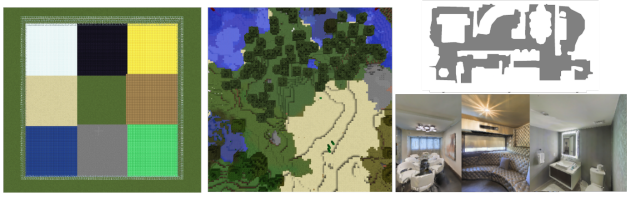
PixelEDL: Unsupervised Skill Discovery and Learning from Pixels

Roger Creus Castanyer[1] Juan José Nieto[1] Xavier Giro-i-Nieto[1,2,3]
[1]*Universitat Politècnica de Catalunya* [2]*Barcelona Supercomputing Center* [3]*Institut de Robòtica i Informàtica Industrial, CSIC-UPC*
creus99@protonmail.com juanjo.3ns@gmail.com xavier.giro@upc.edu

1. Introduction

We tackle embodied visual navigation in a task-agnostic set-up by putting the focus on the unsupervised discovery of skills (or options [2]) that provide a good coverage of states. Our approach intersects with empowerment [10]: we address the reward-free skill discovery and learning tasks to discover *what* can be done in an environment and *how*. For this reason, we adopt the existing Explore, Discover and Learn (EDL) [1] paradigm, tested only in toy example mazes, and extend it to pixel-based state representations available for embodied AI agents. The information-theoretic paradigm of EDL [1] aims to learn latent-conditioned policies, namely skills $\pi(a|s,z)$, by maximizing the mutual information (MI) between the inputs s and some latent variables z. Hence, EDL [1] consists of unsupervised skill discovery and training of reinforcement learning (RL) agents without considering the existence of any extrinsic motivation or reward. We present *PixelEDL*, an implementation of the EDL paradigm for the pixel representations provided by the AI Habitat [11] and MineRL [3] environments. In comparison with EDL, PixelEDL involves self-supervised representation learning of image observations for information-theoretic skill discovery. Still, PixelEDL aims to the maximize the MI between inputs and some latent variables and for that it consists of the same three stages of EDL (explore, discover and learn).

Figure 1. Top-down views of the three considered environments: (i) a custom "toy example" Minecraft map, (ii) a Realistic Minecraft map, and (iii) a Habitat apartment.



By breaking down the RL end-to-end training pipeline into the three stages, we also simplify the implicit difficulty in learning both representations and policies from a high dimensional input space all at once [7].

The results presented in this extended abstract are further extended in our project site[1].

2. Methodology

We assume an underlying Markov Decision Process (MDP) without rewards: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$ where $\mathcal{S}$ is the high-dimensional set of states (defined by image pixels). $\mathcal{A}$ is the action space and $\mathcal{P} = p(s|s,a)$ is the transition function. Moreover, we define the objective of PixelEDL as the maximization of the MI in equation (1), which requires knowledge of the unknown distributions $p(s), p(s|z), p(z|s)$.

$$\begin{aligned} \mathcal{I}(S,Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

2.1. Exploration

The first task to tackle in PixelEDL is exploration. Without any prior knowledge, a reasonable choice for discovering state-covering skills is to define the distribution over all states $p(s)$ uniformly. However, training an exploration policy to infer a uniform $p(s)$ is not feasible in PixelEDL since it deals with the high-dimensional pixel space. To overcome this limitation we adopt a non-parametric estimation of $p(s)$ by sampling from a dataset of collected experience. Hence, in PixelEDL the goal of the exploration stage is to collect a dataset of trajectories containing representative states that the learned skills should ultimately cover. PixelEDL adopts a random exploration of the environment through agents that perform random actions within a discrete action space (i.e. move forward, turn left, turn right) and collect the trajectories generated by the environment. For our custom Minecraft map, random policies from agents instantiated in the center of the map are capable of covering

[1] https://imatge-upc.github.io/PixelEDL/

a complete set of representative states of the environment given a large number of episodes. In the realistic Minecraft map the random agents do not cover as many representative states as in the custom map but still provide enough coverage of the state space. However, in order to obtain a set of representative states of the Habitat apartment we let the agents start in different random navigable points of the map at each episode.

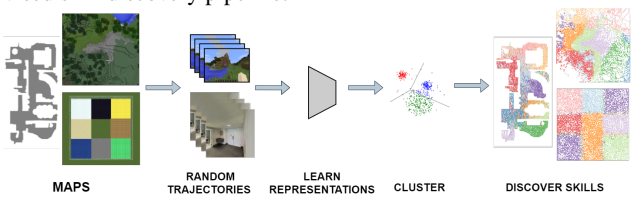
2.2. Skill Discovery

The Discovery stage of EDL aims at finding the latent representations z that will ultimately condition the agent policies to learn the skills $\pi(a|s, z)$. Hence, the goal of PixelEDL in this stage is to model $p(z|s)$ as a mapping of the states to their representations and to model $p(z)$ as a categorical distribution of meaningful representations.

Ideally, we aim to obtain representations of the image observations that encode existing similarities and spatial relations within the environment [6]. Furthermore, we aim to find z that are representatives of a meaningful segmentation of the state space. In this work the representations z will be later used to condition a navigation task. Previous works [16] have reported the challenges of unsupervised learning of representations from images that encode valuable features for RL agents in a 3D environment. For modelling $p(z|s)$, we study the performance of two different approaches: (i) a *contrastive* one, that uses a siamese architecture and aims to project positive pairs of input images closer in an embedding space, and (ii) a *reconstruction* one, that use a Variational Autoencoder (VAE) [5] with categorical classes, namely Vector Quantisation VAE (VQ-VAE) [9], to train the model to reconstruct the observations. For the contrastive approach, we use the adaptation to CURL [14] proposed by Stooke et al. [15], namely Augmented Temporal Contrast (ATC). Compared to CURL, in ATC the positive pairs of inputs consist of two image observations belonging to the same exploration trajectory. That is, we train both ATC and VQ-VAE so that a positive pair of inputs consists of two observations of the same trajectory with a delay $d \sim \mathcal{N}(\mu, \sigma^2)$. We experiment with $\mu = 15$ and $\sigma = 5$. Hence, in both ATC and VQ-VAE we perform a data augmentation in the temporal domain. Our experiments indicate that the capabilities of both ATC and VQ-VAE for modelling $p(z|s)$ are promising and we have not yet observed important differences to justify using one over the other.

After the visual representation learning, we model a categorical distribuiton $p(z)$ by clustering the embedding space of the representations. Yarats et. al [17] use a projection of the embeddings onto the prototypes which define a basis of the embedding space to perform the cluster assignments. However, in VQ-VAE, this clustering is implicit in the model since the cluster centroids are actually the representatives of the model's codebook. Also, for ATC we apply a K-means [8] clustering for modelling $p(z)$ with the cluster centroids. After modelling $p(z)$ and $p(z|s)$, we complete the stages of representation learning and skill discovery. Figure 2 summarises the aforementioned pipeline. We provide more details in our project site.

Figure 2. Self-Supervised representation learning and unsupervised skill discovery pipeline.



2.3. Learning

Given a model of $p(z)$, we make use of the formulation of Universal Value Function Approximators (UVFA) [12] to train a policy to maximize the MI (1) between the inputs and z. That is, we exploit z as navigation goals or intrinsic objectives to learn the goal-conditioned skills: $\pi(a|s, z)$. Hence, we feed the concatenation of the encoded observation and z to the RL agents. Thus, at each step, the policy predictions depend not only on the current agent state but also on z. EDL [1] maximizes the forward form of the MI (1). That is feasible in EDL because the technique is applied to toy mazes where the states of the MDP are defined by 2D coordinates. In this way, EDL models $p(s|z)$ by variational inference and maximize the MI by deriving a reward that involves computing euclidean distances in the state space of coordinates. However, as in PixelEDL we deal with the image space, it is not coherent to match the euclidean distance in the image observation space with the distances in the 3D environment. For this reason we make use of the reverse form of the MI (1) and we model $p(z|s)$ with the encoder that learns latent representations from image observations. Finally, we craft a reward distribution that maximizes the MI (1) between the inputs and the skills by taking into account the distances in the latent space of the representations. Concretely, we assign a positive reward to an action a that positions the agents in a state s only if the encoded image observation is closest to the skill-conditioning z among all $z \sim p(z)$. We provide a report of the reward distributions in our project site.

For implementing the aforementioned training pipeline of PixelEDL, we use the baseline RL models provided by both Habitat and MineRL environments. These are Proximal Policy Optimization (PPO) [13] in Habitat, and Rainbow [4] in MineRL.

References

[1] Víctor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giro-i Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020. 1, 2

[2] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016. 1

[3] William H. Guss, Mario Ynocente Castro*, Sam Devlin*, Brandon Houghton*, Noboru Sean Kuno*, Crissman Loomis*, Stephanie Milani*, Sharada Mohanty*, Keisuke Nakata*, Ruslan Salakhutdinov*, John Schulman*, Shinya Shiroshita*, Nicholay Topin*, Avinash Ummadisingu*, and Oriol Vinyals*. Neurips 2020 competition: The MineRL competition on sample efficient reinforcement learning using human priors. *NeurIPS Competition Track*, 2020. 1

[4] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 2

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[6] Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2010. 2

[7] Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019. 1

[8] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967. 2

[9] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017. 2

[10] Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer, 2014. 1

[11] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019. 1

[12] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015. 2

[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[14] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020. 2

[15] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020. 2

[16] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018. 2

[17] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021. 2