Generative Networks to Simulate Environment Dynamics and Improve Sample Efficiency in Reinforcement Learning

Trevor Barron
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University
Tempe, Arizona, 85281
tpbarron@asu.edu
Final Draft

***Abstract*—Sample efficiency is a major obstacle to scaling reinforcement learning to real world problems. In this project I propose learning an action-conditioned generative model representing the environment dynamics and suggest several ways of using this model to intelligently augment training data. I also hope to analyze the learning behavior when incorporating a generative model. To align with the goals of this course I will use environments whose state is an image.**

1. Introduction

Reinforcement learning (RL) in continous state and action spaces has been well studied but remains a difficult problem. Convergence to a solution often requires many samples and it can be difficult to fully explore the state space as some states are only accessible after long sequences of actions.

Unlike many other applications in machine learning, data is far less available in RL. Often RL algorithms are trained by interaction with a simulated environment as in the well known successes with Atari 2600 games. A major obstacle in scaling reinforcement learning to the real world is sample efficiency. We can simulate an Atari game millions of times on a computer but we cannot execute a command on a robot millions of times due to time and hardware constraints.

There are two clear approaches to solving the problem above. Either (1) improve the algorithms to require less data or (2) find more data.

Both of these approaches have been explored recently and, not surprisingly, both have provided improvements. **I propose a method to tackle this problem that falls between the two approaches: make an RL algorithm more sample efficienct by incorporating a generative model trained on the environment dynamics that intelligently simulates additional environment data.** This can be thought of as a sort of *imagination* that the agent can interact with instead of interacting with the true environment. This approach relies on an assumption that is it is easier to model the dynamics of an environment than it is to learn a complex task within that environment. I posit that this is a reasonable assumption as an agent must understand the dynamics before it can behave intelligently in those dynamics.

This would work by training an action-conditioned generative model on data from the true environment to predict both the successive state and reward given the previous state and action. The key idea is then to use the latent space of the generative model to sample sequences in an intelligent manner for training our agent. For example, we could sample from the z-space prioritizing locations where the agent has a high temporal difference error or according to a metric that maximizes information regarding the environment dynamics.

Key aspects of the project will include:

- Empirical analysis of action-conditioned video generation models including BiGANs and autoencoders with and without recurrence
- Analysis of trajectories in z-space of generative models
- Analysis of methods for choosing the point in z space to generate trajectories including:
  - Temporal difference error
  - Information maximization (using variational methods)
- Ideally, a mathematical analysis of how training on sample from a generative model biases learning.

2. Details of the generative model

Formally, we would like to train a generative model to express the dynamics of a Markov Decision Process (MDP) where S is the state set, $A(s)$, $s \in S$ is the action set in state s, and R is the reward set. Then we model the MDP with a start state distribution $s_0 \sim P_0(S)$, a state transition probability $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$, and a reward $r_t = r(s_t, a_t, s_{t+1}) \in R$. We want a model that jointly approximates both the state transition probability and the reward probability. Precisely, a function, $f : S_t \times A_t \rightarrow S_{t+1} \times R_t$. I have yet to conclude whether this needs to be probabilistic or not and that may depend on the metric used to draw samples.

I plan to use a network structure similar to that proposed by Leibfried, et al. It is similar to an auto-encoder structure

where an input image is mapped to a latent space at which point it is combined with an action. The target is the successive state and reward. Their paper claims to have success predicting as far as 200 steps in the future. I may experiment with adding recurrence as these states, actions, and rewards are clearly temporally correlated. I have spent much time looking into which model would be ideal for this application and will consider using a Generative Adversarial Network [2]. I would need a BiGAN [1] in order to map from each state back to the z-space. If time permits I may try a BiGAN structure on a simple problem to examine if its accuracy improves upon the autoencoder model.

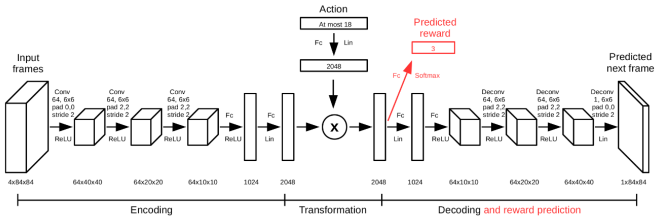

Figure 1. Network structure proposed in [4]

The generative model requires samples from the true environment for training. **To align with the goals of this class I will work on problems where each state is an image.** This turns the problem into an action-conditioned video prediction task where each each output state also has an associated reward.

3. Details of the trajectory generation

Once a sufficienctly accurate generative model has been trained we generate novel trajectories by choosing an initial z value, generating a state, and having the agent interact with the generated trajectory. The main advancement comes from intelligently choosing the initial value for z. Since we know all of the trajectories that have been experienced by the agent we can find how these trajectories move through the z-space. Given this information we would then like to sample trajectories starting from positions in z space chosen intelligently. I have thought of two promising metrics to do this. First, we can select positions in z space where the agent had a high temporal difference error. Essentially, these are locations where the agents predictions were poor. This method is inspired by Prioritized Experience Replay in DQN [6], which also gauges a states importance by TD-error. Second, we can choose positions that maximize our information of the environment dynamics. We can model this with information theory as choosing the points that provide the maximal information gain regarding our environment model.

Note that this starting point does not need to be a state in the start state distribution of the MDP. Concretely, imagine playing a game of Breakout. When learning it may be more advantagous to play a specific portion of the game repeatedly than to start at the beginning for each game.

This framework provides a potential solution to the sample efficiency problem by generating data from a dynamics model learned during training. In practice an algorithm would likely alternate between interacting with the true environment, updating the generative model and interacting with the simulated environment.

4. Analysis of influence of training on generated samples

Finally, I would like to try to formally analyze what effect training on generated samples has on the convergence of a policy. For example, if we restrict our models to linear systems then we have convergence guarantees for a wide range of RL algorithms. If we incorporate a generative model into the training procedure does that bias learning? Do we still converge in the limit? Or will noise in the generative model lead to instability?

5. Related work

There has been a significant amount of related work. Though I have not seen any work that uses generated sequences for training.

Last year a team at Google published work on speeding up DQN using a model trained on the environment dynamics [3]. They use a linear gaussian model to predict environment states which they added to the DQN's replay memory. They report that it was difficult to produce long range sequences but their work highlights the potential of using a learned model.

Similarly there has been considerable recent work on video prediction [4], [5], [7]. I take inspiration from these papers for details on modeling video.

References

[1] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *CoRR*, abs/1605.09782, 2016.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[3] Shixiang Gu, Timothy P. Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep q-learning with model-based acceleration. *CoRR*, abs/1603.00748, 2016.

[4] Felix Leibfried, Nate Kushman, and Katja Hofmann. A deep learning approach for joint video frame and reward prediction in atari games. *CoRR*, abs/1611.07078, 2016.

[5] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder P. Singh. Action-conditional video prediction using deep networks in atari games. *CoRR*, abs/1507.08750, 2015.

[6] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *CoRR*, abs/1511.05952, 2015.

[7] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 613–621. Curran Associates, Inc., 2016.