# Curiosity-driven Exploration by Self-supervised Prediction

Kartik Patath    Sapan Santosh Agrawal    Vamshi Krishna U    Johnathan A'Vant

***Abstract*— In this project we plan to implement the idea of curiosity based agent which learns explores the environment using self-supervised prediction. The environment that we plan to use for this project is the Minecraft environment and we want to solve the Navigation task of MineRL Challenge, NeurIPS 2019 [1] . Given that we are able to implement the above problem statement, we plan to expand on this curiosity driven approach by improving the current curiosity based algorithm which will act as a baseline approach.**

***Index Terms*—Curiosity based learning, sparse reward, self-supervision.**

## I. Introduction

Rewards from the environment depict how well the agent is performing in the required task [2]. Majority of the techniques in reinforcement learning depend upon the reward from the environment to select the actions which maximize the overall reward. However, in real-world scenario such explicit rewards might be sparse or even not available to the agent.

The idea of Curiosity-Driven learning [3], is to build a reward function that is intrinsic to the agent (generated by the agent itself). It means that the agent will be a self-learner since he will be the student but also the feedback master. Curiosity helps an agent explore its environment in the quest for new knowledge (a desirable characteristic of exploratory behavior is that it should improve as the agent gains more knowledge). Further, curiosity is a mechanism for an agent to learn skills that might be helpful in future scenarios. In this work, we plan to use this approach to tackle the sparse reward problem for navigation in Minecraft environment [1]. However, though the paper claimed to be robust to television screen noise problem, the agent got stuck in a similar problem of predicting dust/dirt motion when agent attacks trees and land spots in MineRL. Thus, to tackle this problem we explored another curiosity based method - RND. In which the next state prediction model employed by ICM is replaced with a Random Distillation Predictor network which induces noise in the prediction model and makes it robust to such television screen noise problem. Here we observe that RND when employed for the mineRL environment outperforms ICM results for both sparse and dense navigate environments.

All authors are from Worcester Polytechnic Institute, (kpatath, ssagrawal, vuppununthala, jaavant)@wpi.edu

The report is organised as follows: Section 2 discusses the related work. ICM is discussed in Section 3, followed by discussion on RND in Section 4. Experimental setup is stated in Section 5. And Section 6 provides the results and analysis, comparing and laying down the benefits and drawbacks of each method. Finally the work is summarised in the Section 7, drawing the conclusion. Future works are stated in Section 8.

## II. Related Work

Significant prior work has been done to tackle to sparse reward problem either by encouraging the agent to explore "novel" states [4], [5] or to perform actions that reduce the error in the agent's ability to predict the consequence of its own actions [6], [7]. This work however, is greatly influenced by [3]. We validate the work on MineRL environment and state the strengths and weakness of the ICM based method. Additionally, we compare the results with that of [8] and discuss how the RND method can significantly improve exploration for the chosen environment.

## III. Intrinsic Curiosity Module (ICM)

The agent is composed of two subsystems: a reward generator that outputs a curiosity-driven intrinsic reward signal and a policy that outputs a sequence of actions to maximize that reward signal. In addition to intrinsic rewards, the agent optionally may also receive some extrinsic reward from the environment. Let the intrinsic curiosity reward generated by the agent at time t be $r_t^i$ and the extrinsic reward be $r_t^e$ . The policy sub-system is trained to maximize the sum of these two rewards $r_t = r_t^i + r_t^e$, with rte mostly (if not always) zero.

We make the agent curious by making it visit the states in the environment which it has rarely explored. Thus, rewarding the agent to explore/find new states. Also,

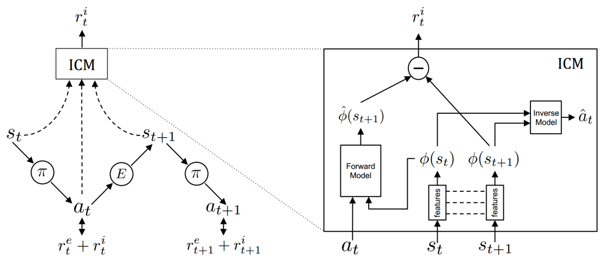

Fig. 1: Flow diagram showing Intrinsic Curiosity Module(ICM) with extrinsic and intrinsic rewards

as we can enable the agent to predict only the states which it has visited in the past by learning the forward dynamics of the environment, the intrinsic reward can be generated whenever the agent fails to predict the next state. Making predictions in the raw sensory space is undesirable because both, it is difficult and it can distract the agent making it more curious about the random/redundant motions in the environment (like motion of clouds, leaves, flowers). This problem is solved by learning only those features in the environment which actually affects the agent's action. This feature space is trained by maximizing the networks ability to predict the action $(a_t)$ which leads to next state $(s_{t+1})$ given the current state$(s_t)$ as shown in Fig. (1).

$$\hat{a_t} = g(s_t, s_{t+1}, \theta_I) \tag{1}$$

$$\min_{\theta_I} L_I(\hat{a_t}, a_t) \tag{2}$$

$L_I$ is the loss function that measured the discrepancy between the predicted and the actual actions and $g$ is the inverse dynamics model. The forward model in then trained to predict the feature encoding of the state at next time stamp $t+1$,

$$\hat{\phi}(s_{t+1}) = f(\phi(s_t), a_t; \theta_F) \tag{3}$$

$$\min_{\theta_F} L_F(\hat{\phi}(s_{t+1}), \phi(s_t)) \tag{4}$$

where, $L_F$ is the loss function and $f$ is the forward dynamics model. Thus, the intrinsic reward can be formulated as,

$$r_t^i = \frac{\eta}{2} \parallel \hat{\phi}(s_{t+1}) - \phi(s_{t+1}) \parallel_2^2 \tag{5}$$

where $\eta$ is the scaling factor. To generate the curiosity based intrinsic reward signal, the forward and inverse dynamics loss are optimized combined.

## IV. Experimental Setup

### A. Environment

This work aims to compare the performances of ICM based curiosity module and RND on a MineRL environment. MineRL is the research project started at Carnegie Melon University to develop various Reinforcement Learning algorithms to better understand the stochastic environment. [9] shows the baselines for hosting a competition where there are multiple environment of the game MineRL. MineRL-v0 Dataset (as mentioned in [9] – One of the largest imitation learning datasets with over 60 million frames of recorded human player data. The dataset includes a set of environments which highlight many of the hardest problems in modern-day Reinforcement Learning: sparse rewards and hierarchical policies. For this project, we have showed results by using various navigation environments like MineRLNavigateDense-v0, MineRLNavigate-v0 and MineRLNavigateExtremeDense-v0 [10]. The envrionments mentioned below are used with wrappers to stack the environment images, combile multiple actions to one, discretizing the continuous observation spaces etc.

*1) MineRLNavigate-v0::* In this environment, the goal of the agent is to move to a location depicted by a diamond block. Apart from the standard observation space, the observation space in this environment also has compass angle, which points towards the goal location, 64meters from the start. There is a small horizontal offset which is random from compass location and it might happen that it is slightly below surface level. A sparse reward of +100 would be given upon reaching the goal. At this point, the episode terminates. [10] In this task, the agent must move to a goal location denoted by a diamond block. In addition to standard observations, the agent has access to a "compass" observation, which points near the goal location, 64 meters from the start location. The goal has a small random horizontal offset from the compass location and may be slightly below surface level. The agent is given a sparse reward (+100 upon reaching the goal, at which point the episode terminates). [1]

*2) MineRLNavigateDense-v0::* MineRLNavigateDense-v0 environment is a variant of the MineRLNavigate-v0 with dense reward shape. The agent is given a positive/negative reward based on how close/farther the agent moves towards/away from the goal. The agent is spawn on random survival map [10]

*3) MineRLNavigateExtremeDense-v0::* This environment is almost similar to MineRLNavigateDense-v0 except that the agent is spawn in extreme hills biome. [10]

## B. ICM

The forward model as shown in Fig. 1 has two fully connected layers connected by ReLu [11]. Each of these layers have 519 units. The inverse model first convert the current state $s(t)$ and the next state $s(t+1)$ using a series of Convolutional Neural Networks(CNN) [12] with wrapped environment images as input. The CNNS have kernel sizes of 8, 4, 3 respectively and stride lengths of 4, 2 and 1. Leaky ReLU [13] non-linearity used after each CNN layer. The output of CNN layers is flattened and then connected to a fully connected layer. The dimensionality of feature vector $\phi(s_t)$ is 3136. For the inverse model, $\phi(s_t)$ and $\phi(s_{t+1})$ are conacatenated into a single vector and passed as inputs to a fully connected layer of 519 units followed by an output of fully connected layer with 7 nodes to predict one of the 7 possible output actions. The forward model is constructed by concatenating $\phi(s_t)$ with $a_t$ and passing it into a sequence of two fullyconnected layers with 519 and 512 units respectively.

## C. Random Network Distillation

Random Network Distialleation (RND) is another method based on prediction that helps our agents to explore our environment through curiosity. It taxes visiting unfamiliar states by measuring the toughness of predicting the next state. In ICM, we train the forward model to predict $\phi(s_{t+1})$ using $s_t$ and $a_t$ which leads the agent getting stuck at scenarios which are called as ”Noisy TVs” (discussed elaborately in Results section part C) where the forward prediction model fails to accurately predict the next state and hence generates a huge curiosity reward (intrinsic reward) which prompts the agent to spend more time in front of such Noisy TVs. Hence we can say that the prediction error is high where the predictor fails to generalize from previously seen examples and because the target is stochastic.This is prevented in RND by replacing the forward model with predicting the output of a fixed and randomly initialized neural network (hence the name RND) onto the next state, given that we provide next state itself as an input. The intuition here is the fact that predictive models have low errors in states similar to the ones they have been trained on. In particular our agent’s predictions of the output of a randomly initialized neural network will be less accurate in unseen (new) states than in states the it has already visited frequently. Hence we can say that we are able to avoid getting stuck at Noisy TVs. The

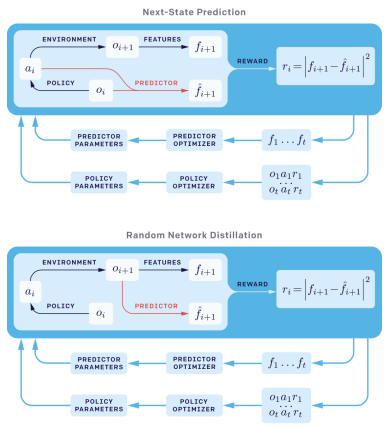

Fig. 2: Image showing the difference between next state predictions of ICM(top) and RND(bottom) algorithms []

Fig.2 we can see the difference in next state predictions of ICM and RND.

## D. Training

The training was performed on a machine with the configuration: i5 processor, 15GB RAM, Nvidia Geforce 980 gtx with 2048 gpu cores are used for training. For each of the models, the training was done for approximately half a day. In the section V the results of training and intrinsic rewards are presented.

## V. Results

## A. Extrinsic Reward

In our experiments, we try to compare the performance of the ICM and RND model. Through the use of the multiple environments provided by MineRL, we can see how well our models perform based on an agent providing itself with an intrinsic reward and additionally being provided with a extrinsic reward from the environment.

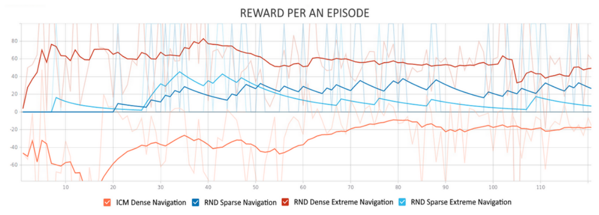

Fig. 3: Reward Per Episode

In the situation of the extreme dense reward navigation environment, the RND model obtains its best performance. This is merely expected as such a model provides an extrinsic reward every few frame steps. In the combination of a dense extrinsic reward and an intrinsic reward, the agent obtains the best performance compared to any other environment. In the two other sparse reward environment tested on for the RND models, we still see a well suited performance given a nearly infinite exploration state and sparse reward. The extreme environment takes longer to converge, as expected, give that the environment provides the agent with a more difficult terrain to navigate compared to the normal navigation environment. Such results show that the RND model can provide a proper intrinsic reward that motivates the agent to curiously explore it's environment an obtain an improved reward. Whereas the ICM model is ill suited for such tasks in the MineRL provided environments.

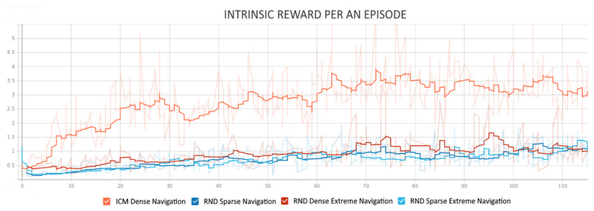

Fig. 4: Intrinsic Reward Per Episode

## B. Intrinsic Reward

Results in figure 4 indicate the intrinsic reward generated by the agent through the use of ICM or RND. A spike in the chart of figure 4 can be noted as an state that the agent poorly predicted. This could be seen as a good thing, as the agent will receive a higher incentive to curiously explore the unexpected state. The RND intrinsic rewards relatively sit around each other, due to a normalization that happens to the reward before it is returned to the agent. However in the ICM, even though it is a few magnitudes larger than the RND rewards, continually learns and has many drastic spikes. As noted, these spikes relate to the agent exploring an unexpected state which is good, but in the corresponding chart for the extrinsic reward in figure 3, we see that the agent never learns or obtains a positive reward. This oddly shaped relationship was investigated and was identified to be the result of the ”Noisy TV” problem.

## C. Noisy TV

The noisy tv problem is well document downfall of the ICM agent [8]. In the event where the environment provides and stochastic state that is independent of the agent's actions, for example, a tv that changes stations every few seconds on its own, the agent begins to procrastinate by becoming more curious of such state. With that being said, if the agent looks at the tv it is unable to predict, the ICM generates a high intrinsic reward for the agent. After investigating the results of the ICM it was discovered the many states in the MineRL MineRL environments mimicked the noisy tv problem. Some of these situations were, the agent being drawn to run to the edge of the world to watch the game engine randomly render the screen, punching stones to watch particles fly out of the stone, watch the clouds move across the sun, and many more. Such procrastination of the environment had a drastic effect on the ICM model's performance. Given that the RND model was designed to address this problem, allowed the RND model to proactively explore the environment without getting stuck at noisy TV states.

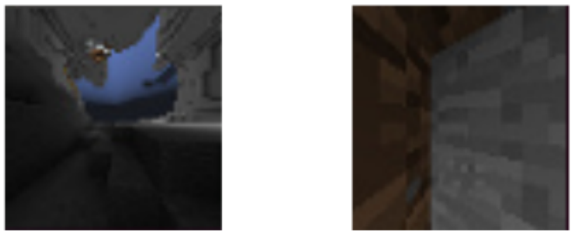

Fig. 5: Noisy TV Examples in MineRL

## VI. Discussion

In this work, we show that the use of curiosity learning for exploration by the use of the RND model enables an agent to productively play MineRL's navigation environments. We additionally demonstrate the RND model significantly outperforms the ICM model in any MineRL environment. Inversely we show that the ICM model's feature representation and forward model is incapable of producing a productive intrinsic reward; that enables the agent to curiously explore its environment without procrastinating at certain states. After an investigation into the failure of an agent to obtain an extrinsic reward in the MineRL environment using the ICM Model, we obtain evidence that many states of the MineRL environment mimic the noisy tv problem.

Given that the original use of curiosity learning focuses on Atari games [3], our work provides insight into any future work that focuses on 3D environments with more rich graphics and post-processing effects. Our work shows that potentially any future work revolving around the use of modern games and implementation of curiosity learning will need to address, naturally occurring noisy TV states that exist throughout the environment.

## REFERENCES

[1] "Neurips 2019 : Minerl competition," accessed: 2019-10-31.

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013.

[3] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," 2017.

[4] 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*, 2016, pp. 1471–1479.

[5] M. Lopes, T. Lang, M. Toussaint, and P.-Y. Oudeyer, "Exploration in model-based reinforcement learning by empirically estimating learning progress," in *Advances in neural information processing systems*, 2012, pp. 206–214.

[6] S. Mohamed and D. J. Rezende, "Variational information maximisation for intrinsically motivated reinforcement learning," in *Advances in neural information processing systems*, 2015, pp. 2125–2133.

[7] J. Schmidhuber, "A possibility for implementing curiosity and boredom in model-building neural controllers," in *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, 1991, pp. 222–227.

[8] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros, "Large-scale study of curiosity-driven learning," 2018.

[9] W. H. Guss, C. Codel, K. Hofmann, B. Houghton, N. Kuno, S. Milani, S. P. Mohanty, D. P. Liebana, R. Salakhutdinov, N. Topin, M. Veloso, and P. Wang, "The minerl competition on sample efficient reinforcement learning using human priors," *CoRR*, vol. abs/1904.10079, 2019. [Online]. Available: http://arxiv.org/abs/1904.10079

[10] "Minerl competition - general information," 2019. [Online]. Available: http://minerl.io/docs/environments/index.html

[11] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[12] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[13] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.