Transfer Learning with Random Network Distillation

Omer Gul[1]

Department of Computer Science[1]*, Stanford University, Stanford, CA 94305*

Motivation

- Reinforcement learning (RL) algorithms require **dense rewards** to achieve good performance and often fail with **sparse rewards.** To combat this, exploration bonuses (**intrinsic rewards**) are used.
- A persistent issue with RL is "**training on the test set**". It is important to measure the **generalizability** of our algorithms as well.
- **Random Network Distillation** (**RND**) is a state-of-the-art exploration bonus proposed by OpenAI. My goal is to determine how well policies trained with intrinsic rewards only **transfer** to other environments with intrinsic and extrinsic rewards.

Algorithms

- Proximal Policy Optimization (**PPO**) is a highly efficient and robust **policy gradient method**, alternating between sampling observations and performing **multiple epochs of optimization**.
- To minimize changes in policy, PPO uses the **surrogate loss**:

$$L(\theta) = \hat{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)],$$
$$\text{where } r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$$

- Advantages are computed using Generalized Advantage Estimation **(GAE)**:

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1},$$
$$\text{where} \quad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

- RND produces intrinsic rewards using a synthetic **prediction problem**. A **target** and **predictor network** sharing the same architecture are randomly initialized.
- The predictor network is made to **learn the outputs** of the target network on observations.
- The intrinsic reward is given by:

$$\|\hat{f}(\mathbf{x};\theta) - f(\mathbf{x})\|^2$$

- The intrinsic and extrinsic returns are estimated by **two separate value heads**.
- Extrinsic rewards are **episodic**, while intrinsic rewards are **non-episodic.**

Architectures

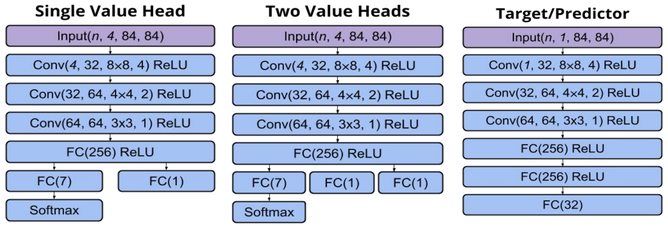

Figure 1: Shared network architecture for the policy and value functions

Figure 2: Network architecture used to compute intrinsic rewards

Sonic Benchmark

- The Sonic Benchmark uses the trilogy of **Sonic the Hedgehog** games on the Sega Genesis to **test for transferability**. All levels from the games are randomly split into **train** and **test** levels. I consider 8 train and 3 test levels for this project.
- A model is trained on the train set indefinitely. The learned policy is transferred to the test set and the **average total reward per episode** over 1 million frames is used as the evaluation metric.
- The states are the four most recent frames of the game, each converted to **grayscale** and **resized** to 84x84.
- The actions Sonic can take are {{LEFT}, {RIGHT}, {LEFT, DOWN}, {RIGHT, DOWN}, {DOWN}, {DOWN, B}, {B}}
- The main reward is Sonic's **horizontal offset**, normalized per level to sum to **45**. A reward for time linearly decaying from **5** is added.

Figure 3: Sonic rolling down a slope in Chemical Plant Zone

Figure 4: Sonic bouncing off a spring in Green Hill Zone

Figure 5: Sonic defeating an enemy in Death Egg Zone

Methodology

- Following Nichols et al, I use PPO as a **baseline**.
- I first consider the performance of PPO on the test levels **without** any transfer learning.
- I then **jointly train** a PPO agent on all training levels for **2 million frames** each. The learned policy is used for evaluation on test set.
- For my proposed model, I combine PPO and RND as done by Burda et al.
- I similarly evaluate one RND agent **without** any transfer learning and then train another RND agent jointly on all training levels using **1 million frames** per environment.
- All networks are trained using Adam with a learning rate of 0.00025

Analysis

- Standard PPO **outperformed** each method.
- Training for all methods is highly **unstable**, with the total reward fluctuating regularly over training without a clear trend.
- In the case of Joint PPO, transfer learning results in greater stability early on in training.
- The quality of transfer learning is dependent on the **quality of the transferred policy**. Although Joint RND has a better average score than standard RND, the maximum values it attains are lower.
- The number of parallel environments is a determining factor of the performance of RND.

Results

Algorithm	Chemical Plant Zone Act 2	Green Hill Zone Act 3	Death Egg Zone Act 2	Average
PPO	**8.40345395**	**3.03227703**	3.320360146	**4.918697042**
Joint PPO	5.017792998	1.761001096	**4.05359179**	3.610795295
RND	1.107330284	1.267312163	1.696329927	1.356990791
Joint RND	2.404323031	1.207537965	1.365825208	1.659228735

Table 1: Average of the total rewards per episode obtained after training the methods on the test levels for 1 million timesteps

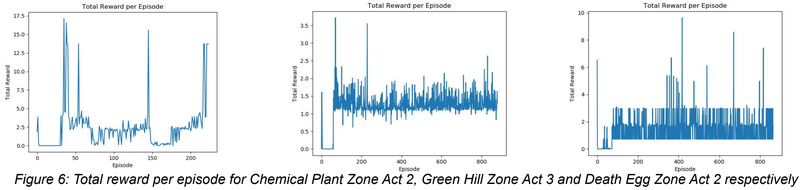

Figure 6: Total reward per episode for Chemical Plant Zone Act 2, Green Hill Zone Act 3 and Death Egg Zone Act 2 respectively

Algorithm	Angel Island Zone Act 1	Oil Ocean Zone Act 1	Spring Yard Zone Act 1	Ice Cap Zone Act 1	Launch Base Zone Act 1	Hill Top Zone Act 1	Hydrocity Zone Act 2	Labyrinth Zone Act 2
Joint PPO	17.77521718	8.742787544	4.319085513	26.20803445	8.729264529	3.793846229	4.709055931	11.42299043
Joint RND	5.191281242	1.591397213	1.306103895	25.53148789	2.383504646	1.483599225	0.9581138524	7.907593397

Table 2: Average of the total rewards per episode obtained while training the models jointly on the training levels.

Future Plans

- Similar to Burda et al, I intend on replicating these experiments with a **larger number of parallel environments.**
- I will do further **hyperparameter tuning** to make training RND more stable.
- I finally intend on testing the performance of **recurrent policies** rather than convolutional policies on RND.

References

- Burda, Y., Edwards, H., Storkey, A., Klimov, O. (2018). Exploration by random network distillation. ArXiv:1810.12894 [Cs.LG], , 1-17.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529-533.
- Nichol, A., Pfau, V., Hesse, C., Klimov, O., Schulman, J. (2018). Gotta learn fast: A new benchmark for generalization in RL. ArXiv:1804.03720v2 [Cs.LG],
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O. (2017). Proximal policy optimization algorithms. ArXiv:1707.06347v2 [Cs.LG], , 1-12.