Competitive Reinforcement Learning for Autonomous Cyber Operations

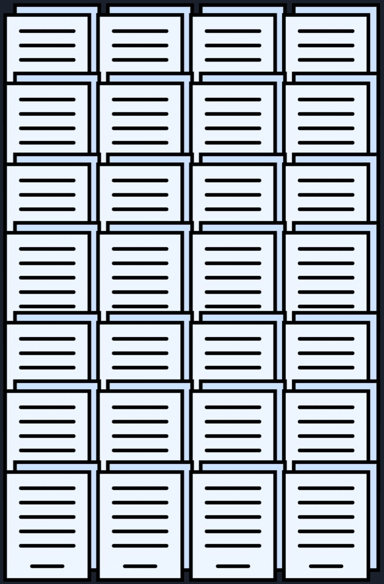
Authored by Garrett McDonald

Affiliated with the *Royal Military College of Canada Department of Electrical and Computer Engineering*

Published on May 1, 2023

Presented by Jimmy Franknedy

Problem - Assist Analysts



Data and Incident Reports

Security Operation Center Analysts

Network Analysis Software

Network Security Systems

Artificial Intelligence

Current Work & Limitations - Skewed and Training Set Up

Skewness

Red ACOs dominated by expert systems

Blu ACOs mainly experimental and simulated

Difficulty in setting up training

R.L training requires a lot of environment detail and hardware speed

Current Work & Limitations - Static Performance

Static Training

R / B agents often train against static opponents

Unknown Potential

Lack of research in determining if R & B agents can both learn optimal policies in parallel

New Approach - Competitive R.L.

AlphaGo

Competitive R.L.

Training agents to make decisions by interacting with an environment and other agents simultaneously to achieve similar / different goals.

Motivation

R.L. Success

R.L. has proven it can master intricate strategies in competitive games and devising innovative approaches to outplay human experts.

Threat model

Training Attacks

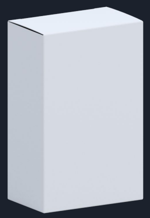
Testing Attacks

Adaptive

Strategic

Meta

Attacker Knowledge



Contribution - Main Idea & Context

Main Idea

"Determine if Red and Blue agents can be trained using competitive RL to approximate their game theory optimal policies in a simulated ACO environment"

Reinforcement Learning Context

Single Agent RL
(Standard RL)

Deep RL

Value Prediction DNN *(Critic)*

Policy Prediction DNN *(Actor)*

+ Temporal Differential Learning

Actor-Critic Framework

Proximal Policy Optimization

Multi Agent RL

Competitive

Cooperative

Mixed

Value Based Alg.

Policy Based Alg.

Counterfactual Regret Minimization (CFR)

Fictitious Play (FP)

Next Up...

(1) Implementation

(2) Evaluation

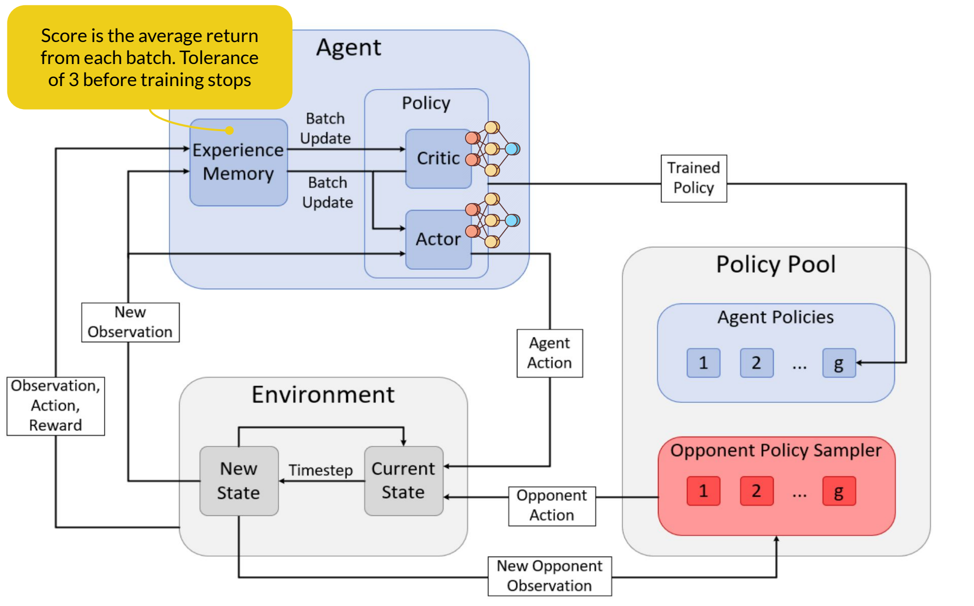
(3) Limitation

Contribution - Implementation

Algorithm 3 Fictitious Play for ACO Environments.

$\Gamma \leftarrow$ initialize training environment
$\pi_0 \leftarrow$ set random initial policies for generation 0 $(\pi_0^{blue}, \pi_0^{red})$
$g = 0$
while within computational budget **do**
  $g \leftarrow g + 1$
  for each player i in $[blue, red]$ **do**
    $\pi_g^i \leftarrow$ set random initial policy for player i generation g
    while π_g^i is improving **do**
      $M^i \leftarrow$ clear memory buffer to store new batch of samples
      while memory buffer M^i is not full **do**
        $\pi^{-i} \leftarrow$ select opponent policy from the pool Π^{-i}
        $\Gamma \leftarrow$ reset the training environment for a new game
        $M^i \leftarrow$ store samples $(u_t^i, a_t^i, r_{t+1}^i, u_{t+1}^i)$ for every timestep t in Γ
      end while
      $\pi_g^i \leftarrow$ update policy using PPO for batch of samples M^i
    end while
    $\Pi^i \leftarrow$ add new policy π_g^i to pool
  end for
end while
Return $(\pi_g^{blue}, \pi_g^{red})$

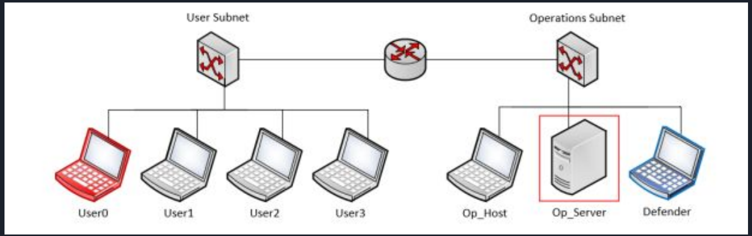
Fictitious Play Implementation



Actor-Critic Framework with Opponent Sampling

Contribution - Evaluation Setup

Network Topology

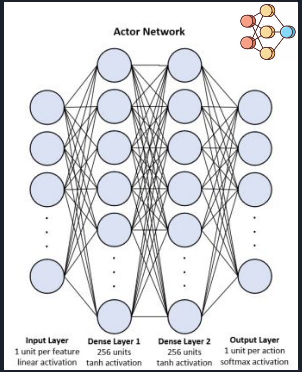


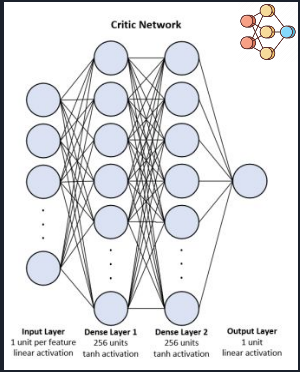
Attacker & Defender Details

- DiscoverRemoteSystems()
- DiscoverNetworkServices()
- ExploitRemoteService()
- Escalate()
- Impact()

- Analyse()
- Remove()
- Restore()

Framework





Parameters

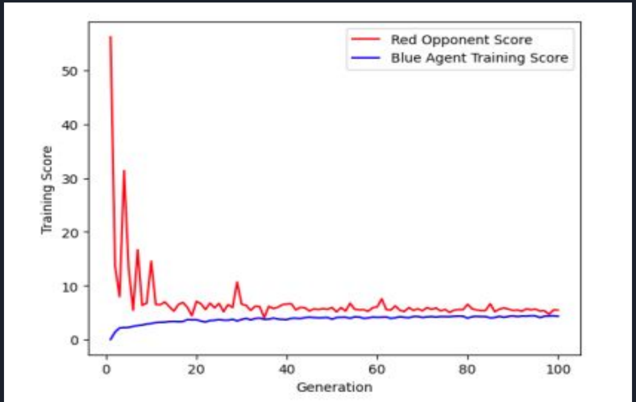
Fictitious Play Parameters	
Generations	100
Tolerance	3
Best-Response Mixing Parameter	0.9
Minmax Evaluation Games	50
PPO Parameters	
Hidden Layers	2
Activation Function	tanh
Units per Hidden Layer	256
Batch Size	61440 (5120 Games)
Learning Rate	1e-3
Discount Factor	0.99
Parallel Workers	40

Rewards

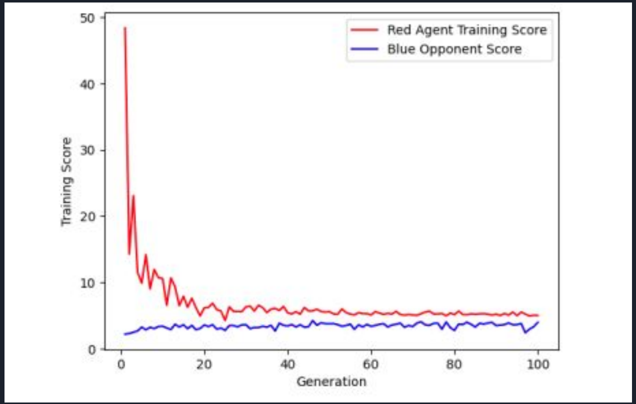
1. Impacting the Op Server. If Red uses the impact action while it has Root privileges on the Op Server, Red scores 10 points.
2. Foothold on an Ops Device. Red scores 1 point for each Ops device it has Root privileges on at the end of a timestep.
3. Foothold on a User Host. Red scores 0.1 points for each User host it has Root privileges on at the end of a timestep.
4. Blue Defender Restores a Device. Red scores 1 point every time Blue restores a device to a clean image.

Contribution - Evaluation Analysis

Nash Equilibrium



Training scores for fictitious play, solving for the Blue minmax policy.



Training scores for fictitious play, solving for the Red minmax policy.

Average scores are taken for each generation when that policy finished training
(i.e average score during the final batch of training was recorded for every new generation)

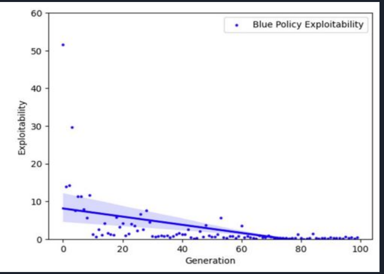
Validation Process

Measured Exploitability

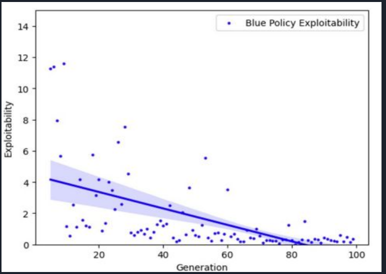
Examined Minmax Policy

Contribution - Validation Analysis (1/2)

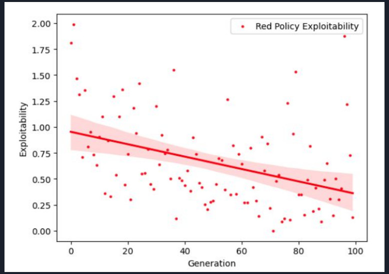
Measuring "Exploitability"



Blue generation's relative exploitability



Same as (left) but first five generations dropped



Red generation's relative exploitability

Difference between an agent's worst-case performance score and what the agent could have achieved by following an optimal policy.

Guaranteed expected return against any possible opponent

$$expl = \mathbb{E}[G \mid \pi_*^i, \pi_*^{-i}] - \mathbb{E}[G \mid \pi^i, \pi_*^{-i}]$$

Agent *i* MinMax Policy for game

Policy used by agent *i*

Opponent's optimal response

Exploitability Calculation

$$\mathbb{E}[G \mid \pi^{blue}, \pi_*^{red}] = \max_{\pi^{red}} \frac{\sum_{k=0}^{50} G(\pi^{blue}, \pi^{red})}{50}$$

$$\mathbb{E}[G \mid \pi_*^{blue}, \pi^{red}] = \min_{\pi^{blue}} \frac{\sum_{k=0}^{50} G(\pi^{blue}, \pi^{red})}{50}$$

Guaranteed expected return

Given a Blu MinMax Policy and a Red policy

Find a Red policy's minimum expected score across all possible opponents

Play a game 50 times and average the score

Worst-Case Performance Score

The max score that a Blue policy allows, and the min score that a Red policy achieves

$$expl(\pi^{blue}) = \mathbb{E}[G \mid \pi^{blue}, \pi_*^{red}] - \mathbb{E}[G \mid \pi_*^{blue}, \pi_*^{red}]$$

$$expl(\pi^{red}) = \mathbb{E}[G \mid \pi_*^{blue}, \pi^{red}] - \mathbb{E}[G \mid \pi_*^{blue}, \pi_*^{red}]$$

Exploitability of a red policy

Expected return given blue's optimal policy and a red policy

Expected return given blue's optimal policy and a red optimal policy

Since the true game theory optimal policy is unknown, the Blue and Red agents with the best minmax scores are used as optimal (benchmark of 0 exploitability)

Relative Exploitability Calculation

Contribution - Validation Analysis (2/2)

MinMax Evaluation

Confirm exactly how accurately each agent has approximated a non-exploitable policy

New dedicated opponents will be trained against each competitive policy using single agent RL with PPO.

train

Lowest expected score across every Blue opponent was 3.97

Dedicated Blue opponent is able to hold the competitive Red agent to a maximum expected score of 3.97

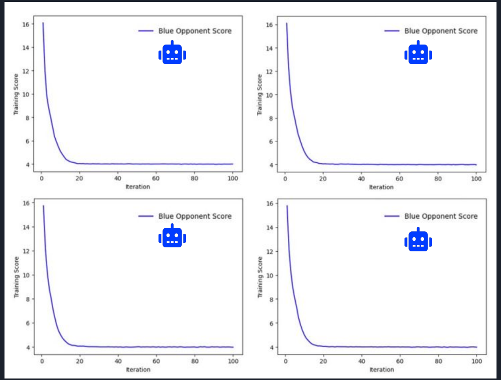
train

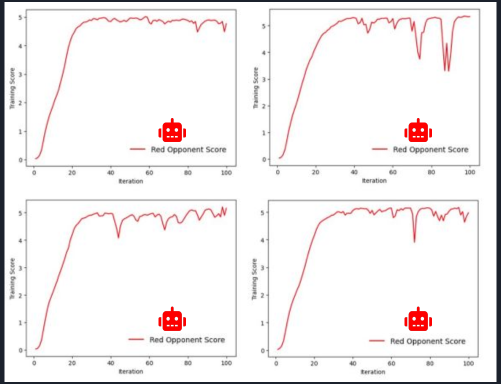
Highest expected score across every Red opponent was 5.39

Competitive Blue agent held the dedicated Red opponent to a score of 5.36

Optimal Min/Max Policy

Dedicated Agent





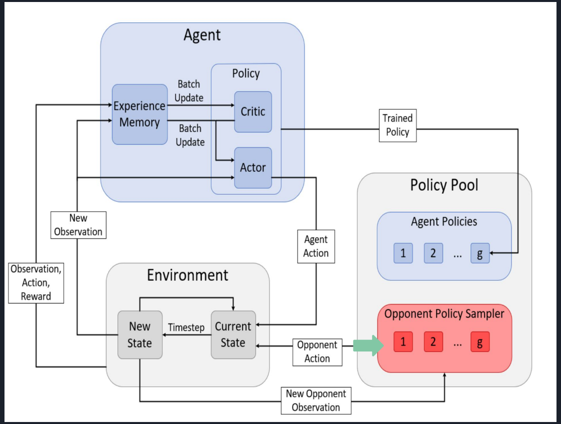
Dedicated Agent Performance

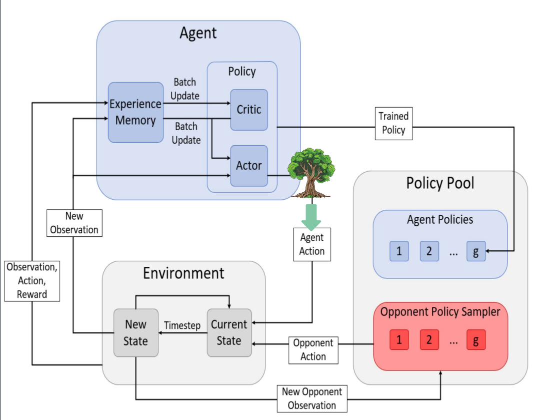
vs.	Comp. Red	Ded. Red	Rand. Red
Comp. Blu	4.79	5.38	0.03
Ded. Blu	4.02	4.98	0.03
Rand. Blu	15.43	12.24	4.06

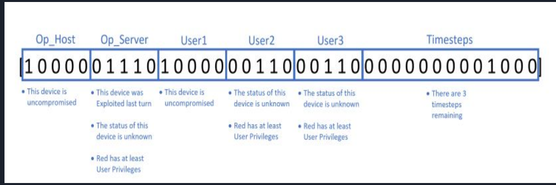
Expected scores were gathered by having each pair of agents play 1000 games and taking their average score

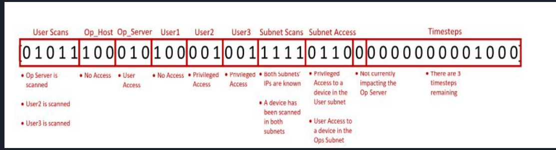
Agent Expected Score Comparison

Contribution - Limitations









Sampling

Uniform sampling, early generations have a significant impact on the existing pool, but later generations

Historic Sampling

Quality Sampling

Forward Search

Agent's value estimation should rely at the depth of the search tree, instead of estimating value at the current state

Realistic Observation Vector

Future research will need to consider what a viable observation vector will look like in a realistic simulation

Final Thoughts