Emergent Complexity via Multi-agent Competition

Bansal et al. 2017

CS330 Student Presentation
Motivation

● Source of complexity: environment vs. agent

● Multi-agent environment trained with self-play
  ○ Simple environment, but extremely complex behaviors
  ○ Self-teaching with right learning pace

● This paper: multi-agent in continuous control
Trusted Region Policy Optimization

- Expected Long Term Reward: $\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t) \right]$
Trusted Region Policy Optimization

- Expected Long Term Reward: \( \eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t) \right] \)

- Trusted Region Policy Optimization:

\[
\eta(\pi) = \eta(\pi_0) + \mathbb{E}_{s_0, a_0, \ldots} \left[ \sum_{t=0}^{\infty} A_{\pi_0}(s_t, a_t) \right]
\]
Trusted Region Policy Optimization

- Expected Long Term Reward: \( \eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t) \right] \)

- Trusted Region Policy Optimization:

\[
\eta(\pi) = \eta(\pi_0) + \mathbb{E}_{s_0, a_0, \ldots, \pi} \left[ \sum_{t=0}^{\infty} A_{\pi_0}(s_t, a_t) \right] \\
= \eta(\pi_0) + \sum_{s} \rho_\pi(s) \sum_{a} \pi(a|s)A_{\pi_0}(s, a)
\]
Trusted Region Policy Optimization

- Expected Long Term Reward: \( \eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t) \right] \)

- Trusted Region Policy Optimization:

\[
\eta(\pi) = \eta(\pi_0) + \mathbb{E}_{s_0, a_0, \ldots, \pi} \left[ \sum_{t=0}^{\infty} A_{\pi_0}(s_t, a_t) \right] \\
= \eta(\pi_0) + \sum_s \rho_\pi(s) \sum_a \pi(a|s)A_{\pi_0}(s, a)
\]

- After some approximation: \( L(\pi) = \eta(\pi_0) + \sum_s \rho_{\pi_0}(s) \sum_a \pi(a|s)A_{\pi_0}(s, a) \)
Trusted Region Policy Optimization

- Expected Long Term Reward: \( \eta(\pi) = \mathbb{E}_{s_0,a_0,\ldots} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t) \right] \)

- Trusted Region Policy Optimization:

\[
\eta(\pi) = \eta(\pi_0) + \mathbb{E}_{s_0,a_0,\ldots,\pi} \left[ \sum_{t=0}^{\infty} A_{\pi_0}(s_t, a_t) \right] \\
= \eta(\pi_0) + \sum_s \rho_\pi(s) \sum_a \pi(a|s) A_{\pi_0}(s, a)
\]

- After some approximation: \( L(\pi) = \eta(\pi_0) + \sum_s \rho_{\pi_0}(s) \sum_a \pi(a|s) A_{\pi_0}(s, a) \)

- Objective Function: \( \max_\theta L_{\theta_{old}}(\theta) - CD_{KL}(\theta_{old}, \theta) \)
Proximal Policy Optimization

- In practice, importance sampling:

\[
\text{maximize } \hat{E}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)} \hat{A}_t - \beta \text{KL}[\pi_{\theta_{old}}(\cdot | s_t), \pi_\theta(\cdot | s_t)] \right]
\]
Proximal Policy Optimization

- In practice, importance sampling:

\[
\max_{\theta} \mathbb{E}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)} \hat{A}_t - \beta \text{KL}[\pi_{\theta_{old}}(\cdot | s_t), \pi_\theta(\cdot | s_t)] \right]
\]

- Another form of constraint:

\[
\mathbb{E}_t \left[ \min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_t) \right]
\]
Proximal Policy Optimization

- In practice, importance sampling:
  \[
  \maximize_{\theta} \hat{E}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)} \hat{A}_t - \beta \text{KL}[\pi_{\theta_{old}}(\cdot | s_t), \pi_\theta(\cdot | s_t)] \right]
  \]

- Another form of constraint:
  \[
  \hat{E}_t \left[ \min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t) \right]
  \]

- Some intuition:
  - First term is the function with no penalty/clip
  - Second term is an estimation with the probability ratio clipped
  - If a policy changes too much, its effectiveness extent will be decreased
Environments for Experiments

- Two 3D agent bodies: ants (6 DoF & 8 Joints) & humans (23 DoF & 12 Joints)
Environments for Experiments

- Two 3D agent bodies: ants (6 DoF & 8 Joints) & humans (23 DoF & 12 Joints)
- Four Environments:
  - Run to Goal: Each gets +1000 for reaching the goal, and -1000 for its opponent
Environments for Experiments

- Two 3D agent bodies: ants (6 DoF & 8 Joints) & humans (23 DoF & 12 Joints)
- Four Environments:
  - **Run to Goal**: Each gets +1000 for reaching the goal, and -1000 for its opponent
  - **You Shall Not Pass**: Blocker gets +1000 for preventing, 0 for falling, -1000 for letting opponent pass
Environments for Experiments

- Two 3D agent bodies: ants (6 DoF & 8 Joints) & humans (23 DoF & 12 Joints)
- Four Environments:
  - **Run to Goal**: Each gets +1000 for reaching the goal, and -1000 for its opponent
  - **You Shall Not Pass**: Blocker gets +1000 for preventing, 0 for falling, -1000 for letting opponent pass
  - **Sumo**: Each gets +1000 for knocking the other down
Environments for Experiments

- Two 3D agent bodies: ants (6 DoF & 8 Joints) & humans (23 DoF & 12 Joints)
- Four Environments:
  - **Run to Goal:** Each gets +1000 for reaching the goal, and -1000 for its opponent
  - **You Shall Not Pass:** Blocker gets +1000 for preventing, 0 for falling, -1000 for letting opponent pass
  - **Sumo:** Each gets +1000 for knocking the other down
  - **Kick and Defend:** Defender gets extra +500 for touching the ball and standing respectively
Large-Scale, Distributed PPO

- 409k samples per iteration computed in parallel
- Found L2 regularization to be helpful
- Policy & Value nets: 2-layer MLP, 1-layer LSTM
- PPO details:
  - Clipping param = 0.2, discount factor = 0.995
Large-Scale, Distributed PPO

- 409k samples per iteration computed in parallel
- Found L2 regularization to be helpful
- Policy & Value nets: 2-layer MLP, 1-layer LSTM
- PPO details:
  - Clipping param = 0.2, discount factor = 0.995
- Pros:
  - Major Engineering Effort
  - Lays groundwork for scaling PPO
  - Code and infra is open sourced
- Cons:
  - Too expensive to reproduce for most labs
Opponent Sampling

- Opponents are a natural curriculum, but **sampling method** is important (see Figure 2)
- Latest available opponent leads to **collapse**
- They find random old sampling works best

Figure 2: Opponent Sampling: Training rewards for two opponent sampling strategies.
Opponent Sampling

- Opponents are a natural curriculum, but **sampling method** is important (see Figure 2)
- Latest available opponent leads to **collapse**
- They find random old sampling works best

**Pros:**
- Simple and effective method

**Cons:**
- Potential for more rigorous approaches

Figure 2: Opponent Sampling: Training rewards for two opponent sampling strategies.
Exploration Curriculum

- **Problem:** Competitive environments often have sparse rewards
- **Solution:** Introduce dense rewards:
  - *Run to Goal*:
    - Distance from goal
  - *You Shall Not Pass*:
    - distance from goal, distance of opponent
  - *Sumo*:
    - Distance from center
  - *Kick and Defend*:
    - Distance ball to goal, in front of goal area
- Linearly anneal exploration reward to zero:
  \[ r_t = \alpha_t s_t + (1 - \alpha_t) I[t == T] R \]
Emergence of Complex Behaviors
Emergence of Complex Behaviors
Effect of Exploration Curriculum

- In every instance the learner with the curriculum outperformed the learner without.
- The learners without the curriculum optimized for a particular part of the reward, as can be seen below.
Effect of Opponent Sampling

- Opponents were taken from a range of $\delta \in [0, 1]$ with 1 being the most recent opponent and 0 being a sample taken from the entire history.
- On the sumo task:
  - Optimal $\delta$ for humanoid is 0.5
  - Optimal $\delta$ for ant is 0
Learning More Robust Policies - Randomization

- To prevent overfitting, the world was randomized
  - For sumo, the size of the ring was random
  - For kick and defend the position of the ball and agents were random

Figure 4: Win-rate of kicker vs iterations with full randomization
Learning More Robust Policies - Ensemble

- Learning an ensemble of policies
- The same network is used to learn multiple policies, similar to multi-task learning
- Ant and humanoid agents were compared in the sumo environment
This allowed the humanoid agents to learn much more complex policies
Strengths and Limitations

Strengths:
● Multi-agent systems provide a natural curriculum
● Dense reward annealing is effective in aiding exploration
● Self-play can be effective in learning complex behaviors
● Impressive engineering effort

Limitations:
● “Complex behaviors” are not quantified and assessed
● Rehash of existing ideas
● Transfer learning is promising but lacks rigorous testing
Strengths and Limitations

Strengths:
- Multi-agent systems provide a natural curriculum
- Dense reward annealing is effective in aiding exploration
- Self-play can be effective in learning complex behaviors
- Impressive engineering effort

Limitations:
- “Complex behaviors” are not quantified and assessed
- Rehash of existing ideas
- Transfer learning is promising but lacks rigorous testing

Future Work: More interesting techniques to opponent sampling