PiCoEDL: Discovery and Learning of Minecraft Navigation Goals from Pixels and Coordinates

Juan José Nieto[1] Roger Creus Castanyer[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*
juanjo.3ns@gmail.com creus99@protonmail.com xavier.giro@upc.edu

1. Introduction

Defining a reward function in Reinforcement Learning (RL) is not always possible or very costly. For this reason, there is a great interest in training agents in a task-agnostic manner making use of intrinsic motivations and unsupervised techniques [7, 6, 15, 2, 14, 3]. Due to the complexity to learn useful behaviours in pixel-based domains, the results obtained in RL are still far from the remarkable results obtained in domains such as computer vision [5, 4] or natural language processing [1, 12]. We hypothesize that RL agents will also benefit from unsupervised pre-trainings with no extrinsic rewards, analogously to how humans mostly learn, especially in the early stages of life.

Our main contribution is the deployment of the *Explore, Discover and Learn* (EDL) [3] paradigm for unsupervised learning to the pixel and coordinate space (PiCoEDL). In particular, our work focuses on the MineRL [9] environment, where the observation of the agent is represented by: (a) its spatial coordinates in the Minecraft virtual world, and (b) an image from an egocentric viewpoint. Following the idea of *empowerment* [10], our goal is to learn latent-conditioned policies by maximizing the mutual information between states and some latent variables, instead of sequences of actions [7]. This allows the agent to influence the environment while discovering available skills.

2. From pixels and coordinates to skills

We formulate a Markov decision process (MDP) as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$. $\mathcal{S}$ is the high-dimensional state space (pixel images and coordinates), $\mathcal{A}$ refers to the set of actions available in the environment and $\mathcal{P}$ defines the transition probability $p(s_{t+1}|s_t, a)$. We learn latent-conditioned policies $\pi(a|s, z)$, where the latent $z \in \mathcal{Z}$ is a random variable.

Given the property of symmetry, the mutual information ($\mathcal{I}$) can be written using the Shannon Entropy ($\mathcal{H}$) in two ways:

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

Maximizing the mutual information (MI) requires knowledge of unknown distributions ($p(s), p(s|z), p(z|s)$). For the former we study two cases: (a) Using expert trajectories, and (b) Using the distribution induced by a random policy. A comparison of both strategies can be found in Section 2.1, for now we assume the latter case of $p(s)$. We rely on variational inference techniques for estimating the mappings from the states to the latent variables and backwards. These are estimated from the rollouts induced by the random policy. Finally, we also need to define a prior $p(z)$ to sample from, which in our case, following EDL [3] is a fixed uniform categorical distribution.

If we proceed with the derivation of the forward form in EDL [3], we can find that in our case the intrinsic objective becomes a distance in the pixel space. Since we cannot assume that it is representative of a meaningful distance in the environment, we discard this approach. Instead, we adopt the reverse form of the MI. We use the VQVAE [13] model, that allows us to estimate the posterior $p(z|s)$ with the encoder $q_\phi(z|s)$ by maximum likelihood on (s, z) tuples and also contains a categorical bottleneck for $p(z)$. Then, our final objective becomes:

$$r(s, z) = q_\phi(z = k|s) = \begin{cases} 1, & \text{if } k = \text{argmin}_j ||z_e(s) - e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$z_e(s)$ is the sum of the outputs of two encoders: (a) a 2D convolutional encoder for the images, and (b) a multilayer perceptron for the coordinates. Both have the same output dimension that allows summing up the resulting embeddings. Then in Equation 2, we find the index of the closest embedding in the VQVAE codebook e. Only if this index matches the sampled latent variable z that is conditioning the policy, it will return a reward of 1. Despite the sparsity of rewards, since we use a $p(s)$ that is induced from a random policy, we

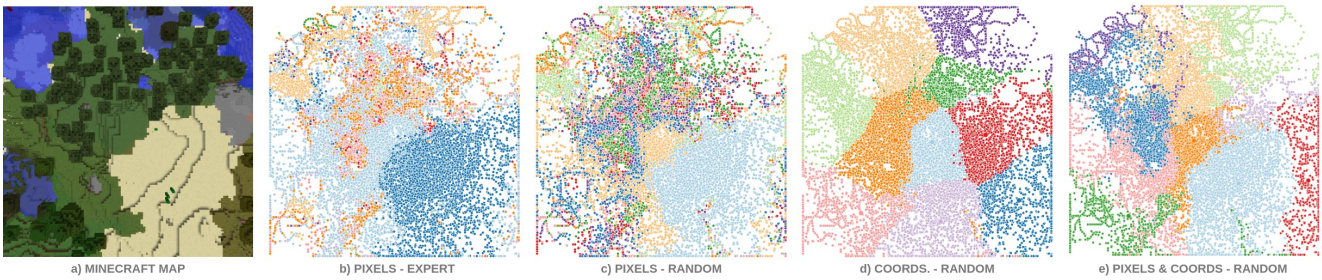

Figure 1. a) Top-view of our Minecraft map. The caption below b) c) d) e) refers to the nature of the states (pixels, coordinates or both) and the type of trajectories (expert or random). Each coloured point indicates the closest centroid to the encoded embedding.

know that these states are reachable and we will not suffer from exploration problems. Also, there are other problems due to multiple states rising positive rewards which could lead to ambiguous objectives. This can be tackled using a smoother reward function such as computing the distance in the embedding state, but in our experiments we did not have any problem by leveraging the previous reward.

In the following subsections we specify the implementation details of our approach.

2.1. Exploration and Skill Discovery

Firstly, we considered using expert trajectories to induce the distribution over the states. We used the MineRL dataset [8], which contains expert trajectories from different Minecraft worlds. Using expert trajectories may seem preferable since they contain human priors that give more weight to those states that are meaningful for discovering useful skills. However, Figure 1b shows that expert trajectories from pixels discover fewer and sparser skills than the those discovered by random exploration in our map, depicted in Figure 1c. This suggests that while the skills discovered by experts may be more generic as they were collected in different worlds, they are not as useful for our particular map as the skills discovered by a random policy. This hypothesis is supported by Figure 2, where we show the reconstructed images for each of the VQ-VAE codebook centroids. The reconstructions belonging to the expert trajectories contain scenarios that cannot be found in our Minecraft map.

In our study case, we aim to learn policies that treat the latent variables as navigation-goals. For this purpose, Figure 1 shows complementary skills discovered from pixels (Figure 1c) or coordinates (Figure 1d). We would like to discover skills that take into account not only the visual similarity but also the position relative to the initial state, so we adopt a solution that considers the two types of state representations (Figure 1e). This way our agent can distinguish between two visually identical mountains located at two different positions in the map.

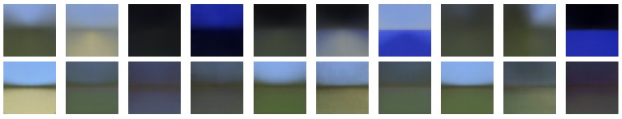

Figure 2. **Top:** images reconstructed from learned centroids using expert trajectories. **Bottom:** images reconstructed from learned centroids using pixels and coordinates from random trajectories.

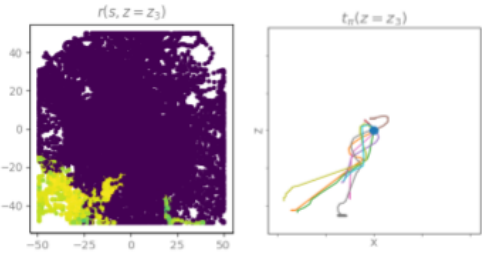

Figure 3. **Left:** Observations in yellow are encoded to the codebook embeddding z_3. **Right:** Trajectories followed by the agent conditioned by z_3.

2.2. Skill-Learning

In the last stage of PiCoEDL, we leverage Equation 2, derived from maximizing the mutual information between states and latent variables to maximize the expected cumulative reward. The latent codes discovered are now treated as goal states in a navigation task. We utilize Rainbow [11] algorithm to train our RL embodied agent. The input to the network is composed of the concatenation of the embedded observation with the latent embedding that is conditioning the policy. For each episode, we sample uniformly from $p(z)$ to determine the conditioning latent.

While there are some policies that are correctly learned, we find some others that do not achieve satisfactory results. We hypothesize that these are the latent codes that encode smaller regions of the state space, and with further tuning may achieve the desirable results. Figure 3 depicts the trained policy conditioned with the third codebook. More examples are available in our project site[1].

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

Acknowledgments

This work was partially supported by the Postgraduate on Artificial Intelligence with Deep Learning of UPC School, and the Spanish Ministry of Economy and Competitivity under contract TEC2016-75976-R. We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPUs used in this work.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1

[2] Y. Burda, H. Edwards, A. Storkey, and O. Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018. 1

[3] V. Campos, A. Trott, C. Xiong, R. Socher, X. Giro-i Nieto, and J. Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020. 1

[4] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers, 2021. 1

[5] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021. 1

[6] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018. 1

[7] K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016. 1

[8] W. H. Guss, B. Houghton, N. Topin, P. Wang, C. Codel, M. Veloso, and R. Salakhutdinov. Minerl: A large-scale dataset of minecraft demonstrations. In *IJCAI*, 2019. 2

[9] W. H. Guss, M. Y. Castro, S. Devlin, B. Houghton, N. S. Kuno, C. Loomis, S. Milani, S. Mohanty, K. Nakata, R. Salakhutdinov, et al. The minerl 2020 competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:2101.11071*, 2021. 1

[10] D. Hafner, P. A. Ortega, J. Ba, T. Parr, K. Friston, and N. Heess. Action and perception as divergence minimization. *arXiv preprint arXiv:2009.01791*, 2020. 1

[11] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, 2018. 2

[12] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019. 1

[13] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017. 1

[14] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017. 1

[15] D. Warde-Farley, T. Van de Wiele, T. Kulkarni, C. Ionescu, S. Hansen, and V. Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018. 1