PLAYING ARCADE GAMES USING NEAT ALGORITHM

AGG IES DO

Presnetd by: Tyrell Allen
Mateo Smith
Deriech C
Dedrian Webb

Presented to Lorem Ipsum

North Carolina Agricultural and Technical State University

What is NEAT Algorithm?

- The NEAT(NeuroEvolution of Augmenting Topologies) Algorithm is an evolutionary algorithm that creates artificial neural networks.

- It combines GA and neural networks to optimize structure and weight on the neural network for specific tasks

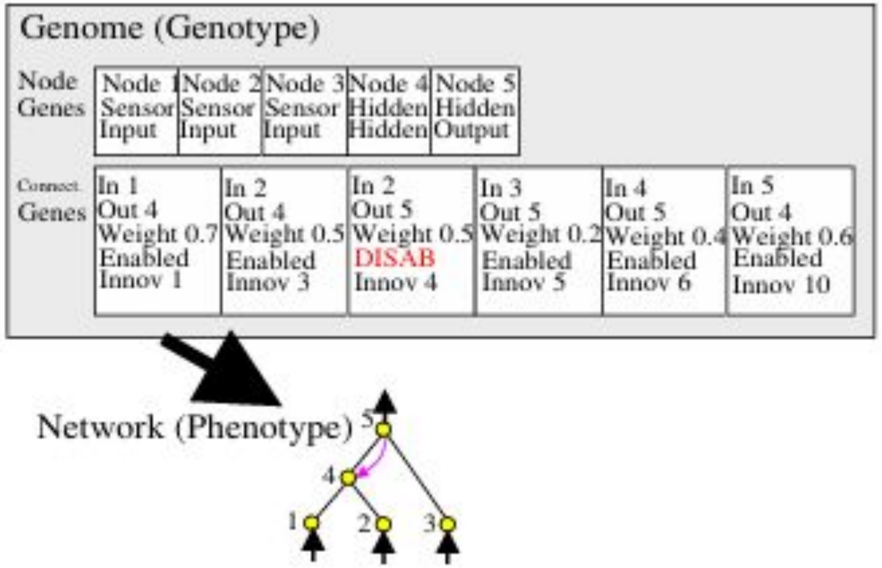
- NEAT algorithms create systems that are good at specific tasks

How does NEAT Algorithm work?

Genetic Encodings - Genomes have a list of genes that contain:

- In/out Nodes
- weight value
- enable bit
- innovation number

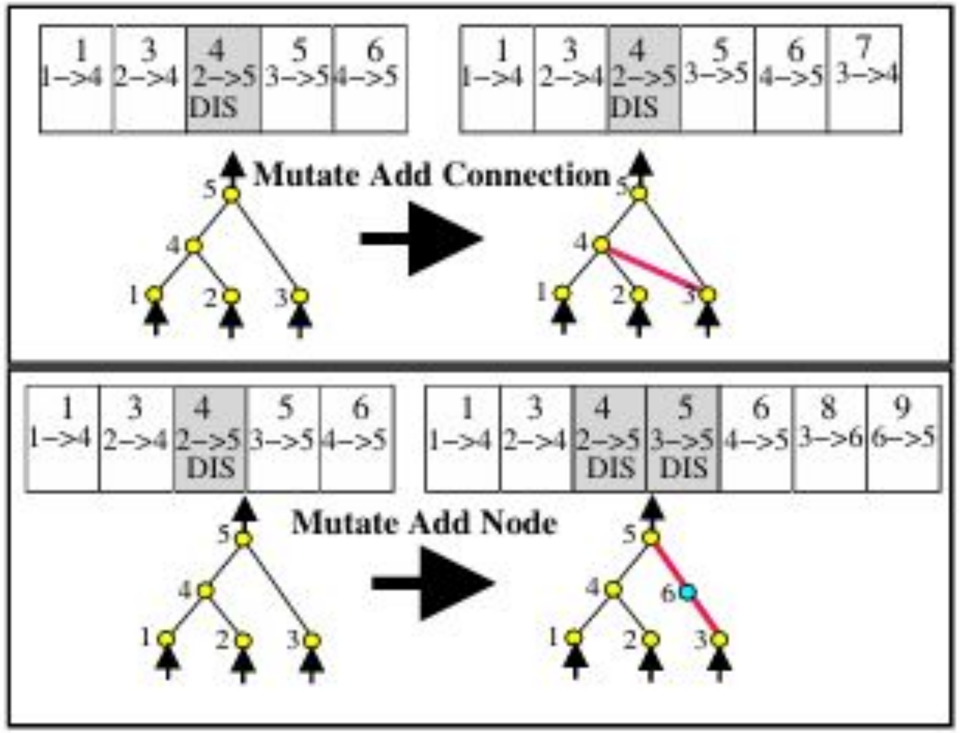
This list represents how a model is made



How does NEAT Algorithm work?

Historical Markings - A global innovation number allows for the tracking of structural mutations:

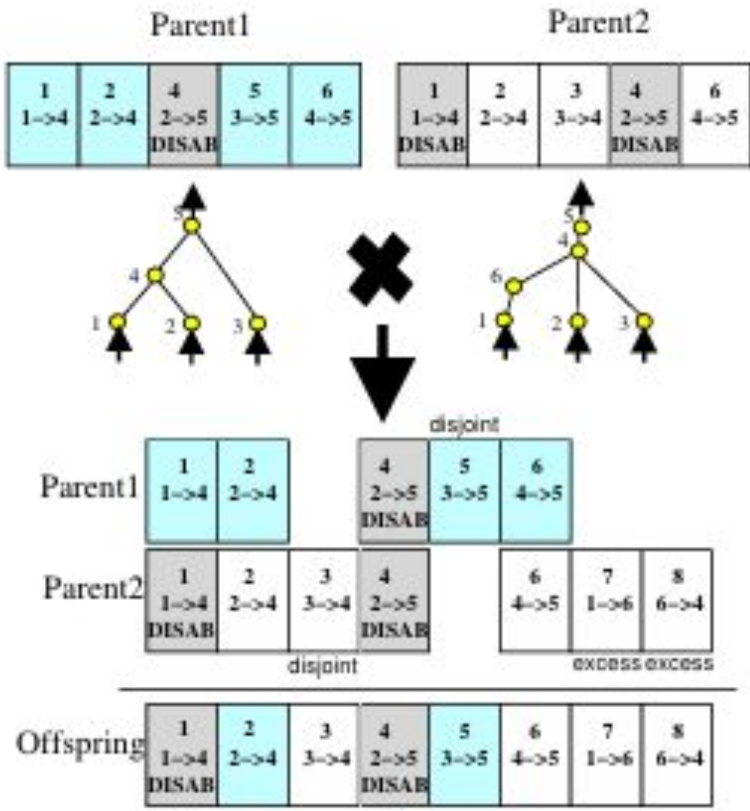
- Adding connections
- Adding Nodes



How does NEAT Algorithm work?

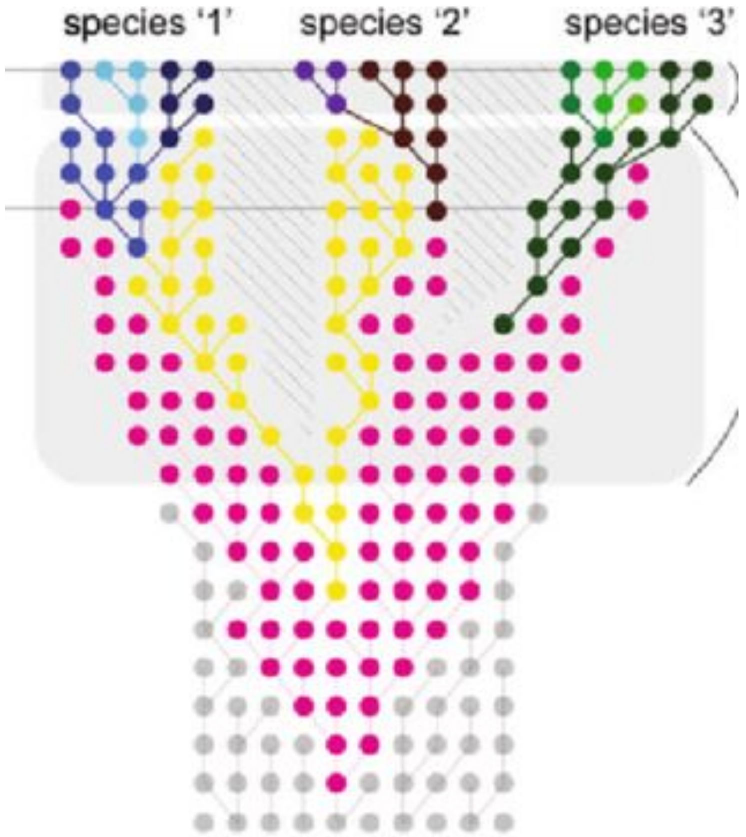
Crossover - Because of historical markings, crossover is very simple:

- Pick 2 parents
- offspring receives all genes similar to both parents
- offspring receives the access genes of the better parent
- if both parents are equal, the access genes received are random



How does NEAT Algorithm work?

- Speciation - Maintains innovation by allowing similar individuals to compete with each other, as opposed to competing with the entire population
- By competing with similar genes, you allow for certain innovations to optimize their strategies before having them compete with other strategies



Advantages & Limitations of NEAT Algorithm

Advantages

Adaptive Neural Topologies: NEAT allows neural networks to adapt their structures, enhancing their problem-solving capabilities.

Preserves Diversity: Speciation maintains diversity, ensuring exploration of innovative solutions.

Applicability: NEAT finds applications in diverse domains, from robotics to optimization problems.

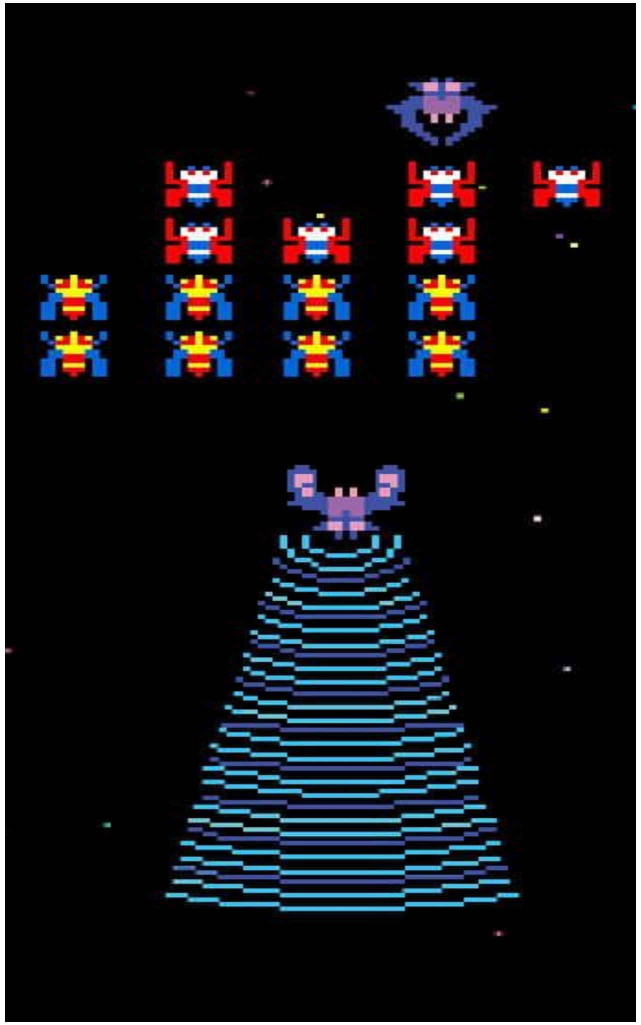
Limitations

Complexity: NEAT's implementation can be complex due to the management of genomes ar speciation.

Computationally Intensive: The evolution process may require substantial computation, especially for larger networks.

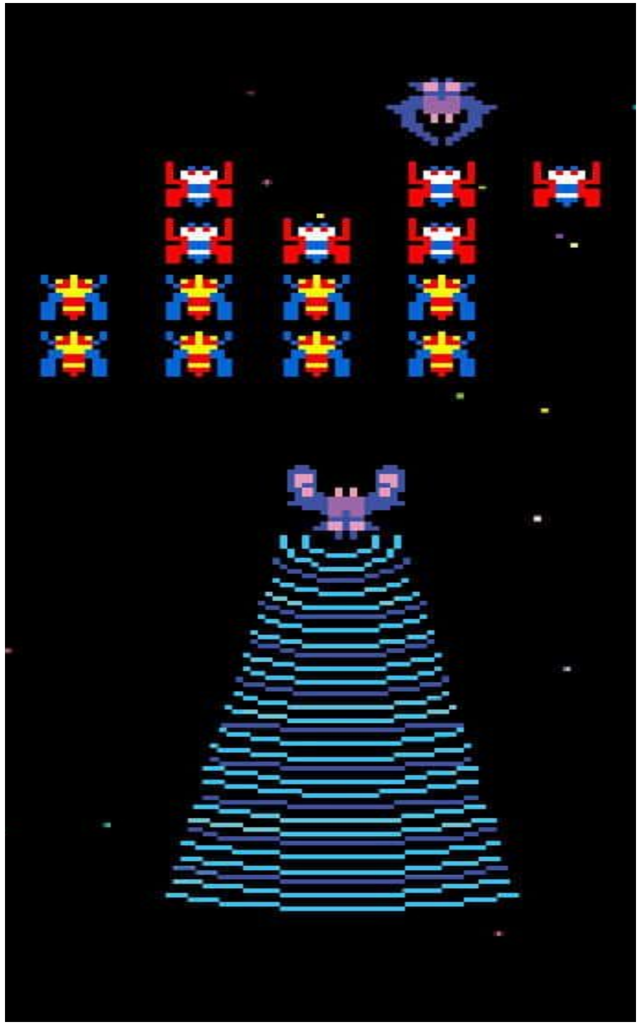
Arcade Game: Galaga

- Galaga is a game made in 1981 by Namco and Midway
  - You control a spaceship tasked with shooting down a squad of aliens
  - The aliens have a setup phase and an attack phase
    - In the setup phase, the aliens will fly onto the screen to where they will idle
    - In the attack phase, the aliens will fly down to attack you, then go back to their idle area

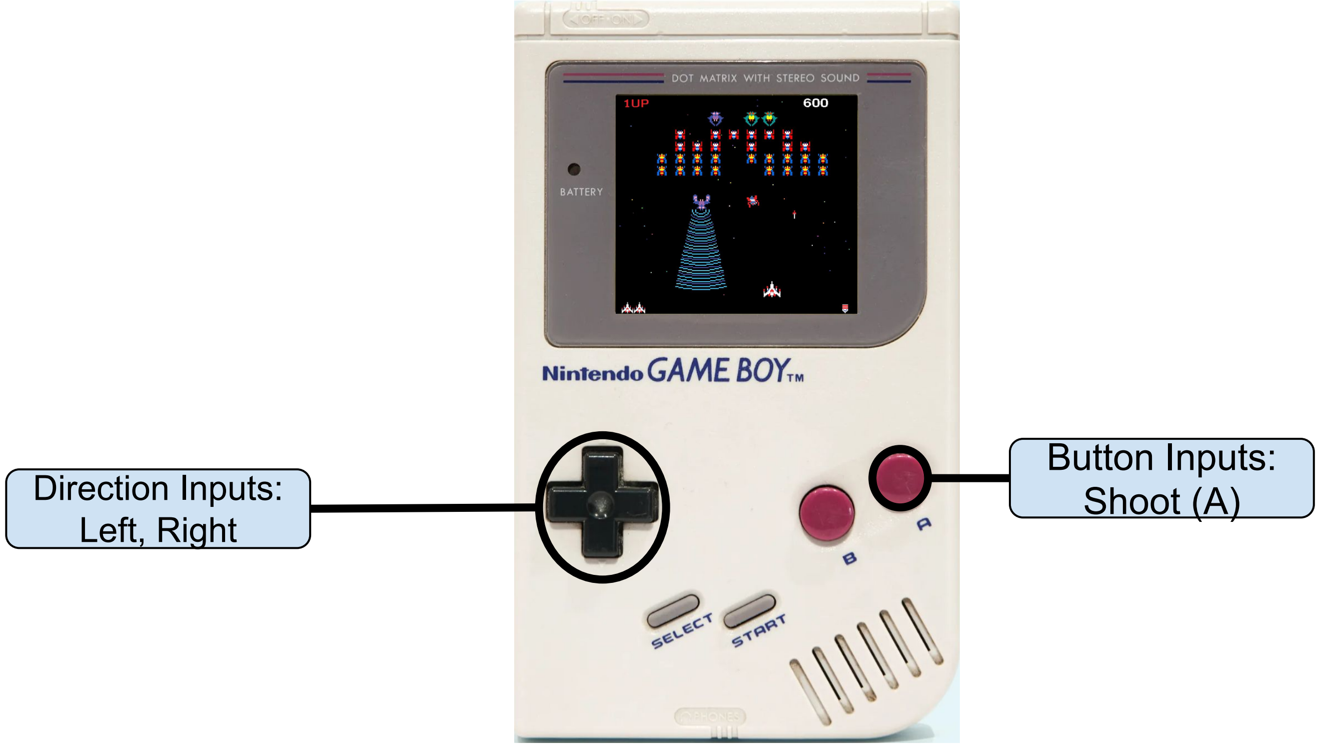


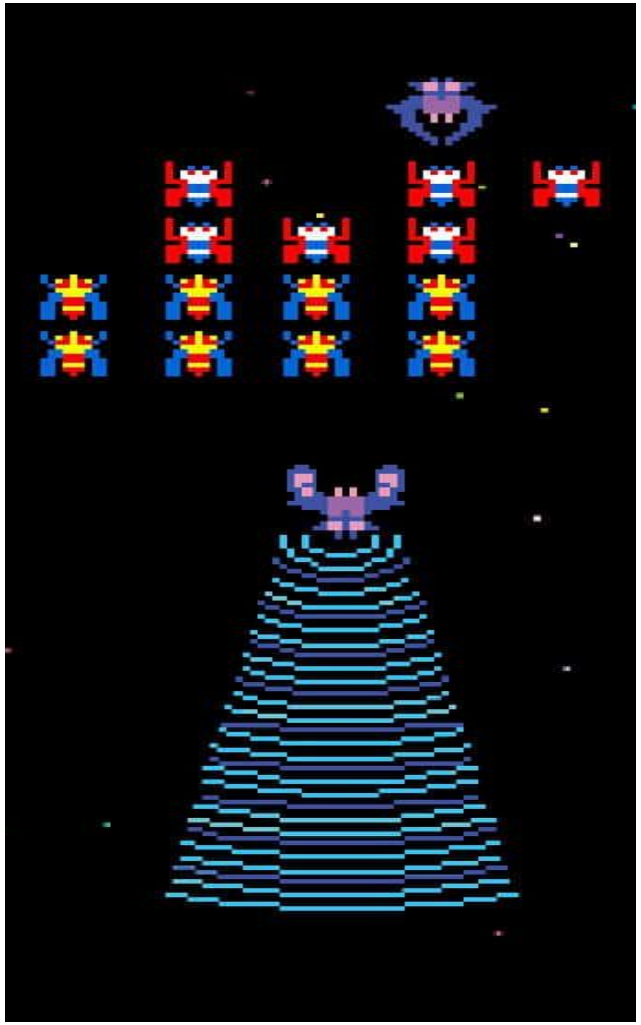
Arcade Game: Galaga

- You as the spaceship, are tasked with:
  - Shooting down enemy aliens
  - Moving out of the way of incoming aliens
  - Progress through as many of the levels as possible



Arcade Game: Galaga

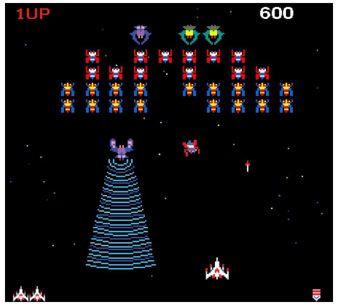


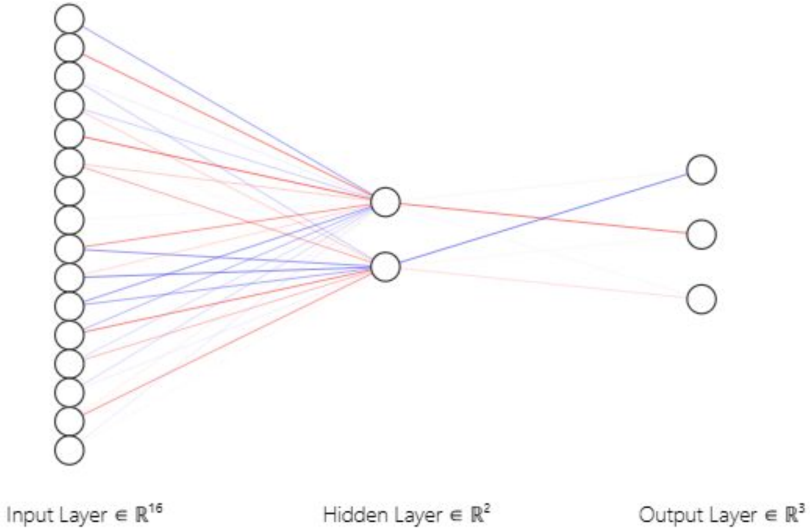


Because the only thing the user does during the game is move left, right, and shoot, these are the only buttons we'll map to our Neural Network

Neural Network Template

The Neural Network:

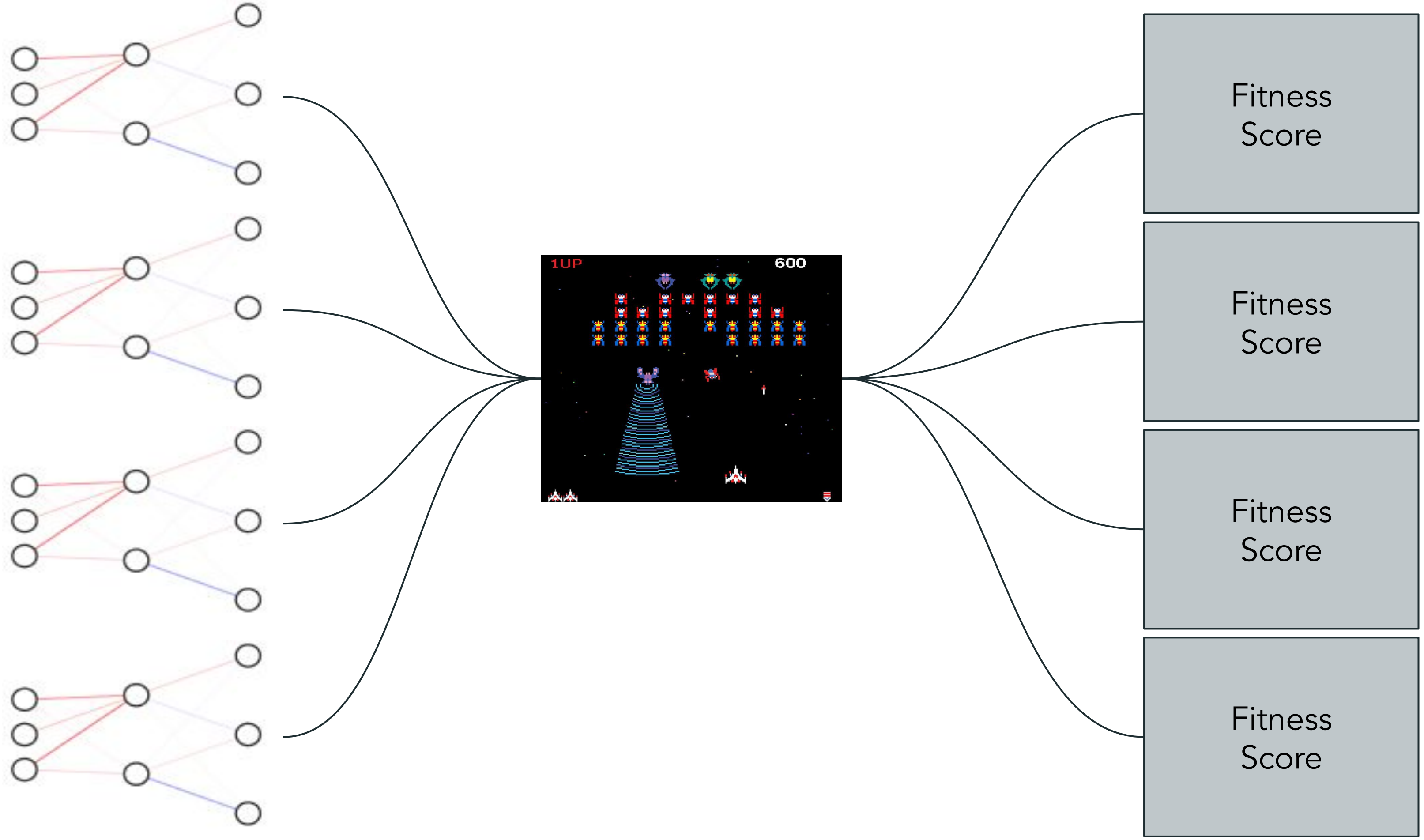




Takes in the game

…And outputs the necessary button inputs

Neural Network Template



FITNESS FUNCTION: INPUT DATA

- When a model dies, collect values from GameBoy Work RAM…

```
WRA0:CC70  26 09 00 00 00 00 00 00 06 10 01 08 01 00 00 00
WRA0:CC80  02 03 00 01 23 00 18 00 00 00 00 00 00 00 03 00
```

Offset	Name	Size (Bytes)	Endianness
0xCC70	Time Elapsed	2	Big
0xCC7A	Score as BCD	5	Little
0xCC80	Lives Left	1	Little
0xCC84	Shots Fired	2	Little
0xCC86	Enemy Kill Count	2	Little

FITNESS FUNCTION: COMPUTATION

..and calculate the model's fitness.

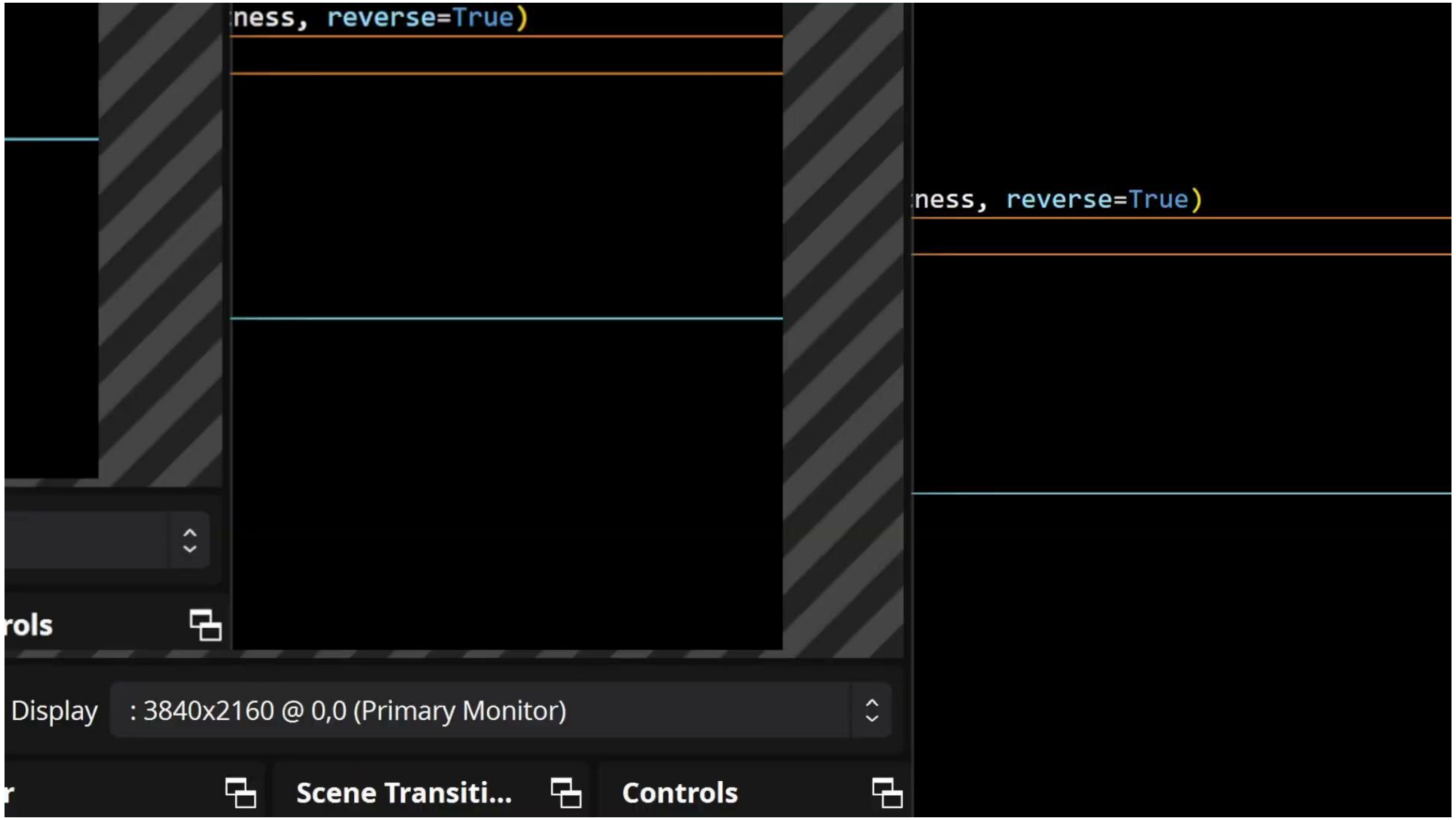
Normalize all 3 function variables for weighting

Current weights may be subject to change based on results

Weighting allows us to determine the best "recipe" to best generate the best model

```python
# ceilings for normalization
MAX_SCORE = 99999
MAX_TIME= 254
MAX_ACCURACY = 100

# weightings
SCORE_WEIGHT = .3
TIME_WEIGHT  = .45
ACCURACY_WEIGHT = .25

def calc_fitness(score,time,shots,hits):

    accuracy = hits/shots

    scor_weighted = (score/MAX_SCORE)*SCORE_WEIGHT
    time_weighted = (1-(time/MAX_TIME))*TIME_WEIGHT
    accu_weighted = (accuracy/MAX_ACCURACY)*ACCURACY_WEIGHT

    fitness = scor_weighted + time_weighted + accu_weighted # max fitness = 1

    return fitness
```

Galaga Demo


ness, reverse=True)
ness, reverse=True)
rols
Display
: 3840x2160 @ 0,0 (Primary Monitor)
Scene Transiti...
Controls

Game 2: Super Mario Land

- Basic 2D platformer - go left to right
- Fitness function with primary basis on left->right movement
- Population size = 400
- Simpler for NEAT than Galaga - better performance?

Super Mario Land: Fitness Function

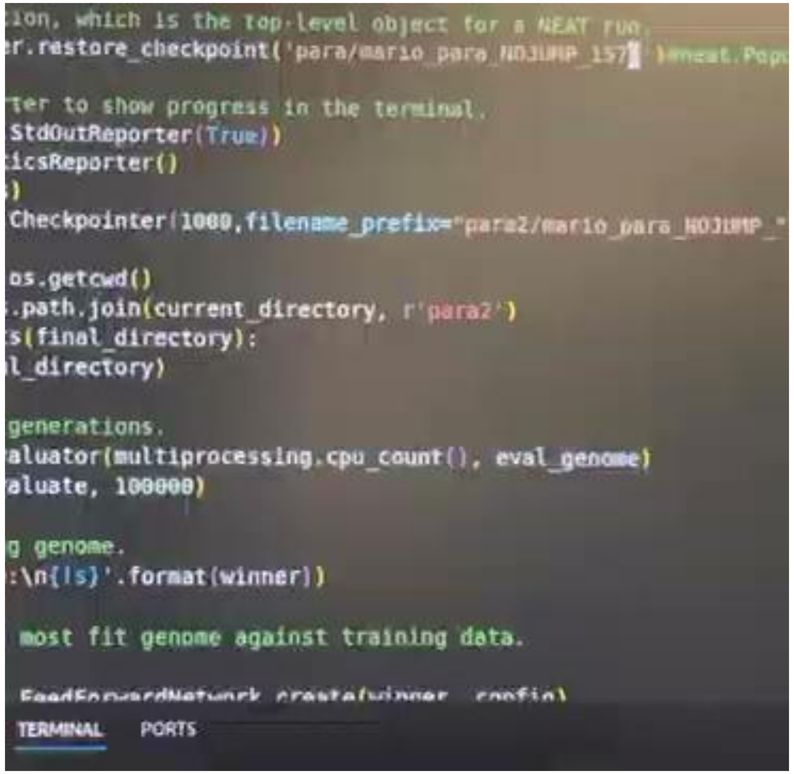
- Measure: left->right distance
  - incremented by boolean multiplication
    - (speed > 0) * (next_tile != last)

Offset	Name	Size (Bytes)
0xC001	Next Tile	1
0xC208	Y Speed	1
0xC20C	X Speed	1
0xC20A	On Ground?	1

```
def calc_fitness(dist):
    fitness = dist
    return fitness
```

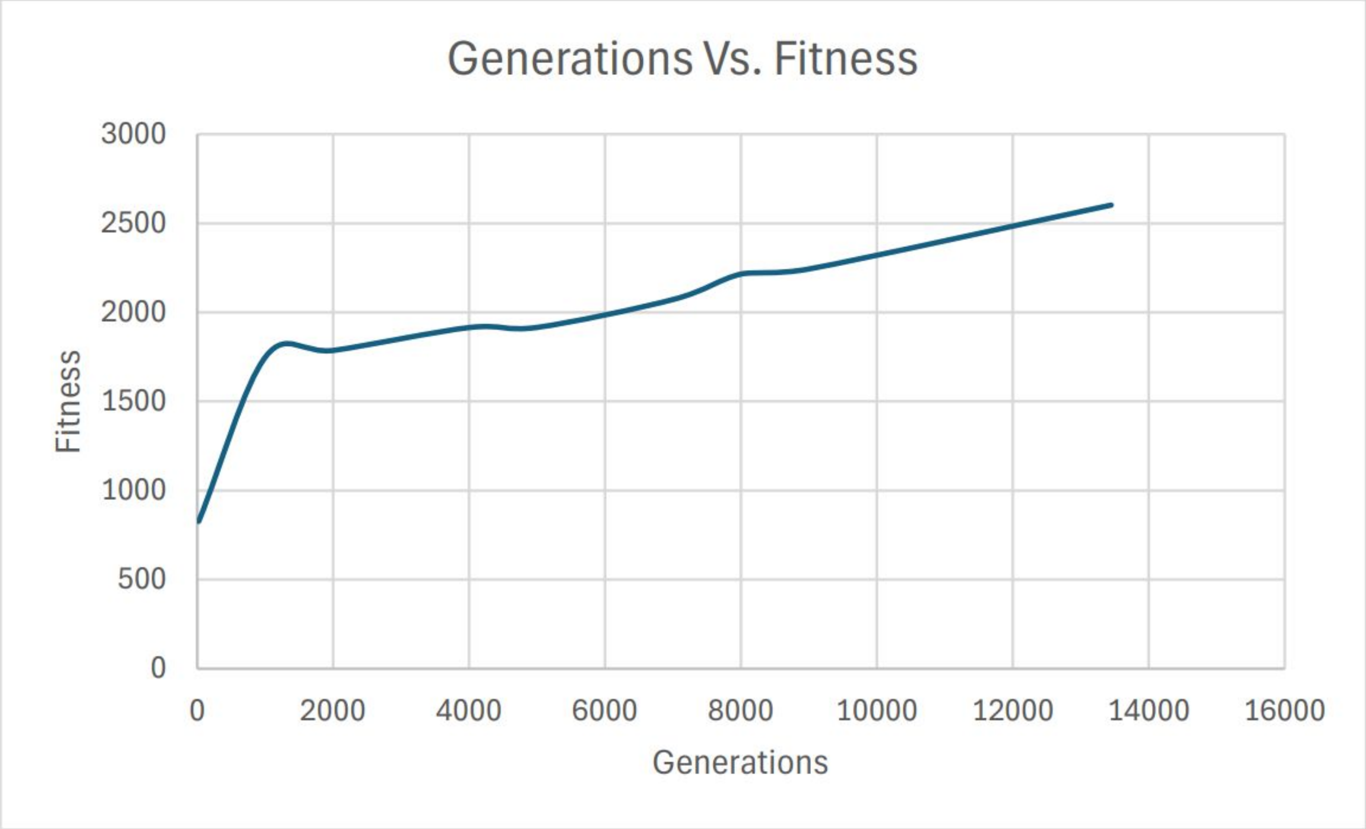
Super Mario Land Evolution: Gen. 21

Super Mario Land Evolution: Gen. 1540



Super Mario Land Evolution: Gen. 13446

Super Mario Land Evolution: Trend



CONCLUSION

We have been able to:

- Emulate Galaga and Super Mario Land using Python
- Use a Neural Network Model to play Galaga + SML
- Give Model a score based on various criteria
- Use Genetic Algorithms to create the best model for playing Galaga + SML
- Finally compare and contrast optimal model design and results from both games

Tools we used:

- Pyboy (emulates the game)

Timeline

Initial Galaga Example	4/3/24
Initial Mario Example	4/22/24
Minimum Viable Product Completion	5/2/24
Full Completion	5/6/24

Sources

- https://bgb.bircd.org/manual.html
- https://datacrystal.tcrf.net/wiki/Super_Mario_Land/RAM_map
- Stanley, Kenneth O, & Miikkulainen, R. (2002). Efficient evolution of neural network topologies. Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600). https://doi.org/10.1109/cec.2002.1004508
- Goldberg, D. E. (2013). Genetic algorithms in search, optimization, and machine learning. Pearson.
- SethBling. (2015, June 13). Mari/O - Machine Learning for Video games. YouTube. https://www.youtube.com/watch?v=qv6UVOQ0F44
- PyBoy Github: Baekalfen/PyBoy: Game Boy emulator written in Python (github.com)
- McIntyre, A., Kallada, M., Miguel, C. G., Feher de Silva, C., & Netto, M. L. neat-python [Computer software] CodeReclaimers/neat-python: Python implementation of the NEAT neuroevolution algorithm (github.com)

THANK YOU

Any Questions?