DQN and Atari Games

Ron Parr
CompSci 590.11

About Atari Games

• Atari 2600 was an early generation video game that accepted ROM cartridges enabling a variety of games on the same hardware – arguably the first widely successful system of this type

• Had very limited storage, graphics, sound, computation
  • 160x192 resolution
  • 128 bytes of RAM
  • 4K ROM cartridges
  • 1.2 Mhz processor

• Simple controller with 1 button, 8-directional joystick

• In short, it was the equivalent of an Xbox, Playstation or Switch in RP’s youth
Why Atari as an RL Benchmark?

- Easy to simulate
- Widely available
- Small(ish) discrete action space
- Large number of games possible in a common platform
- Diversity in types of games, including many that required somewhat long term behavior
- No difficult object recognition problems involved (graphics too crude)
- Not obviously easy

Some challenges

- Single frame is not a Markovian state (partial solution: stack frames)

- Games designed for human time scale responses, for changing actions every 1/60 second (solution: make actions sticky)

- Flicker – some objects appeared only in odd or even frames [see, e.g., the ghosts in Pac-Man] (partial solution: input is max over two adjacent frames)
Switch to David Silver’s Slides
(We’ll jump in at slide 11, and return after slide 21)

Lessons learned

• From TD-Gammon to DQN surprisingly little as changed
  • Still no stability or performance guarantees despite changes
  • Training still requires massive amounts of data
  • Convnets, small changes in training make a big difference (as in deep nets)

• Yet everything has changed
  • After years of frustration in applying RL to hard problems, now people want to apply RL to everything
  • Harder games
  • Power management in data centers
  • Robotic control
After DQN/Atari

• Some concern that community is focused too much on game playing

• Learning (only) value/Q-functions from images is limiting
• Combine with recurrent network techniques (e.g. LSTM) to handle state that isn’t directly observable
• Combine with search for for turn-based games with known models

Unsatisfying aspects of DQN/Atari success

• No high level knowledge (all new games learned from scratch)

• Training time is quite large (50M frames)

• Solutions lack robustness (adding irrelevant “distractor” graphics can cause strange behavior)

• Some evidence that solutions may be partly memorized (poor generalization)