Comparative Analysis of Learning Techniques with Atari Games

Mauricio Negrete (238917)

University of Trento

me.negreterodriguez@studenti.unitn.it

The following work is a comparative analysis by using Evolutionary Algorithms (E.A.) and Neural systems focusing mainly on Bio-Inspired algorithms. The agents primary goal is to acquire the ability to play a diverse range of classic Atari video games. To uncover their strengths, weaknesses and effectiveness in adapting to different gaming scenarios. The repository for this project is found in [1]

1. Introduction

Video games are one of the biggest media of entertainment and Artificial Intelligence (A.I.) plays a big factor shaping their immersive experiences, in this work, both settings are taken advantage of by using E.A. with a Genetic Algorithm and Neural systems such as NEAT and DQN — state-of-the-art algorithms in Neuro-evolution and Reinforcement Learning, respectively — to excel at learning how to play a range of video games.

The complexity in video games has increased over the years such that it might require several inputs by the user to perform several actions or solving complex puzzles, because of this, the comparative analysis made here will focus in a much simpler task: learn how to play classic Atari games.

Given the diverse range of video games available on the Atari console, concentrating analysis on three specific titles: *Space Invaders*, *Ms. Pacman*, and *Kung Fu Master*. These games were chosen because each one presents a different level of objective and rules while also increasing the complexity in the possible number of inputs an agent could choose, proving a suitable platform for evaluation

The primary objective of this report is to assess the performance of the mentioned algorithms in mastering these challenging gaming tasks. The aim is to gain insights on the adaptability and effectiveness of these algorithms.

2. Background

In this section, background on the algorithms used to train the agents is provided. Our primary focus remains on studying Bio-Inspired algorithms, for this the chosen algorithms were GA and NEAT, but also Deep Q-Network (DQN) is incorporated as a benchmark for comparison.

Genetic Algorithms (GAs) are optimization methods inspired by natural selection. They work by evolving a population of potential solutions through processes like selection, crossover, and mutation.

NEAT (NeuroEvolution of Augmenting Topologies) is an neural system algorithm that evolves both the structure and weights of neural networks. It starts with simple networks and evolves them by adding or removing connections and nodes.

Reinforcement Learning (RL), a machine learning paradigm, involves an agent acquiring the ability to make a decision by interacting with an environment and receiving feedback in the form of rewards or penalties.

In this work, DQN is employed, although the deep details are not covered in this study, it is worth noting that it learns to make decisions through interactions with its environment, receiving rewards and updating its values that help it anticipate future rewards for different actions in a given situation.

3. Implementation

In this section, implementation details of the earlier discussed algorithms are explored. Leveraging several existing Python packages throughout implementation. Specifically, for the Genetic Algorithm (GA), *deap* [2] was employed; for NeuroEvolution of Augmenting Topologies (NEAT), *inspyred* [3] is utilized; and for Deep Q-Network (DQN), opting for a quick implementation using *stable-baselines3* [4], developed by OpenAI [5].

For Atari games, Gymnasium [6] is used, a fork of the original OpenAI Gym, its a collection of environments intended for Reinforcement Learning (RL) that also includes an implementation of Atari environments based on [7]. Advantage of these environments is taken to apply them to our problem.

In the case of NEAT, Feed Forward networks were tested but a Recurrent Network architecture gave better results when training a video game environment due its memory capabilities for remembering past actions. Because the huge amount of information generated in each image, pre-processing is required. Initially, downsizing the image and retain only the gray-scale for faster training of the agents. Further enhancing performance was tried with image manipulation techniques using OpenCV [8] such as noise addition and blur to encourage the model to focus on essential features and smooth out irrelevant details, to help the model to focus on important features. Configuration files for each game are adopted to optimize fitness criteria, score and survival.

For GA, following a similar procedure, image preprocessing is conducted in a similar manner, the image is resized and works in gray-scale. A Convolutional Neural Network (CNN) architecture is constructed using Keras [9]. The CNN comprises two convolutional layers with ReLU activation functions, followed by a flattening layer and two fully connected layers. The model is trained using mean squared error loss and the Adam optimizer. To accelerate computation, operations are parallelized due to the resource-intensive nature of deap. Various population sizes, generations, and operator

configurations are explored.

For both GA and NEAT, a simple but effective fitness function is formulated to reward survival time alongside score and penalizing life loss.

$$F_{\text{fitness}} = R_{\text{score}} + R_{\text{survival}} - L_{\text{lost}} \quad (1)$$

Attempts to incentive certain actions over others (e.g., "Move and Fire" instead of only "Move" in Space Invaders) prove less effective, also tracking enemy fire could be potentially be a good reward but is not possible due to the limited information provided by the emulation in it's current state.

In both of the previous implementation there's no time limit or early stop counter, each step taken will wait till time runs out on the video game or it loses all its lifes.

For DQN training, stable-baselines3 is relied upon, a python package that facilitates a quick environment implementation. This framework allows us to specify the number of environments, aiding in parallelizing the training process but because of this, a large amount of computational resources are needed, to handle it, screen size is further downsized from the default to accommodate multiple environments. Additionally, stable-baselines3 already incorporates optimizations like Frame Skipping, enabling us to skip consecutive frames during gameplay, making training faster without compromising the agent's learning. Moreover, it supports similar pre-processing techniques utilized, such as resizing and grayscale conversion, ensuring that image data is in a suitable format for training. Several experiments are conducted with various timesteps.

4. Experiments & Results

In this section, explanations and analysis of experiment results commence with the NEAT algorithm, then move on to discuss the outcomes of the Genetic Algorithm (GA) implementation, followed by an examination of the Deep Q-Network (DQN). In each of them the performance of these algorithms is explored across the three mentioned game environments, aiming to provide insights into their effectiveness at task at hand. A general overview of the best fitness obtained on Table [1].

One notable observation is the distinct behavior displayed by the three algorithms during testing:

For Space Invaders, the GA took preference over the fire action and rarely moved being an easy target. In DQN, it would have a higher survivability by moving and firing. In case of NEAT it takes advantage also both actions and even being able to shoot the purple enemies that give a higher score.

In the case of Ms. Pacman, the GA stayed in a nearby area from the start where it was able to find the dots. In DQN, it tended to explore more the map and having a higher survival rate avoiding the ghosts. Meanwhile, NEAT once it found one of the energy dots it would stay in the same spot for the ghosts to reach it.

Finally, in Kung Fu Master, the GA occasionally executed a "punch" action, but otherwise the character remain static. In DQN it showed a variety of actions including "Move," "Punch," and "Kick." Meanwhile, NEAT consistently prioritized the "Kick" action, resulting in a significantly higher score depending on the kind of enemy it appeared.

Table 1: *Comparison of performance metrics (average scores) across different algorithms for various Atari games.*

	GA	NEAT	DQN
Space Invaders	165	230.46	270
Ms. Pacman	214.132	249.13	399
Kung Fu Master	88.80	103.51	121

4.1. NEAT

For each of the video games, a specific configuration file is created. Since each game may reward score points for different actions and also have a varying number of outputs from the network. Adjusting fitness thresholds accordingly, though our primary criterion remained the mean fitness. To maintain consistency across experiments, most settings were kept uniform. Different activation functions were explored including *tanh*, *relu*, and *sigmoid*. Through experimentation, it was found that utilizing 10 hidden nodes, limiting max stagnation to 5, and employing 2 elitism gave our best results. Due to hardware constraints, despite its seemingly low size, the population size was set to 100 individuals, evolved over 50 generations and resetting the population upon extinction. Running these simulations could be resource-intensive, sometimes taking several hours to complete. For full settings, refer to the configuration files available at [1].

The findings gathered are presented in Annex A. In both Ms. Pacman and Space Invaders, a single species overtook around the 20th generation, persisting and even showing continued improvement until their eventual extinction around the 30th generation. Surprisingly, after extinction, there was a slight enhancement in results for both games.

However, the scenario differed significantly in Kung Fu Master, the most complex tested game. Here, the performance was notably weaker, with little improvement in fitness averages and two instances of population extinction. This outcome suggests that the complexity of the game, coupled with its diverse range of inputs, would most likely would benefit with a longer training and fine-tuning of hyperparameters.

4.2. Genetic Algorithm

In the experiments with Genetic Algorithms (GA), a challenge was encountered with the initial fitness function. It only rewarded the agent based on the game score, leading to a tendency for agents to get stuck on specific actions without attempting to survive the game environment. For instance, in Space Invaders, agents learned to only use the "FIRE" action without focusing on survival tactics. To address this, the fitness function [1] was introduced, which significantly improved our results. This function was later incorporated into the NEAT implementation, that gave further enhancements.

After resolving this issue, a uniform configuration for all video games was adopted. Several experiments were carried with various configurations, ultimately selecting the one that yielded the best results. This involved modifying crossover, mutation, selection, and other hyperparameters. Specifically,

employing Gaussian mutation for weight perturbation, two-point crossover for genetic material exchange, and selection based on fitness values to retain the fittest individuals. These methods were chosen to effectively explore and exploit the search space inherent in image-based tasks.

Due to the resource-intensive nature of the deap library, even after parallelizing it, The population size was limited to 50 individuals and set the number of generations to 20. On average, training sessions took 3-4 hours to complete.

Despite deap longer training times compared to other methods, its performance did not consistently outperform them. However, the achieved fitness values were deemed satisfactory given our population and generations were much less.

4.3. DQN

In the case of DQN, as mentioned in previous section, tends to require also a lot of resources to train it in parallel with several environments, and besides the change of the screen resolution the settings were set to default to the ones of stable-baseline3 when creating the environment. Several timesteps were tested, and finally adjusted it for 300,000 time steps because it took about 25-50 minutes to train. While it would be considered a small amount of timesteps using a RL algorithm, it provided good results with this amount.

Consistent with our expectations and findings from previous experiments with NEAT and GA, DQN showcased superior performance in games like Space Invaders and Ms. Pacman. It also exhibited comparatively lower fitness scores in Kung Fu Master which could mean also that more timesteps could be required to learn more complex games.

5. Conclusions

In conclusion, our work offers a glimpse into the efficacy of Evolutionary Algorithms and Neural systems in tackling classic Atari games. While our initial tests provide promising results, it's clear that further experimentation with better hardware is needed.

As mentioned, a significant limitation of our work lies in the hardware constraints under which it was tested. The restricted computational resources imposed limitations on crucial aspects of our experiments, such as the population size and the number of generations explored. Increasing these parameters could potentially lead to improved algorithmic performance.

Also, because the same strategies were used for all the games, this approach is not the best fit for every game as each has its own challenges and optimal strategies. It would be beneficial to try out different strategies built to the specific challenges of each game individually. But for practical purposes and experimentation, it served well.

Additionally, when the fitness function was adjusted, there were improvements in performance for both Genetic Algorithms and NEAT. This shows the importance that a well defined fitness function can ultimately lead to more effective optimization and better outcomes.

Also in the result, several peaks on the fitness were noticed compared to the average one. This could have been caused due to the random nature of video games, for example, in Space Invaders, could have happen that more "purple ships" appeared and got hit or in Kung Fu Master, could have happen that more simple enemies appeared rather than complex ones. This shows us that finding the right way to measure success can greatly impact how well the algorithms perform. Therefore, exploring different ways to define success could lead to better results overall.

After all the conducted trials made, it's evident that each algorithm has its strengths and limitations. GA showed satisfactory results, especially after refining the fitness function, but longer training times and inconsistent performance were noted. NEAT exhibited competitive performance, particularly excelling in simpler games, yet struggled with more complex environments like Kung Fu Master, but as mentioned, more testing with different settings and increasing population and generations could achieve better solutions proving the capabilities of NEAT. On the other hand DQN demonstrated superior performance, leveraging deep learning techniques, but also faced challenges in adapting to complex dynamics. Ultimately, the choice of algorithm should be tailored to the specific game environment and computational resources available.

6. References

[1] "Bio-inspired project," https://github.com/DinosaurMauricio/biop, accessed: April 7th, 2024.

[2] "Deap," https://deap.readthedocs.io/en/master/, accessed: April 7th, 2024.

[3] "inspyred," https://pythonhosted.org/inspyred/, accessed: April 7th, 2024.

[4] "Stable-baselines3," https://stable-baselines3.readthedocs.io/en/master/, accessed: April 7th, 2024.

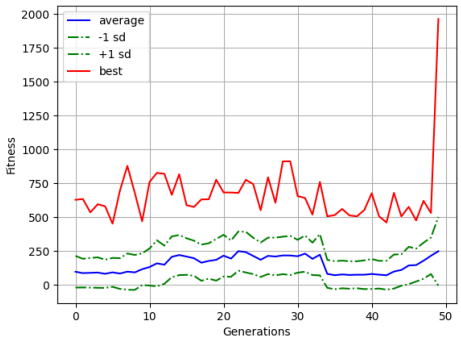
[5] "Openai," https://openai.com/, accessed: April 7th, 2024.

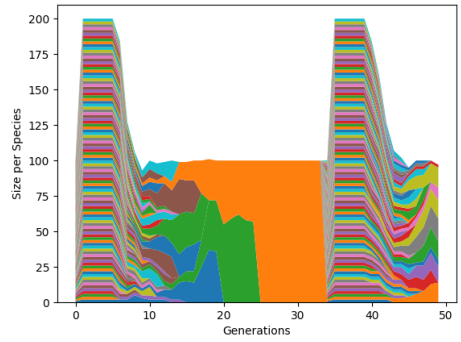
[6] "Gymnasium," https://gymnasium.farama.org/n, accessed: April 7th, 2024.

[7] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, jun 2013.

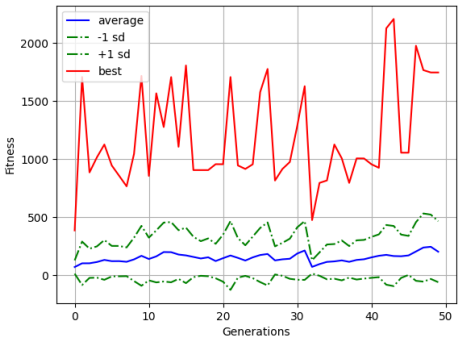
[8] "Opencv," https://opencv.org/, accessed: April 7th, 2024.

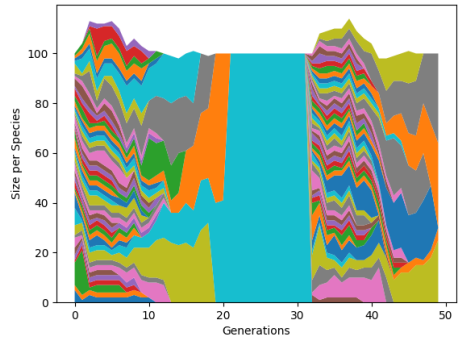
[9] "Keras," https://keras.io/, accessed: April 7th, 2024.



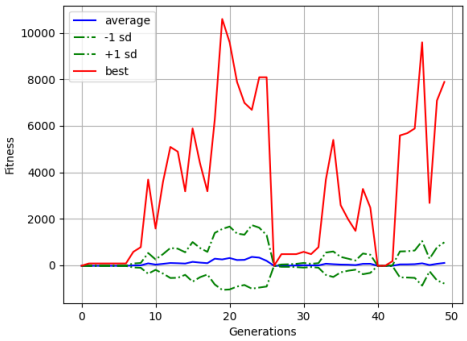


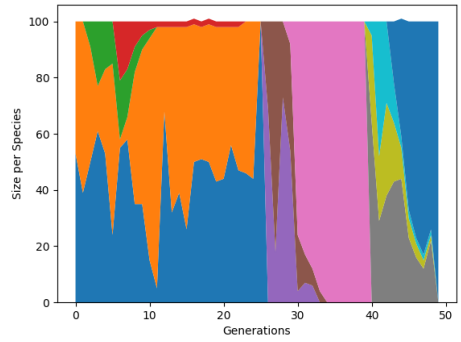
(a) ***Space Invaders***





(b) ***Ms. Pacman***





(c) ***Kung Fu Master***

Figure 1: *Average Fitness (top) and Speciation (bottom) of several tested games including* ***(a)Space Invaders***, ***(b) Ms. Pacman*** *and* ***(c) Kung Fu Master***