Open AI Five

Steven Kollortz
Heidelberg University

July 25, 2019
Overview

1. Defense of the Ancients 2
   - Why choosing DotA
   - Complexity

2. Timeline

3. How did they do it?
   - Size
   - Architecture
   - Proximal Policy Optimization
   - Learning
Why choosing DotA

- One of the most popular games on twitch
- Runs on Linux
- Supports an API
- Partially-observed state
- High-dimensional, continuous action and observation space
- Long term planning
- Hoped that in order to solve it, it would require new techniques
Complexity

- An average game lasts around 45 minutes
- With 30 frames per second resulting in 80000 ticks
- OpenAI Five observes every fourth frame

<table>
<thead>
<tr>
<th>Game</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>40</td>
</tr>
<tr>
<td>Go</td>
<td>150</td>
</tr>
<tr>
<td>DotA 2</td>
<td>20000</td>
</tr>
</tbody>
</table>

**Table:** Number of moves before a game usually ends

- 170,000 possible actions per hero with an average of 1000 valid each tick
Complexity

- Big observation space
- 20000 numbers representing what a human would be able to see as well
- Mostly floating points

<table>
<thead>
<tr>
<th>Game</th>
<th>Observation space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>8x8 board with 6 pieces plus minor history</td>
</tr>
<tr>
<td>Go</td>
<td>19x19 board with 2 pieces plus Ko</td>
</tr>
<tr>
<td>DotA 2</td>
<td>20000 numbers</td>
</tr>
</tbody>
</table>

**Table:** Size of observation space
Timeline

- November 2016 development started
- May 2017 1.5k mmr tester (bottom 15%) is better than the bot
- Early June bot beats 1.5k mmr player
- Late June: bot beats 3k mmr
- July bot beats 7.5k mmr
Timeline

- August 7th beat Blitz (6.2k former pro) 3-0, Pajkatt (8.5k pro) 2-1 and CC&C (8.9k pro) 3-0.
  They bet Sumail (8.3k pro, top 1v1 player) would win against the bot.
- August 9th beat Arteezy (10k pro, top player) 10-0. He says Sumail could figure out this bot.
- August 10th beat Sumail 6-0, Sumail said it is unbeatable.
- Sumail also played the August 9th version where he goes 2-1.
- A lot of people play the bot afterwards, but do not win in a standard game.
- September 7th the first pro beat it with normal gameplay.
Timeline

- June 2018 OpenAI five on amateur / semi pro level (4-6k mmr) with restricted rules
- Early August 2018 plays on roughly 6-7k mmr with lightly restricted rules (18 heros)
- Loses against against pro teams (7-8k mmr) at The International 8
- Wins against different pro teams 2-0 from October to February 2019, most notably Alliance with team earnings over 3 million dollar.
Timeline

- April 2019 defeats OG, the winner of TI 8.
- April 2019 OpenAI five arena opens where it ends up with a score of 7215–42.
- 10 losses were against the same team.
A doubling of computation every 3.5 months
## Computation

<table>
<thead>
<tr>
<th></th>
<th>1v1 bot</th>
<th>OpenAI five</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPUs</td>
<td>60000 cores on Azure</td>
<td>128000 cores on GCP</td>
</tr>
<tr>
<td>GPUs</td>
<td>256 K80</td>
<td>256 P100</td>
</tr>
<tr>
<td>Experience</td>
<td>300 years per day</td>
<td>180 years per day per hero</td>
</tr>
<tr>
<td>Observation size</td>
<td>3.3 kB</td>
<td>36.8 kB</td>
</tr>
<tr>
<td>Observations per second</td>
<td>10</td>
<td>7.5</td>
</tr>
<tr>
<td>Batch size</td>
<td>8,388,608</td>
<td>1,048,576</td>
</tr>
<tr>
<td>Batches per minute</td>
<td>20</td>
<td>60</td>
</tr>
</tbody>
</table>
Architecture

OpenAI Five Model Architecture

Available Actions:
- Softmax
- Sample/Argmax
- Selected Action
- Softmax
- Sample/Argmax
- Other X
- Softmax
- Sample/Argmax
- Other Y
- Softmax
- Sample/Argmax
- Hero X
- Softmax
- Sample/Argmax
- Hero Y
- Softmax
- Sample/Argmax
- Neutral X
- Softmax
- Sample/Argmax
- Neutral Y
- Softmax
- Sample/Argmax
- Mobile X
- Softmax
- Sample/Argmax
- Mobile Y
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Taunt
- Softmax
- Sample/Argmax
- Target Unit
Architecture

- Allied & enemy glyph cooldown
- is Night
- time until creepwave
- time since enemy courier last seen
- time until night
- courier number of flask, clarity, enchanted mangoes, town portals, magic sticks
- Total value courier items

nearby terrain 8x8 grid of height, traversability, creep occupancy for each hero in team
Architecture
Architecture
Architecture
Bots perspective
Proximal Policy Optimization

- Performs comparably to Trust Region Policy Optimization and Actor Critic with Experience Replay
- Easier to implement
- Easier to tune
Proximal Policy Optimization

- TRPO and ACER approximate the second order derivative and its inverse
- PPO uses multiple epochs of stochastic gradient descent to perform each policy update
- Reduce the amount of bad decisions by penalizing or clipping the difference between the old and the new policy
- Clipping yielded best results
Learning

- Inverse reinforcement learning
- Self play
- 80% against itself and 20% against past versions
- Cost intensive
- Transfer learning
### Reward function

<table>
<thead>
<tr>
<th>Score</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experience</td>
<td>0.002</td>
</tr>
<tr>
<td>Gold</td>
<td>0.006</td>
</tr>
<tr>
<td>Mana</td>
<td>0.75</td>
</tr>
<tr>
<td>Hero Health</td>
<td>2</td>
</tr>
<tr>
<td>Last Hit</td>
<td>0.16</td>
</tr>
<tr>
<td>Deny</td>
<td>0.2</td>
</tr>
<tr>
<td>Kill</td>
<td>-0.6</td>
</tr>
<tr>
<td>Death</td>
<td>-1.0</td>
</tr>
<tr>
<td>Mega creeps</td>
<td>4.0</td>
</tr>
<tr>
<td>Win</td>
<td>2.5</td>
</tr>
</tbody>
</table>
Reward function

- Negative reward for leaving the lane early in training
- Zero sum rewards
- No communication channel
- Teamspirit is controlled by a parameter $\tau$

$$hero\_rewards[i] = \tau \times \text{mean}(hero\_rewards) + (1 - \tau) \times hero\_rewards[i]$$

- $\tau$ anneals from 0.2 to 0.97 during training
- Later rewards are discounted by half, roughly every 10 minutes
Rapid

- Rapid is a reinforcement learning training system
- Supports Kubernetes, Azure and GCP
- Allows to run PPO in massive scale
- Synchronous gradient descent globally synchronized
- 58MB of parameters have to be synchronized
- takes 0.3 seconds to synchronize 512 GPUs
Rapid

Optimizer + Connected Rollout Workers (x256)

Rollout Workers
~500 CPUs
- Run episodes
  - 80% against current bot
  - 20% against mixture of past versions
- Randomized game settings
- Push data every 60s of gameplay
  - Discount rewards across the 60s using generalized advantage estimation

Optimizer
1 p100 GPU
- Compute Gradients
  - Proximal Policy Optimization with Adam
  - Batches of 4096 observations
  - BPTT over 16 observations

Eval Workers
~2500 CPUs
- Play in various environments for evaluation
  - vs hardcoded “scripted” bot
  - vs previous similar bots (used to compute Trueskill)
  - vs self (for humans to watch and analyze)

Model Parameters
(10M floats)

Optimizers use NCCL2 to average gradients at every step.

Gradient Updates
References

https://openai.com/blog/ai-and-compute/
https://liquipedia.net/dota2/The_International/2018/Main_Event
https://openai.com/blog/how-to-train-your-openai-five/
https://arena.openai.com/#/results
https://openai.com/blog/openai-five/
https://openai.com/blog/more-on-dota-2/
https://openai.com/blog/openai-baselines-ppo/#ppo
https://medium.com/@jonathan_hui/rl-proximal-policy-optimization-ppo-explained-77f014ec3f12
https://gist.github.com/dfarhi/66ec9d760ae0c49a5c492c9fae93984a
Questions?