Never Give up (Learning Directed Exploration Policies):

`https://arxiv.org/abs/2002.06038`

Paper Summary Notes

1 Main Ideas, High Level Assumptions

Context: hard exploration with sparse external reward feedback is difficult. Being greedy after finite number of steps is sub-optimal, and using boltzman exploration is not very efficient. We want a exploration signal that doesn't die over time; hence *never giving up*

- **idea**: reformulate the problem as an mdp with dense rewards by providing our own custom rewards to the agent outside of the ones provided by the environment
- **approach**:
  - construct an intrinstic exploration reward based on short term episodic novelty, as measured by a knn-lookup of memory in a learned vector space
  - construct life-long cross episode novelty as measured by prediction error against a random distilled network
  - combine intrinsic and external reward by directly feeding a weighting factor into the value approximation, using some fixed number of discrete weights in training, thereby jointly learning a set of policies, each with different exploration tendencies.

2 Approach and Formulation

2.1 Reward Architecture

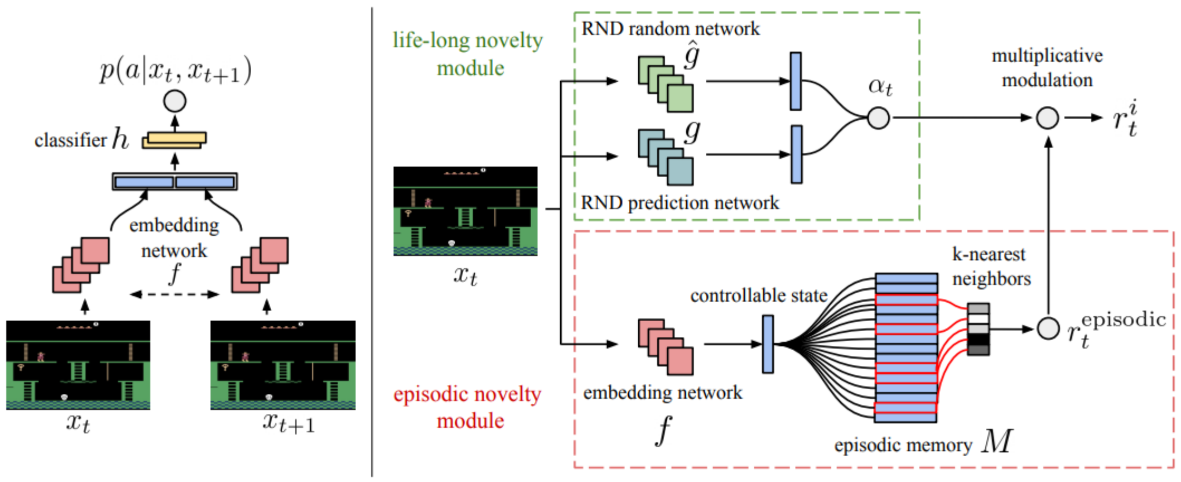

Figure 1: (left) Training architecture for the embedding network (right) NGU's reward generator.

1. notion of state ignores aspects of the env outside agent control
2. rapidly discourages revisiting same state within episode
3. slowly discourages revisiting same state across episode

- embedding network handles (1)
- intrinsic episodic novelty reward handles (2)
- intrinsic life-long novelty reward handles (3)

2.1.1 Embedding Network

- function $f : \mathcal{O} \mapsto \mathbb{R}^k$ maps the current observation to a learnt vector space, referred to as *controllable state space*, with the property that this space is invariant to dynamics not controllable directly by the agent
- we train f by paramterizing a siamese nework:

learn $p(a|x_t, x_{t+1})$ encoded as $h(f(x_t, f(x_{t+1}))$ under max-liklihood, and take the leanrnt embedding given by f, to do all the intrinsic reward engineering explained below

2.1.2 Intrinsic Episodic Reward

- encourages agent to periodically revisit familiar but not fully explored states
- using the self-supervised encoding given by the embedding network f, we keep an episodic controllable state space memory
- given this Memory M, we at each step we perform knn lookup over the memory, and reward for large distances in this vector space using the kernel:

$$r_t^{\text{episodic}} = \frac{1}{\sqrt{n(f(x_t))}} \approx \frac{1}{\sqrt{\sum_{f_i \in N_k} K(f(x_t), f_i)} + c}$$

- will be quickly changes due to explicit write op into memory

2.1.3 Intrinsic Life-Long Reward

- progressively down-modulates states that become more familiar across many episode of training
- let $g : \mathcal{O} \mapsto \mathbb{R}^k$ be a random network that produces some point as a function of our observation
- let $g' : \mathcal{O} \mapsto \mathbb{R}^k$ be a trainable network, that aims to predict the outputs of g
- define the modulation constant to be a normalized version of the mse between the 2 networks:

$$\text{err}(x_t) = ||\hat{g}(x_t; \theta) - g(x_t)||^2$$

- the idea is that seeing a new episode in training will generate a large error, and therefore be rewarded for the novelty
- this intrinsic reward training slowly because we use iterative gradient descent optimizer
- the modulation is integrated to combine with the episodic reward like:

$$r_t^i = r_t^{\text{episodic}} \cdot \min\{\max\{\alpha_t, 1\}, L\}$$

- overtime the intrinsic reward reduces to episodic novelty

2.2 Agent Architecture

- the underlying mdp is changed by providing the reward:

$$r_t = r_t^e + \beta r_t^i$$

- where β governs how much we weigh exploration and how much we weigh exploitation
- the idea is to *jointly* learn a collection of policies, each parameters by a different value of β, so that we can used shared memory, and act we diverse set of behaviours in hard to explore games
- trained with a recurrent replay version of asynchronous DQN and an extra internal state embedding to increase robustness to intrinsic reward

3 Results

3.1 Disco Maze

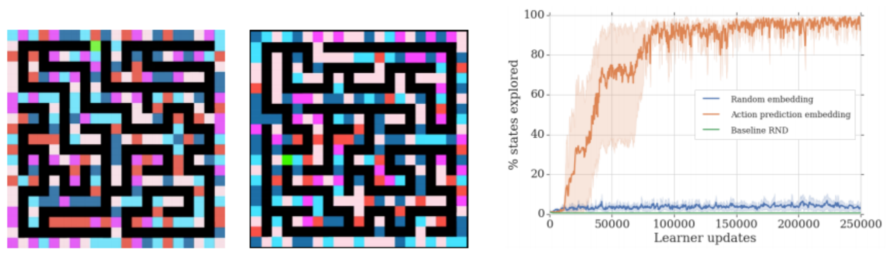

Figure 2: (Left and Center) Sample screens of Random Disco Maze. The agent is in green, and pathways in black. The colors of the wall change at every time step. (Right) Learning curves for Random projections vs. learned controllable states and a baseline RND implementation.

- fully observable maze env, random on each episode
- on each step, the colour so the maze completely change
- this would fool most novelty capturing intrinsic rewards, but NGU is able to survive without touching the wall, learning a form of DFS

3.2 Atari

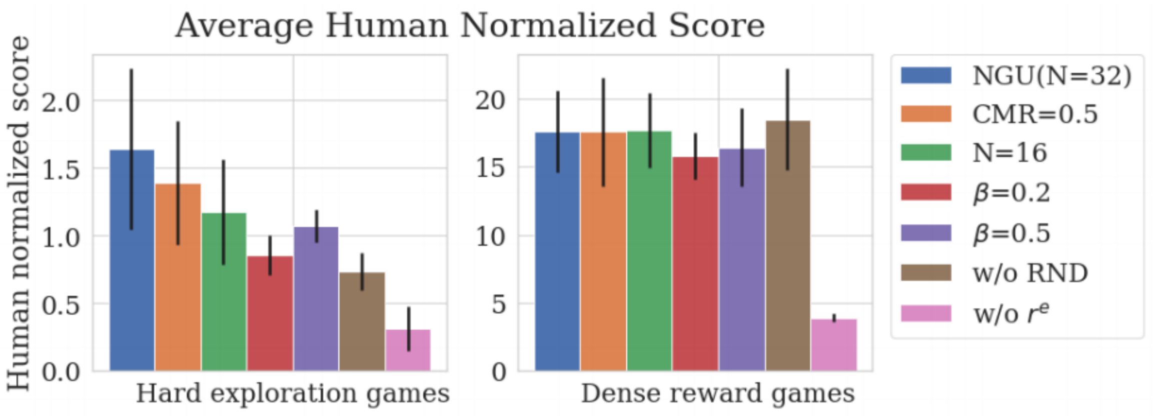

Figure 3: Human Normalized Scores on dense reward and hard exploration games.

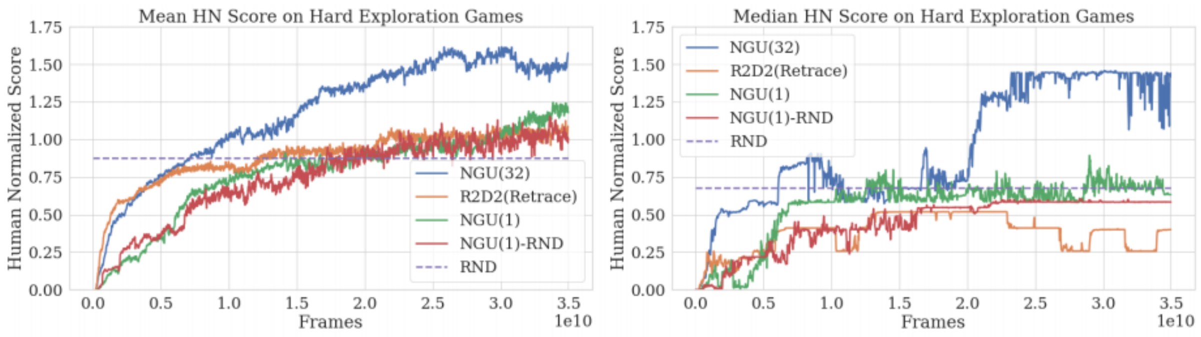

Figure 4: Human Normalized Scores on the 6 hard exploration games.