An Experimentation Toolkit for Robotics Control and Manipulation Tasks using Reinforcement Learning Algorithms

A Robot Learning Gym

Ashwin Reddy[1]

[1]The Harker School

March 22, 2017

Introduction

Goals of the project:

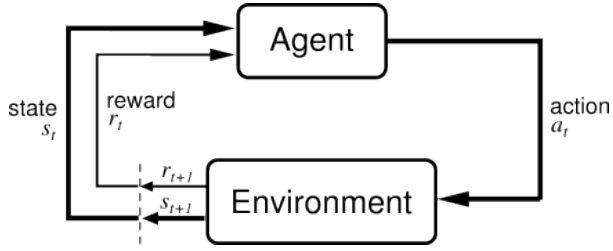
- Build a toolkit for Deep Robot RL experimentation using a simulator
  - Collect models and tasks online
  - Create an API to interface them
- Create a simple benchmark

Motivation

- Such a tool does not currently exist, but it would be helpful
- Standard metrics missing for robot learning algorithms
- Enables more Sim-to-Real learning
- Does not require a real robot to operate

Background: RL

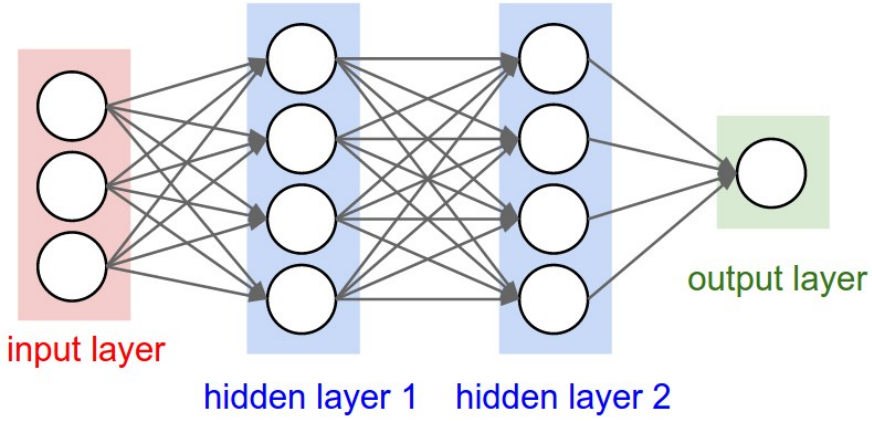
- Traditional Reinforcement Learning Paradigm (MDPs):



- How can we train the agent to pick actions to maximize reward?
- Goal: Learn parameters θ to stochastic policy $\pi_\theta(a_t|s_t)$
- Objective η is max $\mathbb{E}\left[\gamma^t \int_t^H r(t)\mathbf{d}t\right]$

Background: ML

- Machine Learning Paradigm (using Neural Networks):



- Goal: make predictions
- Given many samples of $(x, f(x))$, learn $h_\theta(x) \approx f(x)$

Current Work

- Algorithms are starting to use deep machine learning:
  - Google DeepMind's Deep-Q-Learning which has learned to play Atari games
  - UC Berkeley's Guided Policy Search which taught a humanoid robot to assemble toy airplanes
- Different ML Architectures:
  - Modular Neural Networks
  - Progressive Neural Networks
  - Reccurent Neural Networks

What makes Deep RL for Robotics hard?

- Lack of large datasets (i.e: poor supervision)
- Noisy input and can't be sure of outputs
- Occlusions
- Running on a real robot is expensive
- Curse of dimensionality
- Sparse and time-delayed rewards
- Exploration vs. Exploitation
- Continuous control harder than discretized actions

Current Tools

- OpenAI Gym: Platform for testing general RL algorithms
- MuJoCo: Efficient simulator used by OpenAI Gym
- OpenAI RL Lab: implements some common RL algorithms
- Guided Policy Search: A type of Deep RL algorithm that has a package available online, also uses MuJoCo
- DeepMind's Lab: 3D first-person games for testing Deep RL algorithms

All have elements of robot learning, but don't provide a framework for it

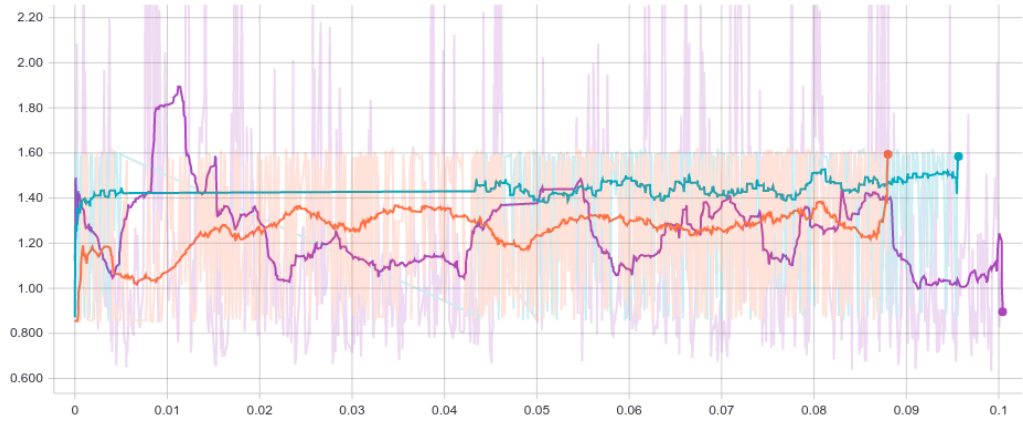
Experimental Design

- Collect MuJoCo Robot Models
- Build an open-source framework with models and other tools
  - OpenAI MuJoCo Python bindings and Gym
- Run RL algorithms

Data Collection

- Task: Peg Insertion with $r = \frac{1}{||x - x^*||}$
- Methods
  - Random: take a sample from the action space
  - Cross-Entropy Method: sample parameters, evaluate, and reuse best ones
  - Policy Gradient Method: Use trajectory $\langle s, a, r, s' \rangle$ to $\theta_{k+1} \leftarrow \theta_k + \alpha \nabla_\theta \eta$
  - Guided Policy Search: uses trajectory optimization to train the neural net based policy

Graphs (Reward vs Time)



- Collected using TensorFlow's TensorBoard Visualizer, using a 0.6 smooth
- Teal line is policy gradient, orange line is cross-entropy, purple is random
- Random has erratic behavior and inconsistent results as expected
- Cross entropy is somewhat unstable but leads to more consistent results
- Policy Gradient seems to have reached a local optimum

Discussion and Future Work

- Continue to maintain the framework and create documentation
- Experiment with modular networks
- Benchmark more deep learning based algorithms
- Try different tasks and rewards

Acknowledgements and References

- Thanks to
  - Mr. Martin Baynes for sponsoring this project
  - Soroush Nasiriany, an undergraduate researcher at UC Berkeley, for feedback on this project