition, a fifth element is added to the action vector to control the yaw (relative rotation of the end-effector) to be able to approach the door. The robot succeeds and receives a reward of +25 when the angle of the door opening reaches thirty degrees or higher. Similar to the pushing task, a new episode starts if the agent reaches the goal.

**Pick-up:** In this environment, the agent needs to grasp and lift a randomly placed object. The agent's goal is to lift the object 5cm above the table. The agent receives a reward of +25 upon success. The object is placed uniformly randomly on a table. Similar to the Opening task, the end effector opening/closing is discretized, meaning when its internal continuous variable is below a threshold, the gripper closes, otherwise it remains open.

All of the tasks are implemented in the Pybullet physics engine Coumans and Bai [2016–2019], which is a free and open-source library that enables fast simulation.

### C.1.1. *MiniTouch* Library:

The task environment used in the experiments is packaged and released as a python library that can be easily plugged into the training code. Setup instructions, code, and other details can be found in the README file included in the repository.

### C.1.2. HaC and baselines:

We used the following open-source implementations for the baselines. We were able to reproduce the results from their papers before attempting to use them as baselines for our model:

**ICM**

https://github.com/openai/large-scale-curiosity

**Disagreement**:

https://github.com/danijar/dreamerv2

https://github.com/pathak22/exploration-by-disagreement

**RND**:

https://github.com/openai/random-network-distillation

https://github.com/ElementAI/MiniTouch

## C.2. Experimental Details

**State space:** The input states are a combination of visual and touch vector input. The visual input is a grayscale rendering of the scene with dimension $84 \times 84$, pre-processed similarly to Mnih et al. [2016]. Image observations are captured from a static camera overlooking each scene. The touch vector input is composed of the 3-dimensional end-effector position, 2-dimensional finger position, ranging from 0 to 1, each denoting how far apart each finger is, and the 6-dimensional force/torque values. In total, the touch vector is 11-dimensional, $S \in \mathbb{R}^{11}$.

**Action space:** Actions are expressed as 4-dimensional continuous vectors. The first three elements describe the desired offset for the end effector at the next timestep based on its current position. The last dimension controls the relative desired distance between the two fingers at the next timestep.

Training. We use SAC Haarnoja et al. [2018] as the optimizer for our agent. Our training is composed of two phases as described in the main paper. (i)Exploration phase, (ii)Downstream task phase. In the exploratory phase (curiosity part) agent is trained using our intrinsic reward alone. In the task phase, network weights are seeded from the ones in the exploratory phase. We also retain the replay buffer from the exploratory phase in the downstream task phase. Duration of the exploration phase can be adjusted in the code using a hyper-parameter *stop-curiosity*. We have two hyper-parameters that change between the two phases, (i) $\alpha$ and (ii) the learning rate of the SAC algorithm. Details of hyper-parameters used for our experiments are outlined in Table C.2.1 and Table C.2.2.

**SAC pretraining and training hyperparameters**

| Parameter type | Value |
|---|---|
| optimizer | Adam |
| Visual network | Table C.2.2 |
| learning rate | $3.10^{-5}$ |
| number of samples per minibatch | 128 |
| reward scale | 100 |
| replay buffer size | $10^6$ |
| number of hidden units per layer | 128 |
| number of hidden layers (all networks) | 2 |
| activation | LeakyReLU |
| discount factor | 0.99 |

**Table C.2.1.** SAC parameters during pretraining and training.

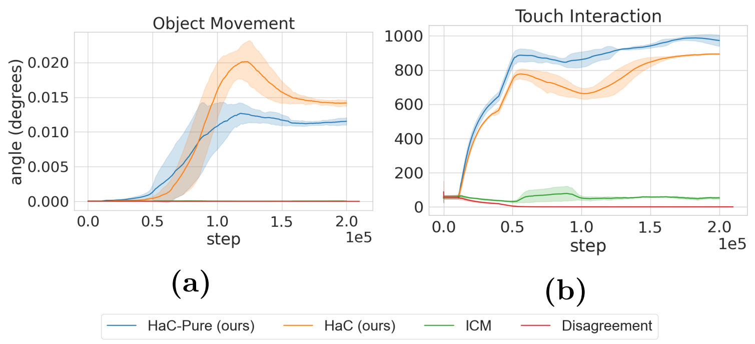


**Fig. C.3.1. Object Interaction for Opening task.** (a) shows the average variance in door angle across the entire episode (note that the absolute variance is low but corresponds to successful door openings towards the end of training for HaC) and in (b), we count the number of touch interactions in the same task.

| **Encoder network** | | | | |
|---|---|---|---|---|
| Layer | Number of outputs | Kernel size | Stride | Activation function |
| Input $x$ | $84 * 84 * 1$ | | | |
| Convolution | 20*20*32 | $8 * 8$ | 4 | LeakyReLU |
| Convolution | 8*8*64 | $4 * 4$ | 2 | LeakyReLU |
| Convolution | 4*4*124 | $3 * 3$ | 1 | LeakyReLU |
| Convolution | 2*2*256 | $2 * 2$ | 1 | LeakyReLU |
| Fully-connected | 256 | 1 | | LeakyReLU |

**Table C.2.2.** Visual network.

## C.3. Object Interaction

As touched up in the experiments section of the main paper, Figure C.3.1 depicts the touch interaction and door movement metrics for the Opening task. We make a similar observation to the result showed in Figure 8.4 for the Playing task. A higher touch-interaction need not indicate better object-movement. Agent can resort to constantly engaging with the object in a passive manner. HaC collects rich interaction data during the exploratory phase and helps the agent in terms of sample efficiency while solving the downstream tasks.

## C.4. Additional Ablations

**Does longer exploration help?** We observe that having a longer exploratory phase of training with intrinsic reward alone usually benefits the overall performance. We can observe from Figure 8.5 that HaC attains decent success in the exploratory phase without any external reward on most of the tasks. This is because it encourages better associations and a larger collection of interesting configurations in the replay. The effect of the exploratory step is further studied and the results on all of the downstream tasks with different duration

| Metric | Pushing | | | Open Door | | | Pick-up | | | Playing | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HaC | RND+*touch* | RND | HaC | RND+*touch* | RND | HaC | HaC-*RND* | RND | HaC | RND+*touch* | RND |
| **Exploration ↑** | **0.403** | 0.181 | 0.077 | **0.669** | 0.564 | 0.380 | **0.063** | 0.013 | 0.005 | - | - | - |
| **Success ↑** | **0.733** | 0.710 | 0.582 | **0.983** | 0.921 | 0.875 | **0.891** | 0.593 | 0.450 | - | - | - |
| **Episode steps ↓.** | **57.84** | 55.20 | 97.51 | **23.34** | 23.54 | 37.92 | **30.54** | 52.29 | 89.88 | - | - | - |
| **Touch-interaction ↑** | **247.79** | 227.75 | 206.02 | **600.1** | 411.43 | 194.22 | **980.7** | 894.5 | 725.1 | **388.15** | 312.0 | 124.5 |

**Table C.4.3. Random Network Distillation (RND) with touch** This table compares the mean evaluations for HaC and RND+*touch* on all the four tasks emphasizing the importance of the cross-modal association(see text). We omit measuring success and episode steps for playing task since success there is equivalent to object-interaction and has no explicit goal.

.

of exploration are compiled in Figure C.4.2. We depict the episode convergence steps and success rate to visualize the trend as expected.

**Touch in Random Network Distillation** We created an additional baseline RND+*touch* similar to touch in future prediction (ICM+*touch*) baseline. In this setting, the input to the random target network and predictor network (that predicts the outcome of target network) consists of both touch and visual feature vector concatenated. The baseline beats the vision only version by a margin for both Pushing and picking tasks as shown in the Table. C.4.3, however for the Open-door tasks the improvement is not that significant. Although RND+*touch* is comparable with HaC for the Pushing task on the success rate, HaC has better sample efficiency as measured by episode steps. We believe this is because object movement is crucial for the pushing task compared to interaction, and prediction based techniques such as RND could be helpful in such settings.

**Comparisons with Generalization to Novel shapes:** It is desirable for an agent to be able to handle diverse shapes in order to be robust across arbitrary manipulation settings. We study this using an environment in which an object is sampled from a thousand procedurally generated objects. The objects are dissimilar with respect to shape and mass but are sampled from the same generative distribution. Out of 1000 different objects, 800 of them are used in the training phase, and we evaluate the agent's effectiveness on the remaining 200 unseen object shapes.

Although generalization to novel shapes is not the problem setting our method focuses on. We perform this proof of concept ablations to investigate that HaC parameters in the exploratory phase are not just reusable across different tasks with similar shapes (Eg: Playing, Pushing, and Picking), but also can help generalize when applied to distinctive shapes sampled from a similar distribution. Figure C.4.3 shows touch interaction and object movement evaluations for a single object exploration task. We compare HaC with ICM and Disagreement baselines. . The results validate that our model generalizes to unseen object configurations.

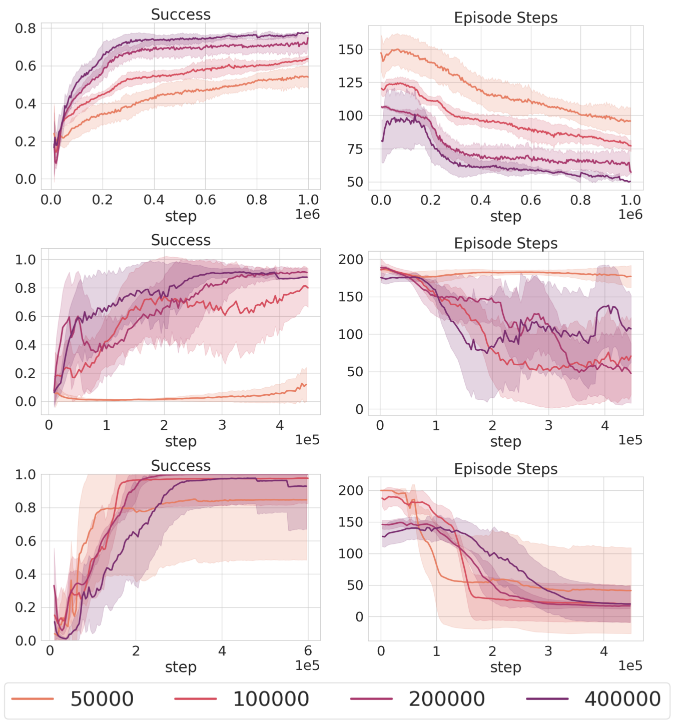
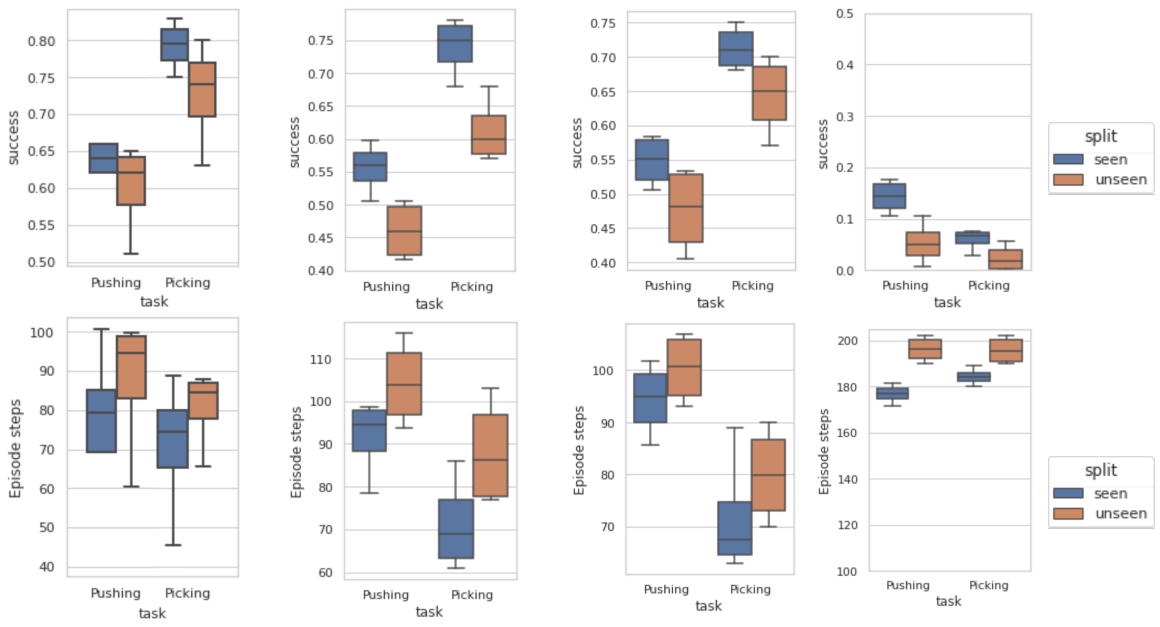


**Fig. C.4.2. Longer exploratory phase helps** Success and episode steps evaluations on different tasks for different lengths of *exploratory* phase. Darker shading indicates longer exploration.

## C.5. Real world use cases:

It is possible to define safety boundaries for a robot arm (e.g. don't hit the table, don't move outside the arena boundaries) and have a robot arm autonomously and task-independently collect data for phase 1 of our method. In addition to this, we would like to point out that (a) in the pushing and pick-up tasks, only 5-10k steps are required for decent performance on the downstream task and (b) what we call a "step" is however long it takes the inverse kinematic solver to move the arm to the new end effector pose (that was generated by the policy). If many of the generated poses cluster together, that dramatically reduces runtime of the curiosity phase making it more suitable to real-world applicability. While no real robot data have been used in the experiments, our method is data agnostic. We note that the physics engine can be replaced by real-world data as there is no bias introduced in process. The networks used for the curiosity prediction task can be easily adapted, if needed, to accommodate inputs from a real robotic platform. As a next step and when restrictions are eased, we plan to implement this method on a physical system and replicate our experiments.



(a) HaC (b) ICM (c) Disagreement (d) SAC

**Fig. C.4.3. Generalization to novel shapes baseline comparisons**. Each column outlines Success and Episode steps for HaC, ICM, Disagreement and SAC baselines respectively. Disagreement has better generalization and reduced variance compared to ICM on the unseen categories. HaC is comparable to Disagreement on generalization performance while displaying better success and Episode steps rates.

# Appendix D

# Appendix for Chapter 10

## D.1. Reference scores

We report the performance of the *Dreamer@2M* oracle agent, trained with environment rewards on the twelve tasks of the URL benchmark. In Figure D.1.1, We show normalized performance to compare to the DrQ-v2's scores from Laskin et al. [2021], which can be found at: `https://github.com/rll-research/url_benchmark/issues/1`.

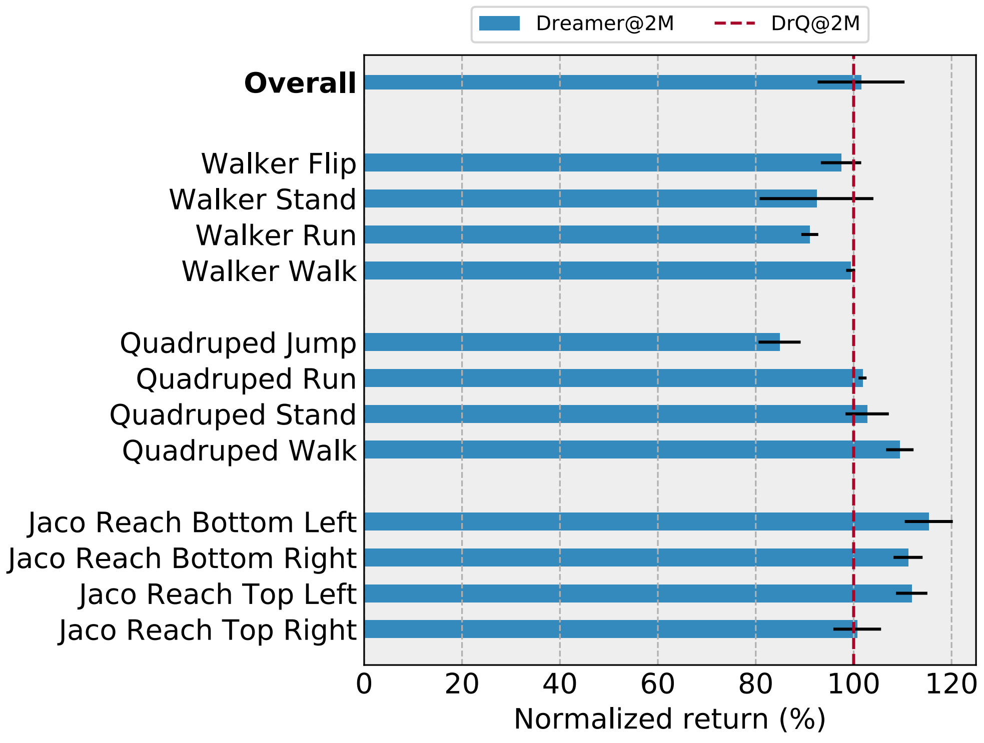


**Fig. D.1.1. Dreamer oracle normalized scores.** Dreamer scores on the URLB tasks normalized by the scores of DrQ-v2 [Laskin et al., 2021].

In Table D.1, we report the mean scores for DrQ-v2, used as reference scores in the URLB paper, and for Dreamer@2M, which we use to normalize returns in our work. We additionally report mean and standard deviations for the best performing baseline from URLB. which is Disagreement [Pathak et al., 2019b], and our best performing method, which is LBS+TD-MPC (with no actor initialization for Jaco). We notice that the LBS+TD-MPC scores approach the Dreamer@2M's scores in several tasks, eventually outperforming them in a few tasks (e.g. Walker Flip, Quadruped Jump). We believe this merit of LBS+TD-MPC is due both to the exploration pre-training, which may have found more rewarding trajectories than greedy supervised RL optimization, and of the improved planning strategy [Hansen et al., 2022].

| Pre-trainining for 2M environment steps | | | | | |
|---|---|---|---|---|---|
| Domain | Task | URLB Expert | URLB Disagreement | Dreamer@2M | LBS+TD-MPC |
| Walker | Flip | 799 | $346 \pm 13$ | 778 | $938 \pm 12$ |
| | Run | 796 | $208 \pm 15$ | 724 | $596 \pm 38$ |
| | Stand | 984 | $746 \pm 34$ | 909 | $973 \pm 14$ |
| | Walk | 971 | $549 \pm 37$ | 965 | $959 \pm 1$ |
| Quadruped | Jump | 888 | $389 \pm 62$ | 753 | $822 \pm 33$ |
| | Run | 888 | $337 \pm 30$ | 904 | $642 \pm 99$ |
| | Stand | 920 | $512 \pm 89$ | 945 | $927 \pm 28$ |
| | Walk | 866 | $293 \pm 37$ | 947 | $816 \pm 61$ |
| Jaco | Reach bottom left | 193 | $124 \pm 7$ | 222 | $225 \pm 6$ |
| | Reach bottom right | 203 | $115 \pm 10$ | 225 | $221 \pm 10$ |
| | Reach top left | 191 | $106 \pm 12$ | 213 | $226 \pm 5$ |
| | Reach top right | 223 | $139 \pm 7$ | 224 | $227 \pm 2$ |

**Table D.1.1.** Performance of expert baseline and the best method on pixel-based URLB from Laskin et al. [2021] and performance of our oracle baseline (Dreamer@2M) and best approach (LBS+TD-MPC), after pre-training for 2M steps and fine-tuning for 100k steps.

In Section 10.4.2, we presented results in the RWRL benchmark using the normalization scores of the vanilla environments, as in Table D.1.1. However, the scores of the oracle may be lower on the more difficult RWRL tasks. In Figure D.1.2, we present the same results re-normalized by the performance of Dreamer@2M trained on each of the 8 tasks (2 vanilla + 6 RWRL tasks on different intensity settings). Other than better highlighting the findings presented in Section 10.4.2, the results show that on the Walker Easy and on the Quadruped Easy and Medium settings our method recovers $> 80\%$ of the supervised baseline performance, while using 20x less task-specific data. This strengthens the hypothesis that our method can be used to transfer efficiently to more realistic settings.

## D.2. Unsupervised Reinforcement Learning Strategies

We summarize the unsupervised RL approaches adopted in our work. For all approaches, rewards have been normalized during training using an exponential moving average with momentum 0.95.

**ICM.**. The Intrinsic Curiosity Module (ICM; Pathak et al. [2017b]) defines intrinsic rewards as the error between states projected in a feature space and a feature dynamics model's predictions. We use the Dreamer agent encoder $e_t = f_\phi(s_t)$ to obtain features and train a forward dynamics model $g(e_t|e_{t-1}, a_{t-1})$ to compute rewards as:

$$r_t^{\text{ICM}} \propto \|g(e_t|e_{t-1}, a_{t-1}) - e_t\|^2.$$

As ICM requires environment states to compute rewards, we train a reward predictor to allow estimating rewards in imagination.

**Plan2Explore.** The Plan2Explore algorithm [Sekar et al., 2020] is an adaptation of the Disagreement algorithm [Pathak et al., 2019b] for latent dynamics models. An ensemble of forward dynamics models is trained to predict the features embedding $e_t = f_\phi(s_t)$, given the previous latent state and actions, i.e. $g(e_t|z_{t-1}, a_{t-1}, w_k)$, where $w_k$ are the parameters of the k-th predictor. Intrinsic rewards are defined as the variance of the ensemble predictions:

$$r_t^{\text{P2E}} \propto \text{Var}(\{g(e_t|z_{t-1}, a_{t-1}, w_k)|k \in [1, ..., K]\}).$$

Plan2Explore requires only latent states and actions, thus it can be computed directly in imagination. We used an ensemble of 5 models.

**RND.**. Random Network Distillation (RND; Burda et al. [2019b]) learns to predict the output of a randomly initialized network $n(s_t)$ that projects the states into a more compact random feature space. As the random network is not updated during training, the prediction

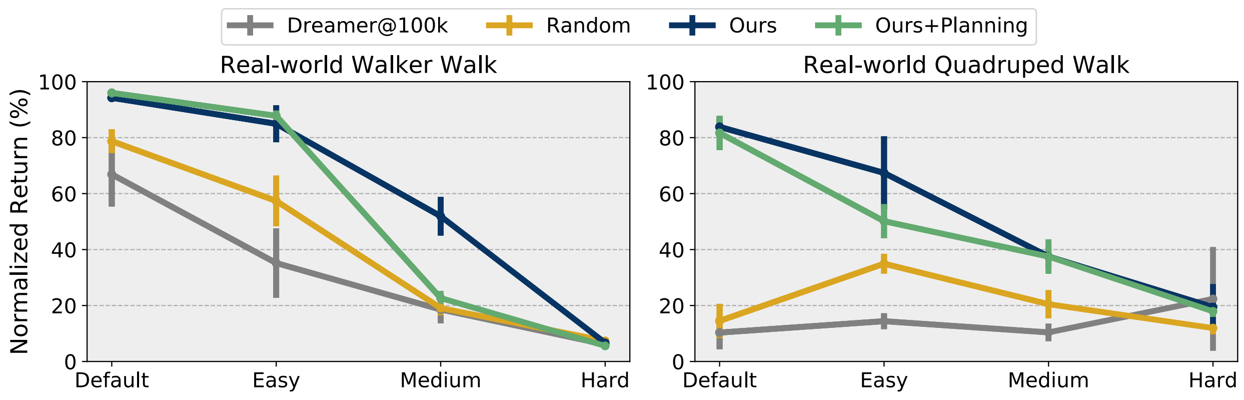


**Fig. D.1.2. Results on RWRL re-normalized.** Results on RWRL normalized by the scores of Dreamer@2M trained on each of the RWRL tasks. Models are pre-trained on the vanilla version of the environment for 2M steps and fine-tuned for 100k steps on the perturbated tasks from RWRL.

error should diminish for already visited states. Intrinsic reward here is defined as:

$$r_t^{\text{RND}} \propto \|g(s_t) - n(s_t)\|^2$$

As RND requires environment states to compute rewards, we train a reward predictor to allow estimating rewards in imagination.

**LBS.**. In Latent Bayesian Surprise (LBS; Mazzaglia et al. [2021]), they use the KL divergence between the posterior and the prior of a latent dynamics model as a proxy for the information gained with respect to the latent state variable, by observing new states. Rewards are computed as:

$$r_t^{\text{LBS}} \propto D_{\text{KL}}[q(z_t|z_{t-1}, a_{t-1}, e_t)\|p(z_t|z_{t-1}, a_{t-1})]$$

As LBS requires environment states to compute rewards, we train a reward predictor to allow estimating rewards in imagination.

**APT.**. Active Pre-training (APT; Liu and Abbeel [2021b]) uses a particle-based estimator based on the K nearest-neighbors algorithm Singh et al. [2003] to estimate entropy for a given state. We implement APT on top of the deterministic component of the latent states $\bar{z}_t$, providing rewards as:

$$r_t^{\text{APT}} \propto \sum_i^k \log \|\bar{z}_t - \bar{z}_t^i\|^2,$$

where $k$ are the nearest-neighbors states in latent space. As APT requires only latent states, it can be computed directly in imagination. We used $k = 12$ nearest neighbors.

**DIAYN.**. Diversity is All you need (DIAYN; Eysenbach et al. [2019]) maximizes the mutual information between the states and latent skills $w$. We implement DIAYN on top of the latent space of Dreamer, writing the mutual information as $I(w_t, z_t) = H(w_t) - H(w_t|z_t)$. The entropy $H(w_t)$ is kept maximal by sampling $w_t \sim p(w_t)$ from a discrete uniform prior distribution, while $H(w_t|z_t)$ is estimated learning a discriminator $q(w_t|z_t)$. Additionally, DIAYN maximizes the entropy of the actor, so we compute intrinsic rewards as:

$$r_t^{\text{DIAYN}} \propto \log q(w_t|z_t) - \log \pi(a_t|z_t)$$

As DIAYN requires environment states and sampled skills to compute rewards, we train a reward predictor to allow estimating rewards in imagination.

## D.3. Algorithm

**Algorithm 3** Model-based Unsupervised RL

```
Require: Actor $\theta$, Critic $\psi$, World Model $\phi$
Require: Intrinsic reward $r^{\text{int}}$, extrinsic reward $r^{\text{ext}}$
Require: Environment, $M$, downstream tasks $T_k$, $k \in [1, \dots, M]$
Require: Pre-train steps $N_{\text{PT}}$, fine-tune steps $N_{\text{FT}}$, environment steps/update $\tau$
Require: Initial model state $z_0$, hybrid planner Plan, replay buffers $\mathcal{D}_{\text{PT}}$, $\mathcal{D}_{\text{FT}}$
 1: for $t = 0, \dots, N_{\text{PT}}$ do
 2:     Draw action from the actor, $\mathbf{a}_t \sim \pi_\theta(a_t|z_t)$
 3:     Apply action to the environment, $\mathbf{s}_{t+1} \sim P(\cdot|\mathbf{s}_t, \mathbf{a}_t)$
 4:     Add transition to replay buffer, $\mathcal{D}_{\text{PT}} \leftarrow \mathcal{D}_{\text{PT}} \cup (\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$
 5:     Infer model state, $z_{t+1} \sim q(z_{t+1}|z_t, a_t, f_\phi(s_{t+1}))$
 6:     if $t \mod \tau = 0$ then
 7:         Update world model parameters $\phi$ on the data from the replay buffer $\mathcal{D}_{\text{PT}}$
 8:         Update actor-critic parameters $\{\theta, \psi\}$ in imagination, maximizing $r^{\text{int}}$
 9:     end if
10: end for
11: Output pre-trained parameters $\{\psi_{\text{PT}}, \theta_{\text{PT}}, \phi_{\text{PT}}\}$
12: for $T_k \in [T_1, \dots, T_M]$ do
13:     Initialize fine-tuning world-model with $\phi_{\text{PT}}$
14:     (Optional) Initialize fine-tuning actor with $\theta_{\text{PT}}$
15:     for $t = 0, \dots, N_{\text{FT}}$ do
16:         Use the planner for selecting action, $\mathbf{a}_t \sim \text{Plan}(z_t)$
17:         Apply action to the environment, $\mathbf{s}_{t+1}, r^{\text{ext}}_t \sim P(\cdot|\mathbf{s}_t, \mathbf{a}_t)$
18:         Add transition to replay buffer, $\mathcal{D}_{\text{FT}} \leftarrow \mathcal{D}_{\text{FT}} \cup (\mathbf{s}_t, \mathbf{a}_t, r^{\text{ext}}_t, \mathbf{s}_{t+1})$
19:         Infer model state, $z_{t+1} \sim q(z_{t+1}|z_t, a_t, f_\phi(s_{t+1}))$
20:         if $t \mod \tau = 0$ then
21:             Update world model parameters $\phi$ on the data from the replay buffer $\mathcal{D}_{\text{FT}}$
22:             Update actor-critic parameters $\{\theta, \psi\}$ in imagination, maximizing $r^{\text{ext}}$
23:         end if
24:     end for
25:     Evaluate performance on $T_k$
26: end for
```

## D.4. Hyperparameters

Most of the hyperparameters we used for world-model training are the same as in the original DreamerV2 work [Hafner et al., 2021]. Specific details are as outline here:

| **Name** | **Value** |
|---|---|
| World Model | |
| Batch size | 50 |
| Sequence length | 50 |
| Discrete latent state dimension | 32 |
| Discrete latent classes | 32 |
| GRU cell dimension | 200 |
| KL free nats | 1 |
| KL balancing | 0.8 |
| Adam learning rate | $3 \cdot 10^{-4}$ |
| Slow critic update interval | 100 |
| Actor-Critic | |
| Imagination horizon | 15 |
| $\gamma$ parameter | 0.99 |
| $\lambda$ parameter | 0.95 |
| Adam learning rate | $8 \cdot 10^{-5}$ |
| Actor entropy loss scale | $1 \cdot 10^{-4}$ |
| TD-MPC | |
| Iterations | 12 |
| Number of samples | 512 |
| Number of elite actions | 64 |
| Mixture coefficient (Actor/CEM) | 0.05 |
| Min std (fixed) | 0.1 |
| Temperature | 0.5 |
| Momentum | 0.1 |
| Horizon | 5 |
| Common | |
| Environment steps/update | 5 |
| MLP number of layers | 4 |
| MLP number of units | 400 |
| Hidden layers dimension | 400 |
| Adam epsilon | $1 \cdot 10^{-5}$ |
| Weight decay | $1 \cdot 10^{-6}$ |
| Gradient clipping | 100 |

**Table D.4.2.** World model, actor-critic, planner (TD-MPC) and common hyperparameters.

For the pure MPC-based experiments, we increased the number of MPPI samples from 512 to 1000, the number of elite actions from 64 to 100, and the horizon from 5 to 15.

## D.5. Additional Results

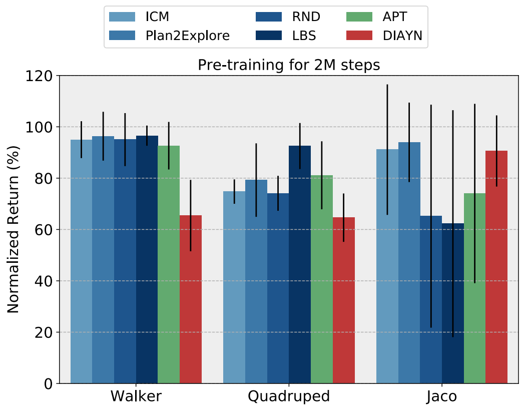
We present complete results, for each unsupervised RL method, for the experiments in Section 10.4, when using only the actor-critic algorithm, in Figure D.6.3, and when also employing the hybrid planner, in Figure D.6.4.

## D.6. RWRL settings

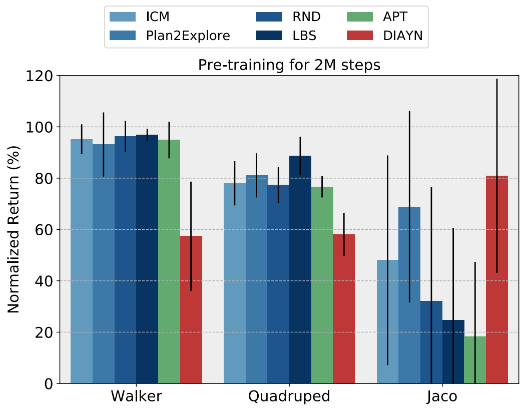
We take the Quadruped and Walker tasks from the RWRL benchmark and replace the low-dimensional sensor inputs with RGB camera inputs. While this removes some of the perturbations planned in the benchmark [Dulac-Arnold et al., 2020a], such as noise in observation space, it introduces the difficulty of a different dynamics in pixel space (due to the other perturbations), compared to the one observed during pre-training in the vanilla simulation environment.

| **Setting** | **Easy** | | **Medium** | | **Hard** | |
|---|---|---|---|---|---|---|
| **System Delays** | *Time Steps* | | *Time Steps* | | *Time Steps* | |
| Action | 3 | | 6 | | 9 | |
| Rewards | 10 | | 20 | | 40 | |
| **Action Repetition** | 1 | | 2 | | 3 | |
| **Gaussian Noise** | *Std. Deviation* | | *Std. Deviation* | | *Std. Deviation* | |
| Action | 0.1 | | 0.3 | | 1.0 | |
| **Perturbation Quadruped** | *[Min,Max]* | *Std.* | *[Min,Max]* | *Std.* | *[Min,Max]* | *Std.* |
| (shin length) | [0.25, 0.3] | 0.005 | [0.25, 0.8] | 0.05 | [0.25, 1.4] | 0.1 |
| **Perturbation Walker** | *[Min,Max]* | *Std.* | *[Min,Max]* | *Std.* | *[Min,Max]* | *Std.* |
| (thigh length) | [0.225, 0.25] | 0.002 | [0.225, 0.4] | 0.015 | [0.15, 0.55]] | 0.04 |

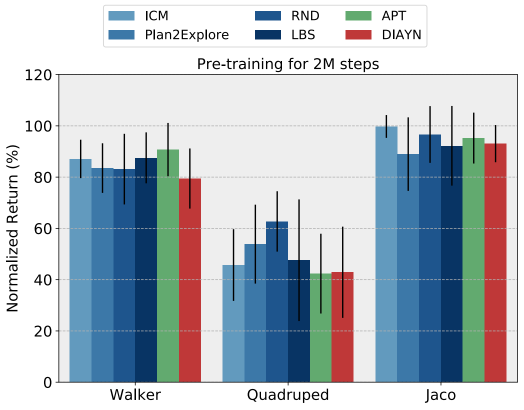
**Table D.6.3.** Perturbations setting for each challenge of our adapted tasks from the RWRL benchmark, in increasing levels of intensity.



**(a)** Default

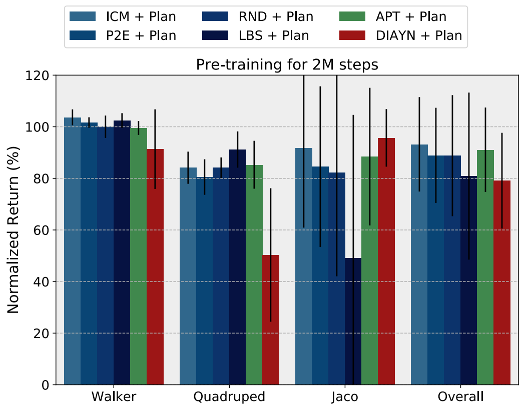


**(b)** w/ PT Critic

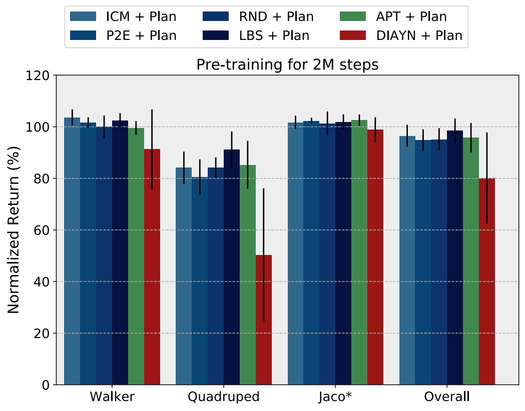


**(c)** w/o PT Actor

**Fig. D.6.3.** Results for all unsupervised approaches, when using actor-critic for action selection.



**(a)** Default + Plan



**(b)** Jaco w/o PT Actor + Plan

**Fig. D.6.4.** Results for all unsupervised approaches, when using the hybrid planner.