Emergent Complexity Via Multi-Agent Competition

Bansal et al.[1]

Presentation by Chris Larsen
NeAR-YaDLL Graduate Researcher

October 30, 2018

Introduction

- Competition and cooperation are vital concepts for AI
- Long term goals include understanding and utilizing these concepts for more intelligent and generalizable agents
- Competition Theory in Human Evolution

Introduction

In this work:

- Environments in RL are extremely important
- More complex environments needed for more complex agents? (hint: no)
- Competition creates complexity

Outline

Multi-agent Markov

- Set of states S
- Set of observations of each agent $O^1, ..., O^N$
- A transition function
- A reward function for each agent
- Each agent's policy function parameterized by a neural network
- Each agent aims to maximize its own expected reward.

Policy Gradients

- Recall that value based reinforcement learning algorithms depend on learning and acting (e-greedy) on accurate approximated value states
- In policy gradient methods the parameters of the policy are directly optimized to output a probability distribution over possible actions to take. In other words the parameters of the policy are being updated to find an optimal policy.

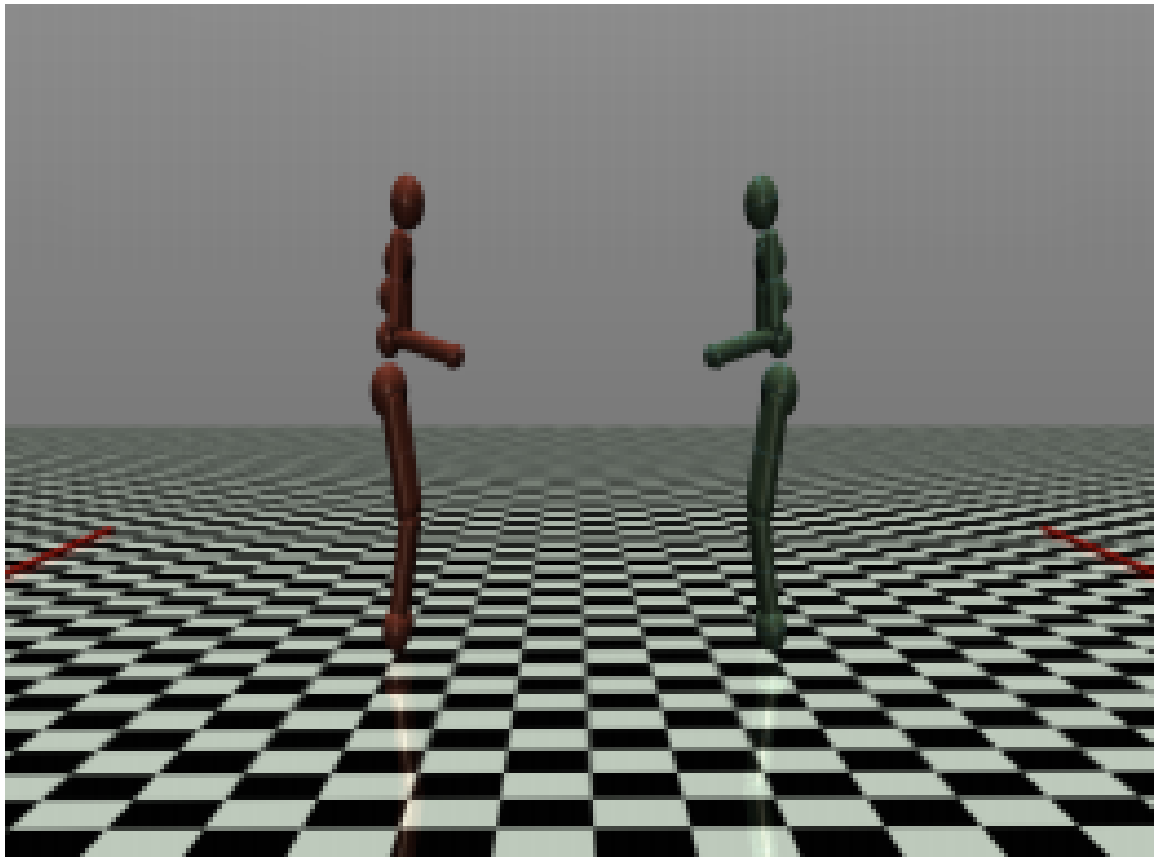
Proximal Policy Optimization

- In this work agents were trained using a popular OpenAI policy gradient method called PPO

policy. Let $l_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ denote the likelihood ratio. Then PPO optimizes the objective: $L = \mathbb{E}\left[\min(l_t(\theta)\hat{A}_t, \text{clip}(l_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$, where $\hat{A}_t$ is the generalized advantage estimate and $\text{clip}(l_t(\theta), 1-\epsilon, 1+\epsilon)$ clips $l_t(\theta)$ in the interval $[1-\epsilon, 1+\epsilon]$. The algorithm alternates be-

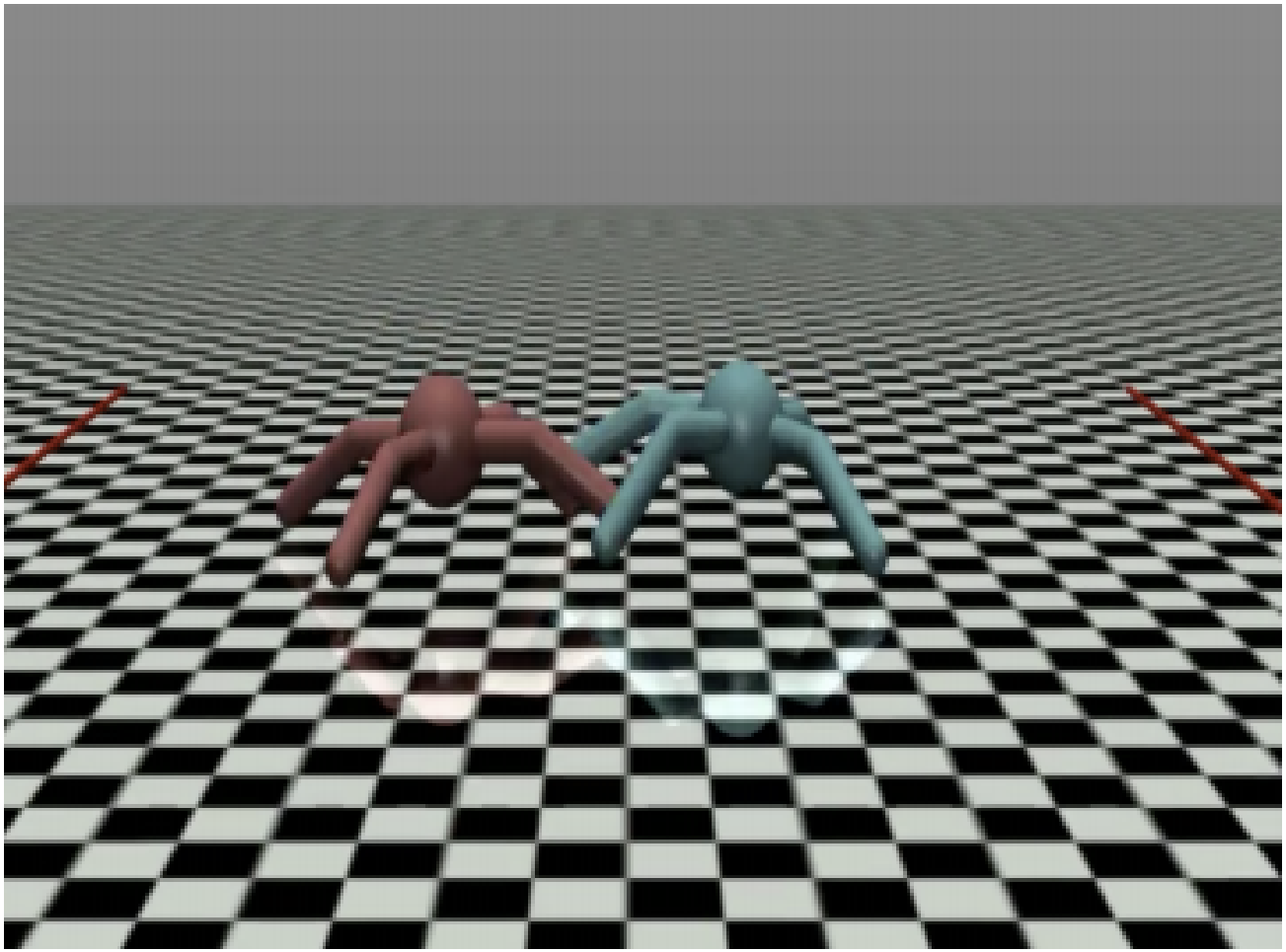
- Main points to remember about PPO is that it is based on Trust Region Policy Optimization which limits the size of the gradient changes
- If an action is taken and the outcome is better than expected do more of that but don't do to much

Run to the Goal



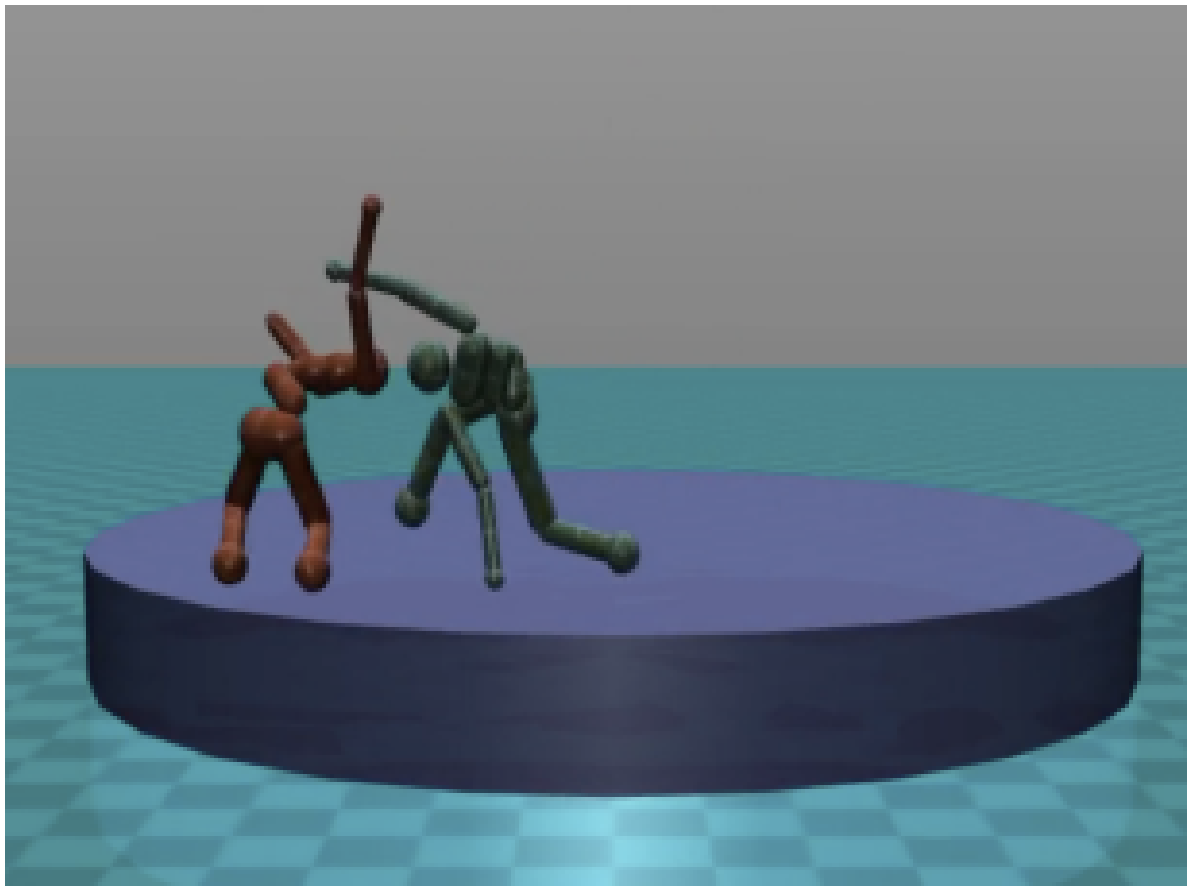
First agent to reach the goal wins
(+1000 if you win, -1000 if you lose)

You Shall Not Pass



Same set up as before but attacker/defender (+1000 for attacker if it crosses the line, +1000 to the defender if it doesn't and the defender is standing.)

Sumo



Knock the opponent off the platform
(+1000 if winner and -1000 if pushed off)

Kick and Defend

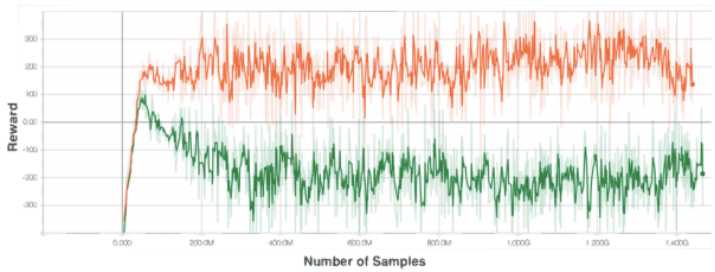
Standard shootout
(+1000 if goal is scored to attacker, and -1000 to opponent, +500 for defender if standing and goal is not scored and another +500 if the defender made contact with the ball)

Exploration Curriculum

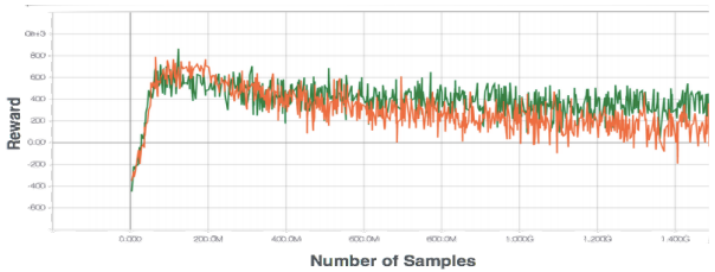
- Sparse Rewards are hard, and a very difficult problem to address
- In this paper they use a technique called exploration reward ("prior knowledge" injection)
  - In the exploration reward, a simple dense reward is given at the beginning of training to encourage standing upright or walked.
  - This exploration reward is linearly annealed to 0 in favor of the competition reward (10-15% of epochs)
  - Examples: distance to goal, velocity in x-direction, control cost, impact cost, standing reward

Opponent Sampling

- All experiments ran using opponent pairs (e.g. you were paired with one partner for the entire training
- Each new episode you could train by using the latest policy network for each agent
- Or you could train against a random old iteration of your opponent leads to more stable training and more robust policies



(a) Latest available opponent



(b) Random old opponent

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

Results

- For "run to the goal" and "you shall not pass" MLP was used
- For "sumo" and Shootout" LSTM was used
- Adam optimizer (0.001)
- PPO clipping $e = 0.2$

Results - Learned Behaviors

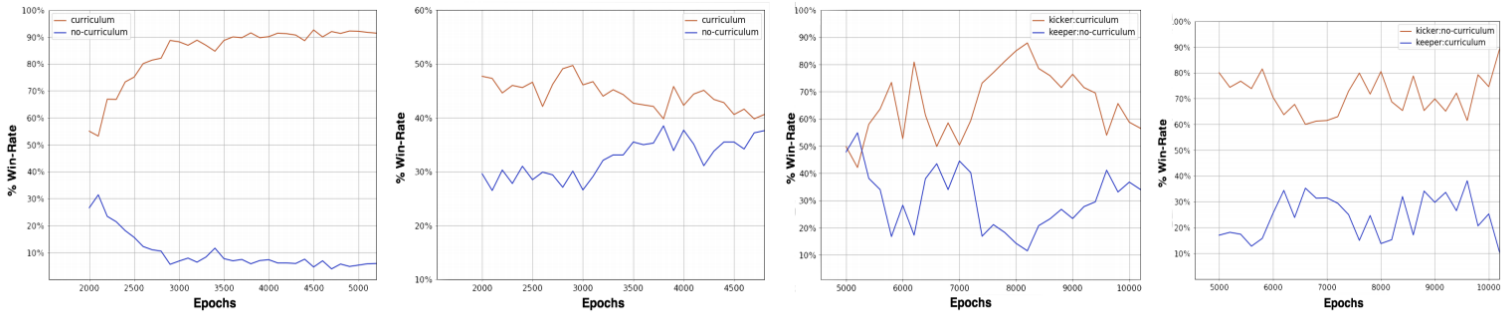
- Run To The Goal
  - "blocking, standing robustly, using legs to topple the opponent"
- You Shall Not Pass
  - "blocking humanoid learn to block by raising its hand while the other humanoid eventually learned to duck in order to cross"
- Sumo
  - "stable fighting stance and learned to knock the opponent using their heads"
- Kick and Defend
  - "kick the ball high and tries to avoid the defender"
  - "fooling behavior in the kickers motions where it moves left and right quickly once close to the ball to fool the defender"
  - "moving in response to the motion of the kicker and using its hands and legs to obstruct the ball"

Generalization

Some experiments were done to determine if policies could be used for other tasks
For example the sumo agent was tested by some wind force and needed to stand up right

Exploration Criteria

- Authors argued that the learned complex behavior was not due to the dense reward signal but rather the nature of competitive play



(a) Humanoid Sumo (b) Ant Sumo (c) Kick-and-Defend

Paper Discussion

- Over-fitting was a problem
  - Add randomization to the environment (i.e. ball position)
  - Use ensemble methods where an agent learns multiple policies simultaneously and for each roll-out of each policy a random other policy is used as the competitor.

- Overall authors showed complex qualitative behaviors from simple environments induced by competition

Thoughts

- The quantitative results and experimentation section was lacking any rigor or useful results
- Although the results weren't great the idea was proven that complex behavior can emerge from competitive play.
- Future work may look at;
  - Adding in some belief state or representation of the opponent in the agent design.
  - Using this strategy in single agent settings (artificially induce a multi-agent game from a single agent one)

Questions

Citation

1. Bansal, T., Pachocki, J., Sidor, S., Sutskever, I. and Mordatch, I. (2018). Emergent Complexity via Multi-Agent Competition. [online] Arxiv.org. Available at: https://arxiv.org/abs/1710.03748 [Accessed 27 Oct. 2018].

Opponent Sampling

δ	1.0	0.8	0.5	0.0	$\mathbb{E}$[Win]
1.0	-	0.26	0.13	0.37	0.25
0.8	0.46	-	0.22	0.52	0.40
0.5	0.59	0.58	-	0.73	**0.63**
0.0	0.55	0.36	0.16	-	0.35
$\mathbb{E}$[Loss]	0.53	0.40	**0.17**	0.54	-

(a) Humanoid Sumo

δ	1.0	0.8	0.5	0.0	$\mathbb{E}$[Win]
1.0	-	0.37	0.35	0.29	0.34
0.8	0.36	-	0.38	0.33	0.36
0.5	0.36	0.39	-	0.33	0.36
0.0	0.51	0.49	0.49	-	**0.50**
$\mathbb{E}$[Loss]	0.41	0.42	0.41	**0.32**	-

(b) Ant Sumo

Table 1: The effect of opponent sampling. $\mathbb{E}$[Loss] and $\mathbb{E}$[Win] are the expected loss and win-rates for agents trained with a particular δ as described in 5.4. For humanoid $\delta = 0.5$ gives highest win-rate and lowest loss, whereas for Ant $\delta = 0$ was best.

Ensemble Methods

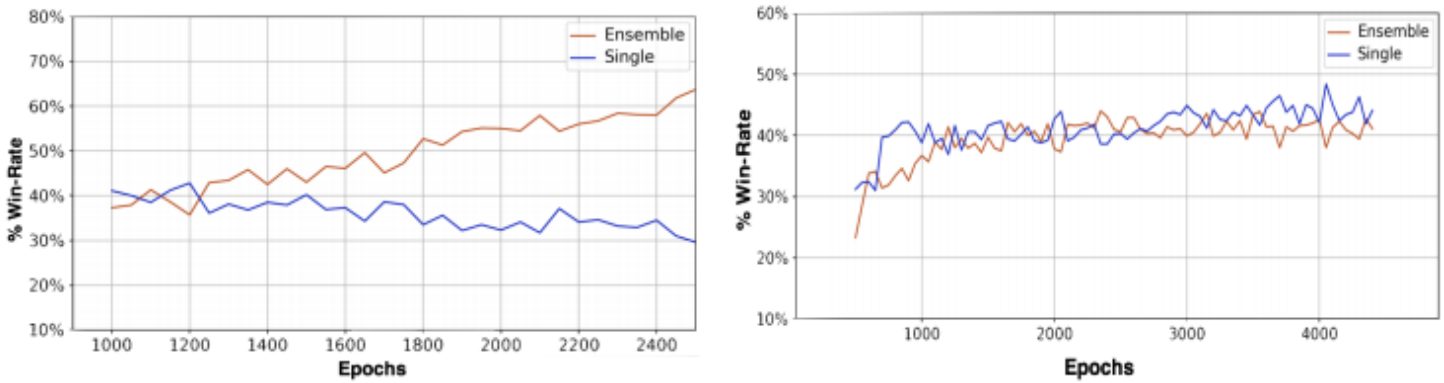


Figure 5: % Win-rate of agents trained in ensemble vs agents trained with just a single policy. Humanoid Sumo (left) and Ant Sumo (right).