Evaluation of Park

Adding the packet classification problem

Harrison Brown for R244
• Platform for researchers to experiment with RL
• 12 systems problems with an easy to use interface
  • Focus on algorithmic challenges
Motivation for Park

- OpenAI gym
  - Interface to experiment, train, evaluate, compare models
- No standard platform for systems problems
  - Helpful for systems researchers
  - Abstracts away systems challenges

Algorithm 2 Interface for simulated interaction.

1: `def env.step(action):`
2: `# OpenAI Gym style of interaction`
3: `server.reply(action)`
4: `state, reward, done = server.listen()`
5: `return state, reward, done`
Goals

Evaluate and extend Park by adding a new RL systems problem: packet classification

Park = 12 Systems Problems
Add packet classification
Park = 13 Systems Problems
Neural Packet Classification (Liang et al., 2019)

- Match a network packet to a rule from a set of rules
  - Objective: minimize the classification time and memory footprint
- Software solutions typically use a decision tree
  - Provides perfect accuracy by construction
  - Several different implementations using heuristics
- NeuroCuts
  - Deep RL solution to build decision trees
NeuroCuts Methods and Formulation

- States: current decision tree
- Action: cut a node or partition a set of rules
- Reward: classification time, memory footprint, or combination of the two
  - Rewards are sparse and delayed, nearly a one-step decision problem
    - Problem is adapted for RL, encodes nodes to fixed size based on dimensions
- For this problem, can cheaply generate samples
Aim of my work

• Adding the packet classification problem to Park
  • Complete environment that measures rewards, produces action spaces, and steps the agent
• Build and train an agent for this problem using the actor-critic method described in the paper or PPO
• Evaluate the usability and extensibility of the Park project
Progress and Plan

- Currently negligible, have random agents running on some of provided problems in Park domain
- True understanding of problem, actor-critic/PPO methods
- Add environment to Park problem set
- Adapt an off-the-shelf implementation of RL algorithm to problem
- Measure performance using provided benchmarks
References
