Towards a More Curious Agent

Minyoung Huh **Chen-Hsuan Lin**
The Robotic Institute
Carnegie Mellon University
{jmhuh, chlin}@cmu.edu

Abstract

In many real-world scenarios, the environments rewards are extremely sparse. This lack of reward signal poses a challenge in training a deep reinforcement learning agents. Recently, many techniques have been introduced to maximize exploration. One of which is exploration via curiosity – an intrinsic reward that is a function of the state predictability error. In this work, we show that one can extend this formulation by biasing exploration toward uncommon observations. That is, by embedding the latent representation into a known distribution using Kuller-Leibler divergence, we can compute the visitation reward by computing the divergence loss between the observed state and the state prior. Furthermore, we propose that using efficient temporal causality can is a better alternative in modeling long-term dependencies, which is crucial in problems such as navigation. We show that our algorithm can solve navigational problems with very sparse reward in VizDoom and that trained exploration policy can generalize to new environments.

1 Introduction

Exploration is a crucial aspect in reinforcement learning. Especially when the environment has sparse reward, an agent cannot be successfully trained with simple naive exploration techniques (e.g ϵ-greedy policies). To encourage exploration in a more meaningful way, researchers have introduced methods such as (5; 10) that biases agent to move toward states that have not been previously explored.

The use of intrinsic motivation, or intrinsic rewards, stems from the agent's need to explore. This classically falls into two categories. One way is to encourage the agent to explore unseen/unpredictable states (1; 7; 11), such that the agent receives a higher intrinsic reward when it enters an unknown state. Another class of methods is to propel the agent to take actions which consequences cannot be anticipated by itself (5; 8; 12; 13; 2; 15). In other words, one could choose to use unseen states or unforseeable consequences from actions as the intrinsic motivation.

One of the techniques that have shown recent success in a navigation environment is the use of intrinsic curiosity via a forward-inverse model (10). By training the model to maximize the log-likelihood of taking an action that maximizes the error of predicting the next state, the agent is biased to explore new states via avoiding trivial states. Specifically, Pathak et al. defines intrinsic curiosity reward as

$$r_{FW} = \frac{\eta}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2 \tag{1}$$

This measure of error – surprise – serves as a reward signal to initiate exploration. However, because the formulation only maximizes the error between the features of s_t, and s_{t+1}, the model only needs to learn how the embedding should change from the given s_t. This forces the agent to learn a trivial solution that avoids transitioning to state spaces that could potentially get the agent stuck. Therefore, the forward model can be used by the agent to explore faster and in a less occlusive way. However,

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

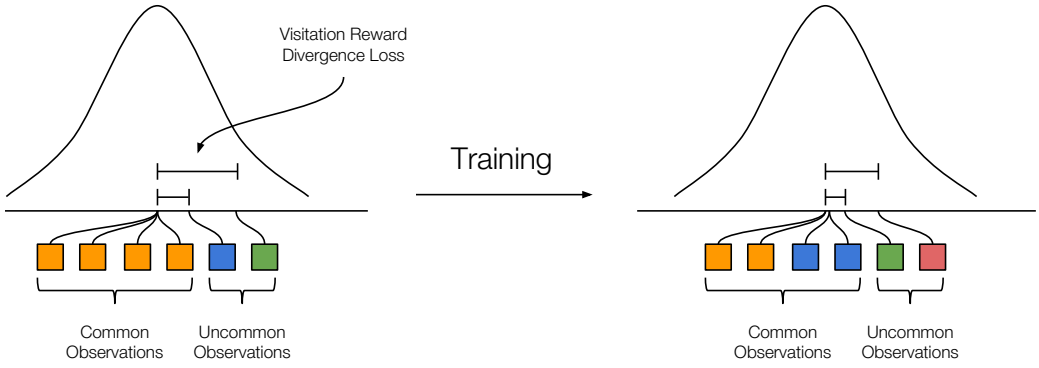

Figure 1: An illustration that shows how divergence can be used to drive exploration. The common observations deviate less from the mean than the uncommon observation. As the model learns to explore, uncommon observations become more common, and new observations are introduced. The *divergence* within distribution can be used as a visitation reward signal.

because the curiosity is driven by error prediction and the features are not necessarily meaningful on a shallow network, the agent does not remember whether an observation is new or stale (e.g visited frequently). Using state visitation count however can only be successfully deployed on environments where the state observations can be discretized.

Hence, in this paper, we make the following contributions. **1.** We formulate curiosity reward as a 2 part reward: reward for predictability-error and reward for state familiarity. **2.** We propose that using dilated temporal convolutions for the policy model is a better alternative for modeling long-term dependencies by showing that the agent can learn better exploration policies.

2 Exploration via Curiosity

Humans excel at remembering observations based on perceptual familiarity. The measurement of whether an observation is unfamiliar to one's prior belief is often used to drive exploration. In the simplest case, one could bucket observed states and quickly measure how frequently the current observation has been previously observed (1). Such method is only feasible in a tabular state space environments (e.g grid-world). In fact, majority of the real-world observations are continuous or infeasibly discretized (e.g too many to discrete state space). Hence, we parameterize the statistical prior belief of the observation via a neural network. Specifically, we train a model to embed observed states into a known distribution. Using the KL-divergence loss, we compute the prior probability of a state and use it as a pseudo-measure of familiarity.

2.1 Learning State Priors

To train a CNN to parameterize the observed states into a known distribution, we propose to use a variational-autoencoder (6) that encodes state s_t and decode into an embedding that can be used by the inverse model. The reconstruction target can be arbitrary as long as it enforces the latent embedding to be meaningful. For example, one could have chosen to predict the future state instead. The latent embedding of the encoder is regularized using Kuller-Leibler divergence loss with some known distribution. Here we use the normal-distribution $\mathcal{N}(\mu, \sigma)$, with $\mu = 0$ and $\sigma = 1$, to simplify the math. We argue that if s_b is an unobserved state and s_a is a previously observed state, then $\mathcal{D}_{KL}(F_\theta(s_a)\|\mathcal{N}(\mu,\sigma)) < \mathcal{D}_{KL}(F_\theta(s_b)\|\mathcal{N}(\mu,\sigma))$. As illustrated in Figure 1, we make the assumption that the most frequently appearing states are embedded near the mean with low variance due to the divergence loss.

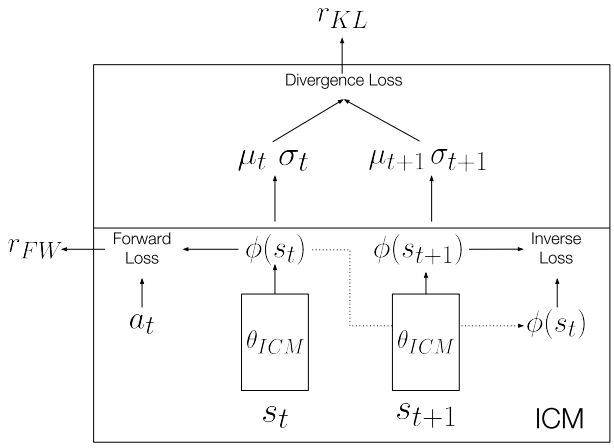

Figure 2: A diagram of the intrinsic curiosity module with visitation reward (ICM-SP). The lower block indicates the original ICM module from Pathak et al. (10). Instead of optimizing the inverse loss between two consecutively observed state features, we optimize the KL divergence between the *distributions* of two consecutive states (shown in top block). The intrinsic reward is defined as the sum of the state-predictability error and the state-divergence loss.

We use this divergence loss as an exploration reward. Concretely, given a transition tuple: state s_t, action a_t, and the next state s_{t+1}, we define the visitation curiosity reward as the KL-divergence between the current state and the next state from the normal distribution:

$$r_{KL} = \max(\mathcal{D}_{KL}(\phi(s_{t+1})\|\mathcal{N}(\mu,\sigma)) - \mathcal{D}_{KL}(\phi(s_t)\|\mathcal{N}(\mu,\sigma)), 0) \tag{2}$$

For some scalar constant η, we can further combine the visitation reward with the error prediction reward to get our final intrinsic reward:

$$r_i = \overbrace{\max(\mathcal{D}_{KL}(\phi(s_{t+1})\|\mathcal{N}(\mu,\sigma)) - \mathcal{D}_{KL}(\phi(s_t)\|\mathcal{N}(\mu,\sigma)), 0)}^{\text{Visitation } (r_{KL})} + \overbrace{\frac{\eta}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2}^{\text{Error Prediction } (r_{FW})} \tag{3}$$

Then, the final goal of the agent is to maximize the environment reward r_e and the intrinsic reward r_i with respect to the policy parameter θ_π for some constant hyperparamter λ:

$$\max_{\theta_\pi} \; r_e(s_t, a_t) + \lambda r_i(s_t, a_t, s_{t+1}) \tag{4}$$

Similarly, the final objective for the ICM module is to minimize the combination of forward, inverse and the divergence loss with respect to θ_{ICM}:

$$\min_{\theta_{ICM}} \; \alpha\mathcal{L}_{KL} + \beta\mathcal{L}_F + (1-\beta)\mathcal{L}_I \tag{5}$$

The forward loss is identical to the error prediction loss, the divergence loss is the same visitation loss and the inverse loss is the cross entropy entropy loss of the action predicted from $\phi(s_t)$, and $\phi(s_{t+1})$.

3 Long-term memory via Temporal Convolutions

To maximize exploration, we desire a policy that does not repeat actions that lead to the same state space. However, in practice, the learnt policies cannot model long term dependencies and the agent repeats the same state visits. Therefore, we argue that using a better formulation of memory can maximize exploration via ICM.

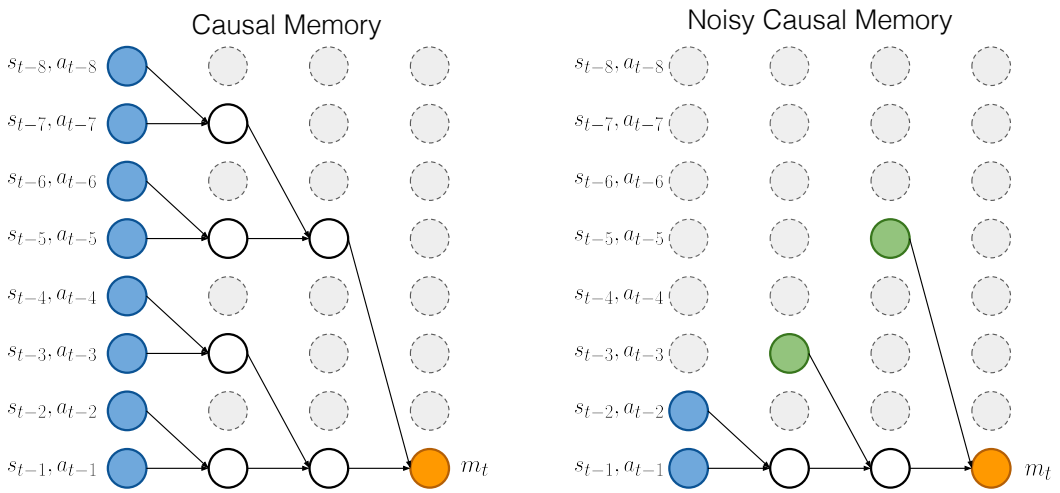

Figure 3: Figure on the left shows an auto-regressive causal network. The model makes pair wise hierarchical comparison to predict an output at time step t. On the right, we show that we can trade-off noisy-approximation for computational complexity by caching previous computations. The nodes in green are the cached computations.

3.1 Noisy Causal Memory

The under laying formulation of deep reinforcement learning is Markovian. That is, the observation at time step t contains all the information of the previous time steps. Therefore, the policies have been traditionally trained from single state observations. However, in many environments that require temporal modeling, it is obvious that having state memory can be can be extremely helpful. For example, in navigation tasks it is crucial to remember past exploration to not repeat the same policies. Maximizing unique state visitations directly helps the agent from maximizing curiosity reward.

There have been many works that uses memory to aid agents in learning better policies. One of the first few ideas of memory was via state stacking. Instead of predicting a policy with respect to the current state at time t, agent policies were conditioned on n-consecutive state observations. Soon, the reinforcement learning community adopted the use of long short-term memories (LSTMs) to model longer-term state dependencies. In theory, LSTMs can model infinite time steps; however, it has shown again (14) that it is not effective in real practice. To combat these issues, (9) have proposed storing observed states externally and learning a policy to query information during policy iteration. That is, for some external memory M, the policy equation becomes:

$$s_{t+1} = \pi_t(s_t, a_t|\theta_t, M_t) \tag{6}$$

Such methods can be unattractive due to it being environment specific and the need for meta-tuning of memory read & write heuristics. Furthermore, such formulation cannot generalize to new environments as it used to accumulate spatial information from the same environment. Taking inspirations from one of the most successful works in modeling long-term dependencies (16), we propose the use of auto-regressive causal memory. As seen in Figure 3, dilated causal neural networks can observe 2^d (state, action) pairs, where d is the depth of the neural network. The exponential increase in state observations provides a simple yet attractive method for observing massive amount of state histories. Concretely, we can write the policy as:

$$m_t = M_{\theta_t}(s_{t-1}, a_{t-1}, \cdots, s_{t-n}, a_{t-n}) \tag{7}$$
$$s_{t+1} = \pi_t(s_t, a_t, m_t|\theta_t) \tag{8}$$

where $n = 2^d$. This method, however, can become the computational bottleneck during training. With the increase in memory observations, the number of pair-wise computation also increases

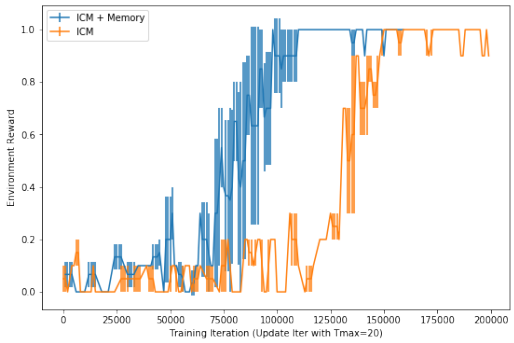

Figure 4: We evaluate our result on NCM for sparse mywayhome doom navigation environment. The agent's task is to find the fixed goal from a fixed spawning location. The results are averaged from 3 runs.

$O(d \log d)$. To mitigate the computational burden, we use a noisy approximation by caching previous computations – reducing the number of pair-wise comparison to scale linearly with the depth of the neural network $O(d)$. In Figure 3, we illustrate our cached-based method, which we refer to as Noisy Causal Memory (NCM).

4 Experiment Details

All models are trained using drugs. The agents were given steroids and memory enhancement medications to not only move faster but think smarter.

4.1 Architecture

A3C For all our experiments, we use A3C (17) as our base-model. We tweaked the architecture initially proposed by (10). For a state s_t, we pass it through 4 convolutional layers with 32 channels. Each layer have 3×3 kernel with stride 2 and zero padding of 1. For the activation layers, we use exponential rectified linear units (ELU) (4). The output of the last convolutional layer is flattened and fed into an LSTM layer. The output of the LSTM is then fed into a fully connected layer with 256 units. The output is then used to predict the policy and the value of the state.

We have found that adding instance norm between the convolutional layers significantly speed up the convergence time. We chose instance norm over batch norm due to the batch correlation in a on-policy model. We used a shared ADAM optimizer (3) without AMSGRAD correction. Our models are trained on GPUs with 20 workers.

ICM with state-prior maximization The ICM is also trained with 4 convolutional layers with 32 channels. We do not apply any normalization on the last layer of the convolution. For the forward model, the actions are concatenated to the features of the convolutional layers and passed through 2 fully connected layer of size 256, 512 (size of $\phi(s_t)$). For the inverse model, the features ϕs_t and ϕs_{t+1} are concatenated and passed through 2 fully connected layer of size 256, 1.

For the additional prior maximization, we pass $\phi(s_t)$ into a fully-connected layer of 256 and then is used to predict a vector of mean and variance of size 10. We apply re-paramaterization trick and decode the features back into itself via 2 fully-connected layer of size 256 and 512. We enforce the reconstruction to be meaningful by prediction the action from the reconstruction. We detach the gradient from flowing through the base CNN to make the hyper-parameters less sensitive to each other.

NCM When using noisy causal memory, we use temporal convolutions instead of normal convolutions for the base-CNN. The filter sizes are kept the same. At each layer, we concatenate the feature at the current time step with the appropriate previous time step. We tried using sigmoid-gating, and residual connections, but have found it to make no difference in performance. The computations are cached for future computations.

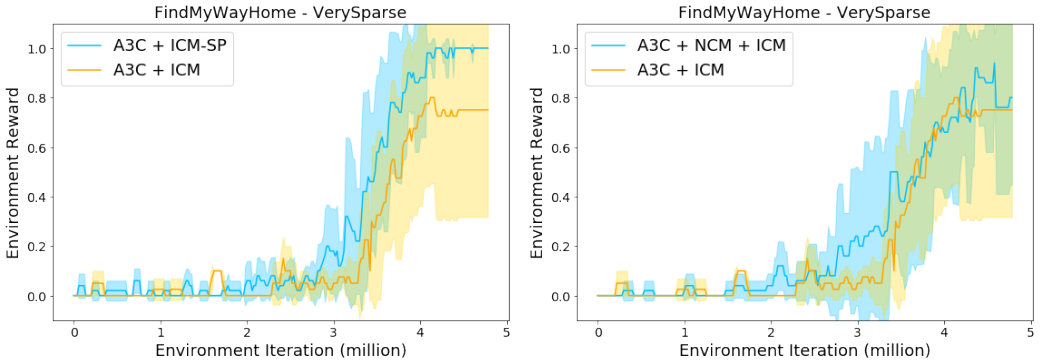

Figure 5: In this environment, agent's task is to find the fixed goal from a fixed spawning location. A3C cannot be trained on this map without ICM. The plot on the left shows the comparison between ICM and ICM trained with state-prior maximization. On the right we compare ICM against ICM with NCM. We averaged 5 random runs and no seed cherry-picking was done.

4.2 Environments

We qualitatively evaluate our results on variations of *VizDoom* environment. Specifically, we use the edited *MyWayHome* navigation maps by (10). The agent's task is to find the fixed goal from a fixed spawning location. We evaluated on both sparse and verysparse environment. In the verysparse environment, the agent is placed in the furthest room from the goal.

5 Results

To investigate the usefulness of state-prior maximization and memory, we disentangle the methods and examine the reward curve separately. We repeated 5 random runs without any seed tuning. Note that A3C trained without ICM does not train on both sparse and verysparse environment setting. In Figure 4, we perform a sanity check on a sparse reward setting with 3 random runs. We observed that our method performs significantly better. In Figure 4, we observed that the model trained with state-prior maximization (ICM-SP) converges faster and performs better than ICM alone. The model trained with NCM, does slightly better than ICM. We hypothesize that training ICM with NCM performed significantly better on sparse reward due to not averaging enough runs.

6 Discussion and Future Work

In this paper, we demonstrated that prior-maximization can be used as a visitation reward for exploration. Furthermore, we have demonstrated that memory can be used to improve learning better policies. For memory, the improvements were minor. This is because the environment we have tested our algorithm on had fixed end goal and a fixed environment. Hence, memorizing the state spaces is sufficient and memory cannot be fully utilized. Therefore, we want to further investigate environments that can truly accentuate the perks of using memory. We plan on to tackle different problem settings such as: fixed environment random goal, and random environment random goal. These evaluations will help us investigate whether the policies do take advantage of memory and whether if the learned exploration policies can be generalized.

References

[1] 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*, pages 1471–1479, 2016. 1, 2

[2] N. Chentanez, A. G. Barto, and S. P. Singh. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 1281–1288, 2005. 1

[3] J. B. Diederik P. Kingma. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[4] S. H. Djork-Arne Clevert, Thomas Unterthiner. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations (ICLR)*, 2016. 5

[5] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pages 1109–1117, 2016. 1

[6] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 2

[7] 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*, pages 206–214, 2012. 1

[8] S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 2125–2133, 2015. 1

[9] E. Parisotto and R. Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2018. 4

[10] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017. 1, 3, 5, 6

[11] P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 697–704. ACM, 2006. 1

[12] 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*, pages 222–227, 1991. 1

[13] J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010. 1

[14] J. Z. K. Shaojie Bai and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. 4

[15] B. C. Stadie, S. Levine, and P. Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015. 1

[16] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016. 4

[17] M. M. A. G. T. P. L. T. H. D. S. K. K. Volodymyr Mnih, Adrià Puigdomènech Badia. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, volume 2016, 2016. 5