Increasing the Upper Bound for the EvoMan Game Competition

Gabriel-Codrin **Cojocaru** Sergiu-Andrei **Dinu** (presenter)
Eugen **Croitoru**

Faculty of Computer Science "Al. I. Cuza" University Iasi, Romania

SYNASC 2020

TOC

Introduction

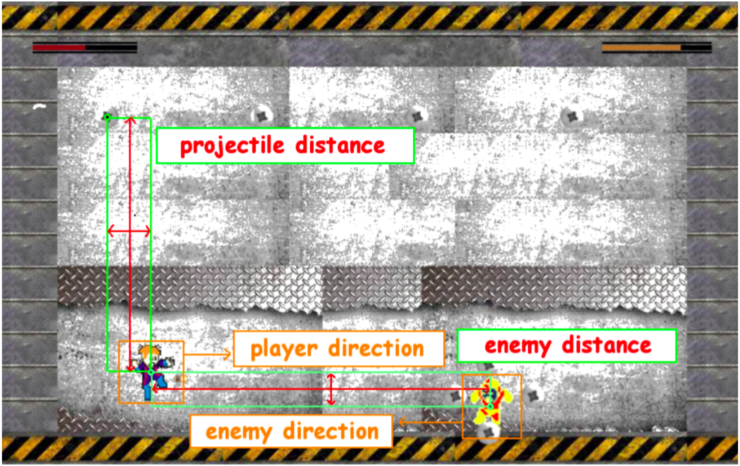
Problem description

Search Algorithms

Results

Introduction: Game

- Aim: play the Megaman/EvoMan game



Introduction: Motivation

- Games are an opportunity to improve our algorithms.
- They can be low-cost simulators for interesting problems or topics.
- EvoMan was used by a competition at WCCI 2020 (where we placed second).
- We found out that we can greatly improve on the upper bound provided.

Introduction: The difference

- The initial competition required training with 4 opponents, and testing on all 8.
- Our approach works better on simply fighting each opponent.
- It significantly improves the state of the art results.

Problem description: Evaluation

$$gain = 100.01 + player\_life - enemy\_life$$

- Both start with 100 life points.
- the absolute maximum score is 200.01.
- There are 8 opponents, each with a different strategy.
- There are 5 difficulty levels, each lowering the damage the player does, and increasing the damage the player takes.

Problem description: Agent

- The game outputs 20 sensor values, updated each simulation frame.
- Each frame, we can perform 5 actions (e.g. move left, shoot).

Problem description: Agent

- The game outputs 20 sensor values, updated each simulation frame.
- Each frame, we can perform 5 actions (e.g. move left, shoot).
- We chose to place an ANN in this loop.

Problem description: Agent

- The game outputs 20 sensor values, updated each simulation frame.
- Each frame, we can perform 5 actions (e.g. move left, shoot).
- We chose to place an ANN in this loop.
- We also remember the past 2 frames' sensors values, and our past 2 actions, for a total of 62 inputs.

Search Algorithms: 2-stage approach

- Idea: use an exploratory algorithm to find a good starting point.

Search Algorithms: 2-stage approach

- Idea: use an exploratory algorithm to find a good starting point. Tested:
  - Random initialisation.
  - Q-Learning.
  - Genetic Algorithm (GA).
  - Particle Swarm Optimisation (PSO).

Search Algorithms: 2-stage approach

- Idea: use an exploratory algorithm to find a good starting point. Tested:
  - Random initialisation.
  - Q-Learning.
  - Genetic Algorithm (GA).
  - Particle Swarm Optimisation (PSO).
- then use an exploitative algorithm to refine that game-playing strategy. We've only used a Proximal Policy Optimisation (PPO) algorithm for this stage.

Search Algorithms: initial comparison

- No pre-exploration algorithm performed better than random initialisation.
- The game is too difficult for basic strategies to constitute the basis for optimal strategies (a misleading / trap function landscape).

Results: Experiment

- PPO ran for between 1000 to 3000 epochs, per opponent, 30 repeats for each datapoint.
- Time: 1000 epochs require ≈ 10000000 ($10M$) Evoman frames.
- We ran specialised and generalised PPO models.
- Specialised means 8 models, 1 for each of the 8 opponents, 1000 epochs.
- Generalised means 1 model for all 8 opponents, 3000 epochs.

Results: vs the upper bound

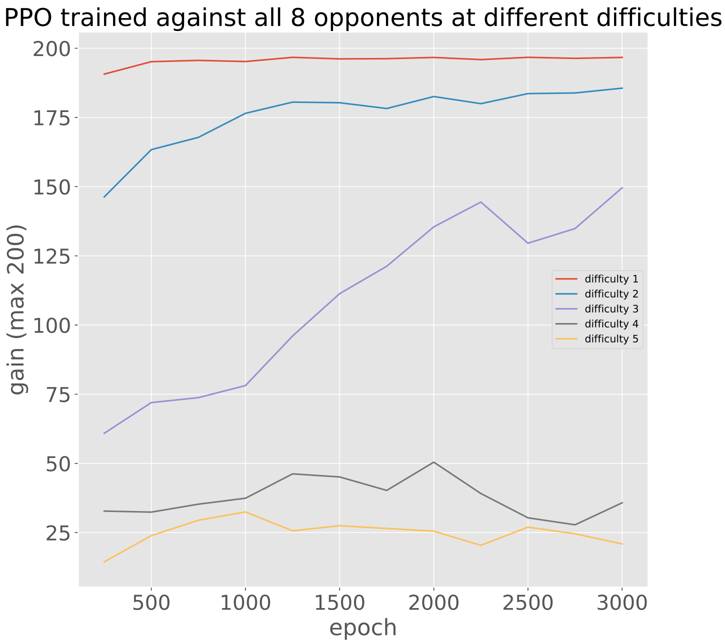
Table: PPO generalised vs PPO specialised vs NEAT specialised, difficulty = 2

Opponent	**PPO gen.**	**PPO spec.**	**NEAT spec.**
1	199.54	199.67	190.01
2	200.01	199.61	194.01
3	194.81	199.94	180.01
4	180.61	195.95	194.01
5	187.77	198.03	194.01
6	174.43	199.67	173.01
7	183.07	197.73	177.01
8	169.31	195.07	186.01
harmonic mean	185.57	198.19	185.67

Results: all difficulties, generalised

Table: PPO (generalised) trained against all opponents (Gain)

Opponent	**Difficulty**				
	1	**2**	**3**	**4**	**5**
1	200.01	199.54	170.75	155.48	41.08
2	200.01	200.01	182.81	168.81	168.51
3	199.38	194.81	164.31	127.54	130.61
4	186.14	180.61	154.23	181.58	60.28
5	196.56	187.77	185.34	142.41	158.31
6	196.35	174.43	179.19	23.84	9.88
7	199.26	183.07	95.40	8.01	8.28
8	196.57	169.31	123.00	41.34	10.01
harmonic mean	196.68	185.57	149.57	35.76	20.89



PPO trained against all 8 opponents at different difficulties
gain (max 200)
epoch
200
175
150
125
100
75
50
25
500
1000
1500
2000
2500
3000
difficulty 1
difficulty 2
difficulty 3
difficulty 4
difficulty 5

Results: all difficulties, specialised

Table: PPO (specialised), trained 1000 epochs (gain)

Opponent	**Difficulty**			
	2	**3**	**4**	**5**
1	199.68	196.51	189.21	166.01
2	199.61	199.91	199.21	197.18
3	199.94	199.51	147.21	72.28
4	195.95	196.14	198.81	191.23
5	198.03	193.38	191.73	196.71
6	199.67	195.18	196.85	198.76
7	197.73	192.45	184.85	176.01
8	195.07	185.37	178.05	172.61
harmonic mean	198.19	194.7	184.12	154.58

Results: percentage of games lost

Table: PPO - percentage of games lost

Opp.	Difficulty								
	Generalised					Specialised			
	1	**2**	**3**	**4**	**5**	**2**	**3**	**4**	**5**
1	0	0	20	23.33	100	0	0	0	13.33
2	0	0	0	0	0	0	0	0	0
3	0	0	0	13.33	3.33	0	0	36.67	100
4	0	0	0	0	60	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	100	100	0	0	0	0
7	0	0	56.66	100	100	0	0	0	0
8	0	0	10	100	100	0	0	0	0

Conclusions

- We have surpassed the state-of-the-art through an exploitative and time-consuming algorithm.
- It is a difficult, misleading problem, since optimal game-playing strategies cannot be easily inferred from suboptimal strategies.
- Thus, the cascade method cannot work well. A memetic algorithm (e.g. PSO + PPO) seems feasible.

- Evoman: Game-playing Competition for WCCI 2020, http://pesquisa.ufabc.edu.br/hal/Evoman.html
- Evoman: Game-playing Competition for WCCI 2020, results http://pesquisa.ufabc.edu.br/hal/Evoman.html#results
- Fabricio Olivetti de Franca, Denis Fantinato, Karine Miras, A.E. Eiben and Patricia A. Vargas. "EvoMan: Game-playing Competition" arXiv:1912.10445
- de Araújo, Karine da Silva Miras, and Fabrício Olivetti de França. "An electronic-game framework for evaluating coevolutionary algorithms." arXiv:1604.00644 (2016).
- Floreano, D., Dürr, P. & Mattiussi, C. Neuroevolution: from architectures to learning. Evol. Intel. 1, 47–62 (2008). https://doi.org/10.1007/s12065-007-0002-4
- M. MEGA, "Produced by capcom, distributed by capcom, 1987," System: NES.

Watkins, C.J.C.H., Dayan, P. Q-learning. Machine Learning 8, 279–292 (1992)

Holland J.H., Genetic Algorithms and Adaptation. Adaptive Control of Ill-Defined Systems, 1984, Volume 16 ISBN 978-1-4684-8943-9

Kennedy, J.; Eberhart, R. (1995). "Particle Swarm Optimization". Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942–1948.

Kenneth O. Stanly; Rist Miikkulainen (2002). "Evolving Neural Networks through Augmenting Topologies". Evolutionary Computation, Volume 10, Issue 2, Summer 2002, p.99-127, `https://doi.org/10.1162/106365602320169811`

John Schulman, Filip Wolski, Prafulla Dhariwal, Alex Radford, Oleg Klimov (2017) "Proximal Policy Optimization Algorithms", arXiv:1707.06347v2

Moscato, P. (1989). "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms". Caltech Concurrent Computation Program (report 826).

Cybenko, G. (1989). "Approximations by superpositions of sigmoidal functions", Mathematics of Control, Signals, and Systems, 2(4), 303–314. doi:10.1007/BF02551274

Wolpert, D.H., Macready, W.G. (1997), "No Free Lunch Theorems for Optimization", IEEE Transactions on Evolutionary Computation 1, 67.

Karine Miras, Evoman, `https://karinemirasblog.wordpress.com/portfolio/evoman/`

Joshua Achiam, 2018, "Spinning Up in Deep Reinforcement Learning" `https://github.com/openai/spinningup`