IPEC Journal of Science & Technology, Vol. 02 (01), June 2023

ISSN: 2583-3286(Online)

Car Racing Game AI Using Reinforcement Learning

Monika Sharma, Tarun Singhal, Taufiq Umar, Uday Raj Sharma, Vinay Kumar Maurya

Department of Computer Science and Engineering,
Inderprastha Engineering College, Ghaziabad, Uttar Pradesh, India

***Abstract*: Car Racing Game AI using Reinforcement Learning is an implementation of a neuroevolution algorithm for training autonomous vehicles to drive in a 2D environment. Here, we evolve artificial neural networks, which are responsible for controlling the vehicle's movements, with the help of genetic algorithms. The project provides a flexible framework for testing different neuroevolution strategies and can be extended to include more complex environments or additional sensors. The results ensured that this approach can be useful in real-life automation driving. As we know in today's world, cars, and other on-road vehicles play a vital role. There would be billions of vehicles on the road right now. So much traffic is also responsible for a large number of deaths and injuries due to accidents. These mostly happen due to human errors like driver negligence, not following traffic rules, etc. In such a case, an autonomous driving car based on AI can be a viable solution.**

***Keywords* - Neuroevolution (NE), Reinforcement learning (RL), Genetic Algorithm (GA), Feedforward Neural Networks (FNN), Elitist selection**

I. INTRODUCTION

RL is a subfield of ML that focuses on training agents to make decisions in an environment, with the goal of maximizing some cumulative reward. It is often used in settings where the optimal decision-making strategy is not known a priori and must be learned through trial and error.
In RL, an agent interacts with the environment by taking action and receiving feedback in the form of rewards or penalties. The agent learns a policy, which maps states to actions and helps in maximizing the expected cumulative reward over time.
Genetic algorithms and neural networks are two popular machine-learning techniques that have shown great success in a wide range of applications. Genetic algorithms can be used to optimize complex solutions by simulating the natural selection process, while neural networks can learn patterns in data and make predictions based on this learning.
In recent years, researchers have been exploring the combination of genetic algorithms and neural networks to create more powerful learning systems. This combination allows for the optimization of the neural network's structure and parameters using genetic algorithms.

Date of Submission: 17 May 2023
Date of Acceptance: 20 June 2023
Corresponding Author: Tarun Singhal
(Email: 1900300100232@ipec.org.in**)**

One such application is in the field of artificial intelligence, where agents with neural networks can learn to perform tasks in a simulated environment.
This research paper presents a class called Agent, which combines a genotype and a feedforward neural network (FNN) using C#. The class takes in a genotype and constructs an FNN based on its parameters. The FNN can then be used to control the behavior of the agent in a simulation.
This paper will demonstrate the effectiveness of the Agent class by implementing and testing a genetic algorithm that uses the class to evolve agents to perform a specific task in a simulated environment.

In addition to the above, this paper also explores the implementation of elitist selection, a widely used technique in evolutionary algorithms. Elitist selection ensures that the best-performing individuals of a population are always preserved, even if their fitness is not improved upon in subsequent generations. This helps to prevent the loss of highly fit solutions and can significantly improve the overall performance of the algorithm.

ISSN: 2583-3286(Online)

II. LITERATURE REVIEW

A lot of research has been done in the field of the development of autonomous car-driving AI:

In 2018, Chaudhari, Raj, et al. [1] used an inception V3 model network that uses CNN to train an AI car to drive autonomously on roads without any human intervention. They have used GTA 5, an open-world game to extract data as frames and simulate the final output. The frames are then processed and fed to the network for training.

In 2021, Gupta, Abhishek, et al. [2] surveyed the theory underlying self-driving vehicles and deep learning perspectives along with their recent implementations.

In 2021, Song, Yunlong, et al. [3] used curriculum RL to train high-speed autonomous racecars to overtake other cars in Gran Turismo Sport. They presented the advantages of curriculum RL over standard RL.

In 2018, Maresso, Brian in [4] "Emergent behavior in neuro evolved agents." presented a simple, generalized process by which neural networks can be trained to control video games or computer simulations. Using techniques from related works in a variety of applications of neural networks and evolutionary algorithms, they synthesized human input from neural networks by way of a simulated vehicle in a testbed racing game. This was done without any training data or hard-coded behaviors.

In 2020, Garnica, Carlos Andrés, Juan Carlos Barrero, and Miryam Liliana Chaves in [5], used two algorithms, namely FNN-GA and Q-learning, implemented in Unity 2D. The result depended on the track designs, evaluation function, and reward system.

In 2021, Fuchs, Florian, et al. [6] presented an autonomous car racing policy that achieves high performance in time trial settings in Gran Turismo Sport simulator. It did not rely on human intervention or explicit path planning.

In 2018, Fayjie, Abdur R., et al. [7] presented Deep Reinforcement Learning autonomous navigation and obstacle avoidance of self-driving cars, applied with Deep Q Network to a simulated car in an urban environment. The approach uses a camera sensor and laser sensor for input.

In 2018, Wang, Sen, Daoyuan Jia, and Xinshuo Weng in [8], used the deep deterministic policy gradient (DDPG) algorithm, which has the capacity to handle complex states and action spaces in a continuous domain. They also used The Open Racing Car Simulator (TORCS) for simulation.

In 2017, Sallab, Ahmad EL, et al. [9] provided a survey of the recent advances in the field of Deep RL and used a framework for a Deep RL pipeline for autonomous driving.

In 2015, Risi, Sebastian, and Julian Togelius in [10] surveyed research on applying neuroevolution (NE) to games. In neuroevolution, ANN is trained through genetic algorithms, taking inspiration from how biological brains evolved.

III. METHODOLOGY

The project is based on an AI car bot traversing through the race course without hitting walls or hurdles in the path. The AI is RL based.

The project is implemented in C# language using Unity editor.

- Feed-forward Neural Network is used for encoding the car bot.
- The user can input the topology for the neural network through the user interface in the unity editor.
- Furthermore, a genetic algorithm is applied to the neural network, through which the AI is able to improve itself generation by generation.
- Elitist selection is used in the genetic algorithm, according to which the best two genotypes are preserved without being mutated so that the best solution isn't lost.

3.1 Reinforcement Learning

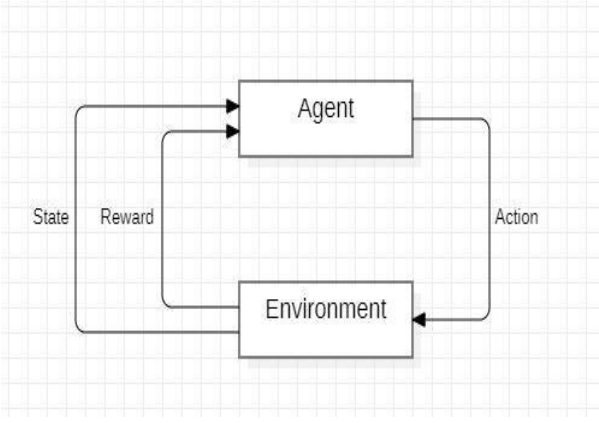

Figure 1. Reinforcement Learning Diagram

ISSN: 2583-3286(Online)

Reinforcement learning (RL) is a type of machine learning that is inspired by the way humans and animals learn from their environment. In RL, an agent interacts

with an environment in a trial-and-error fashion to learn the optimal policy for performing a given task. The agent receives feedback in the form of rewards or punishments for each action it takes in the environment. It then uses this feedback to learn which actions lead to desirable outcomes and which do not. The goal of RL is to maximize the cumulative reward over time by finding the optimal policy. A policy can be defined as the sequence of actions that results in the highest long-term reward. RL has been successfully applied to a variety of domains, including robotics, gaming, finance, and healthcare.

3.2 Feed-Forward Neural Network

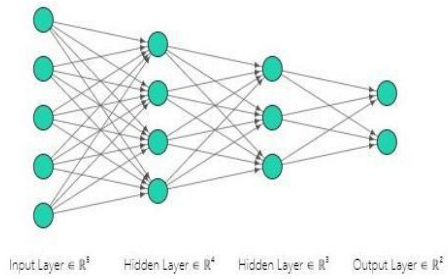

Figure 2. Feedforward Neural Networks Diagram

A feedforward neural network is a type of artificial neural network that uses a directed graph to model the flow of information through the network. It consists of an input layer, zero or more hidden layers, and an output layer. The neurons in each layer are connected to the neurons in the next layer.

In an FNN, information flows from the input layer through the hidden layers to the output layer. The input layer receives input from the environment (the five sensors in this project). Each neuron in any given layer of the network receives input from the neurons of the previous layer. It then applies a mathematical function to the input received and produces an output that is sent to the neurons in the next layer. The output layer represents the values used by the model to perform some action (two neurons in the output layer in this project, one for throttle and one for steering).Each connection between the neurons has some weight. A weight can be defined as some numeric value that can be used to change the strength of the connection between any two neurons. The weights are used to control the influence a neuron can have on the neuron of the next level. The weights and biases of the neurons are adjusted during the training process to optimize the performance of the network.

3.3 Genetic Algorithm

A Genetic Algorithm (GA) is a type of optimization algorithm inspired by the biological process of natural selection. The algorithm starts with a population of potential solutions represented as chromosomes and evolves the population by applying operators such as selection, crossover, and mutation, mimicking the natural process of evolution. Over time, the genetic algorithm generates a new population of better and better solutions over time. GAs have been successfully used in various fields, such as optimization, machine learning, and computer science.

3.4 Elitist selection

In genetic algorithms, the selection is the process of choosing individuals to pass on their genes to the next generation. Elitist selection is a type of selection method where the best-performing individuals from the current generation are guaranteed to be selected for the next generation without any changes to their genes.

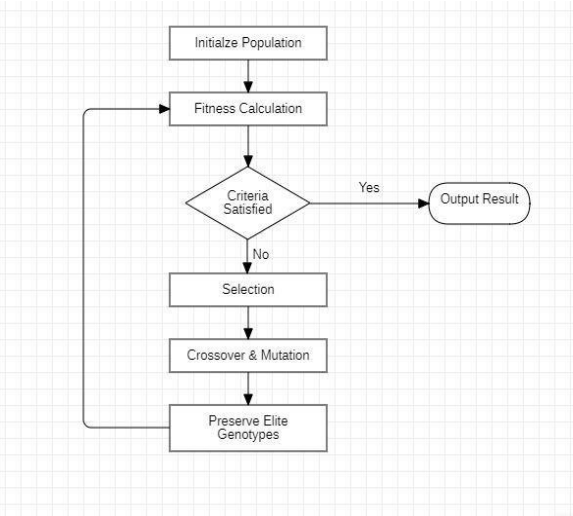

Figure 3. Genetic algorithm with elitist selection flowchart

In other words, elitist selection ensures that the fittest individuals are always carried over to the next generation, regardless of the other selection methods being used. This helps to maintain a high level of fitness in the population and prevents the loss of potentially useful genetic information.

ISSN: 2583-3286(Online)

FNN and GA were combined and elitist selection was used.

1. Population of 20 is initialized with random weights in the neural network.
2. Input is taken in each frame through sensors.
3. Output is calculated through neural networks.
4. Two best-performing genotypes are preserved according to their fitness score.
5. Remaining genotypes are mutated for the next generation.

IV. DATASET AND MODEL DESCRIPTION

Since Feed-Forward Neural Network is being used in combination with a genetic algorithm, we don't require labeled datasets for training the AI model. This is because the model will be able to improve upon itself generation by generation. Although, we would provide the AI with the appropriate environment to train itself. For this, we have built four different race tracks with increasing difficulties. These tracks include various turns and hurdles. The car bot has to move at high speed through the checkpoints of the track avoiding hurdles and walls along the course.

V. RESULTS AND DISCUSSIONS

Feedforward Neural Networks were used for training the AI car bot. FNN was further modified with the Genetic Algorithm optimization technique. A reward system was made and the genotypes as well as the whole population were calculated on the basis of which fitness of the individual genotypes were calculated. The project was implemented in Unity 2D, and C# language was used for scripting.

The simulation was run on track-1 of easy difficulty and each generation was evaluated. The table below provides the generation with their evaluation score having a score of more than 0.2.

Table I
Values used in FNN-GA

Parameter	Value
Probability of a parameter being swapped during the crossover	0.6
Probability of a parameter being mutated	0.3
The amount by which parameters may be mutated	2.0
Percent of genotypes in a new population that are mutated	1.0

Table 2
Generation vs Evaluation data gathered through simulation

Generation	Evaluation
25	0.6179952
35	0.3276338
41	0.3860919
44	0.2594249
47	0.2594249
72	0.6179952
82	0.3860919
91	0.7146537
98	0.7146537
112	0.3639981
131	0.2594249

ISSN: 2583-3286(Online)

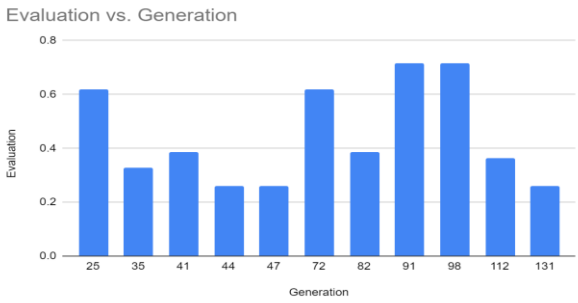

Figure 4. Generation vs Evaluation Bar Chart

The car bot was able to reach the end of the track in generation-91 and generation-98. With this result, we can say that the car AI was able to traverse the course without hitting the walls in a fairly less time period.

VI. CONCLUSION

Feedforward Neural Network combined with Genetic Algorithm was implemented to train AI car bots. The elitist selection was used to select and preserve the best solutions in each generation. This was implemented in Unity 2D and the scripting language used was C#. The results depend upon various factors such as the topology of the network, the difficulty of the track course, the population size, etc. Furthermore, the model didn't need any external data, but only the data generated by the simulated environment.

The car ai was able to adjust the weights to optimal values which can help the car bot to traverse through the race track without hitting any wall or hurdle in the shortest possible time.

This approach can be used to make highly efficient self-driving car bots, but it has some serious drawbacks -

- It highly depends on the hit and trial method where the weights are set randomly at the starting and modified till the optimal weight combination is found.
- It has to be trained for each track separately.
- It cannot handle dynamic environments with high randomness in them.
- The car model has to be trained for many generations which can vary from a few hundred generations to a few thousand generations.

For future study, this model can be further developed to-

- Run simulations in dynamic environments, such as moving hurdles or random hurdles, and real-time traffic scenarios.
- Some deterministic models can be applied that can guide the bot to perform certain fixed actions in certain scenarios. For instance, it can guide the model to how to overtake another car on the race track.
- This model can be trained to drive in any environment rather than in a specific environment.
- Run 3D car bots on a similar model.
- Implement the model on a real-world car, but only in a fully controlled environment.

REFERENCES

[1] Chaudhari, Raj, et al. "Autonomous driving car using convolutional neural networks." 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). IEEE, 2018.

[2] Gupta, Abhishek, et al. "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues." Array 10 (2021): 100057.

[3] Song, Yunlong, et al. "Autonomous overtaking in gran turismo sport using curriculum reinforcement learning." 2021 IEEE international conference on robotics and automation (ICRA). IEEE, 2021.

[4] Maresso, Brian. Emergent behavior in neuroevolved agents. Diss. University of Wisconsin--Whitewater, 2018.

[5] Garnica, Carlos Andrés, Juan Carlos Barrero, and Miryam Liliana Chaves. "Autonomous virtual vehicles with FNN-GA and Q-learning in a video game environment." 2020 Congreso Internacional de Innovación y Tendencias en Ingeniería (CONIITI). IEEE, 2020.

[6] Fuchs, Florian, et al. "Super-human performance in gran turismo sport using deep reinforcement learning." IEEE Robotics and Automation Letters 6.3 (2021): 4257-4264.

[7] Fayjie, Abdur R., et al. "Driverless car: Autonomous driving using deep reinforcement learning in urban environment." 2018 15th international conference on ubiquitous robots (ur). IEEE, 2018.

[8] Wang, Sen, Daoyuan Jia, and Xinshuo Weng. "Deep reinforcement learning for autonomous driving." arXiv preprint arXiv:1811.11329 (2018).

[9] Sallab, Ahmad EL, et al. "Deep reinforcement learning framework for autonomous driving." arXiv preprint arXiv:1704.02532 (2017).

[10] Risi, Sebastian, and Julian Togelius. "Neuroevolution in games: State of the art and open challenges." IEEE Transactions on Computational Intelligence and AI in Games 9.1 (2015): 25-41.

[11] Markowska-Kaczmar, U., & Koldowski, M. (2014). Spiking neural network vs multilayer perceptron: who is the winner in the racing car computer game. Soft

[12] Computing, 19(12), 3465–3478. doi:10.1007/s00500-014-1515-2

[13] Hausknecht, Matthew, et al. "A neuroevolution approach to general atari game playing." IEEE Transactions on Computational Intelligence and AI in Games 6.4 (2014): 355-366.

[14] Ambuehl, Nathan. "Investigating Genetic Algorithm Optimization Techniques in Video Games." (2017).