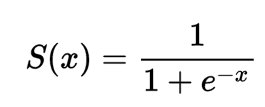
Modular Action AI is a system developed to give creators a simple neural network that will teach an agent depending on the users’ required inputs and outputs.

In this specific example, this system will be developed to teach AI how to move around a level given specific inputs. The goal of this system is to generate a neural network that will reduce the amount of time when building an AI by teaching one instead. This will be done by machine learning.

There are many third-party libraries that have created neural networks such as Keras and TensorFlow but for the purpose of the Task at hand, these will not be used. A custom neural network will need to be built. Some of Unity’s functions will be required such as Mono behaviours and their inbuilt math library.

For writing a neural network, several mathematical operations are to be used. This includes a threshold function or activation function that will determine whether an output is required. We also need to summate all the input values that are given to the hidden nodes but that is not as complex. The activation threshold of a specific neuron is decided by specific functions. There are many functions that can be used such as a sigmoid function, linear function or sigmoid function.



These functions determine the activation value of a neuron and is an important part of the network. Sigmoid functions are used when the output value is expected to be in between 0 and 1 where large numbers clamp to 1 and large negative numbers will clamp to 0 as per the graph. Tanh functions are used when the activation value needs to be in between -1 and 1. The linear functions consist of a flat line where if the input value into the function is below a certain amount, the output will be set to 0. Only after the input has reached a certain point, the output value will increase linearly.

A fitness variable will be needed to determine the better functioning genes to select from. This will be another function that will be determined by what is more valued. In this case, the progress the agent has made in respect to the end and how fast it can get there.

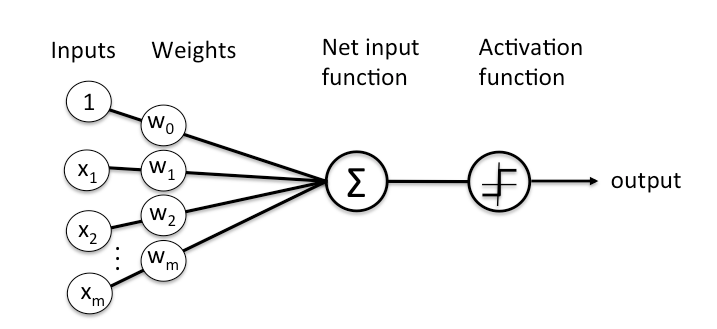
After this crossover has been made between the two of the best networks, there is a small chance of mutation. This will change some weights of the network to influence some change in the overall networks. The whole system is then populated with more networks with the newly updated weights and this process repeats until the network converges.

The advanced algorithm that will be implemented is the neural network system using a single perceptron layer. This is a complex AI that uses machine learning to develop a mind that will react to situations using a set of actions based on given inputs. This system will teach itself using given sets of data. A neural network consists of the input layer, output layer and hidden layers. Ideally, multi-threading will be used to accelerate the speed at which the neural network teaches itself. These layers will have weights and biases associated with each neuron and this impacts the direction we want the AI to develop.

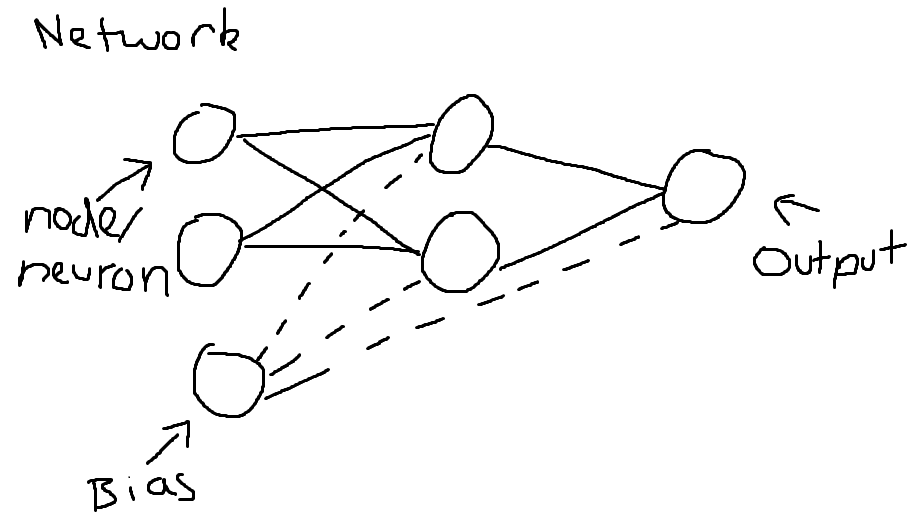
Graphical user interface, text, application

Description automatically generated

The above image is some pseudocode for a genetic algorithm. This will be useful when combining this with a neural network to get a more functional network that will learn faster and better.



The above image is a representation of a node in the network. This is the core functionality of it and will allow the network to “think” based on the input values we give. The network will compute an output and based on that output make decisions.



The above image is what a small network can look like. This network has 2 inputs in the input layer, a hidden layer with 2 nodes, and one output. A bias node is necessary in all networks as this sets a bias value for every node other than the input to ensure the system is being taught correctly.

The AI system will be easily manipulated by the user. The package will include a script that the user can drag onto objects they want to have a network learn for. The user will pick how many inputs and outputs there are. The user, however, must create the actions that are more suited to there use case. This may include functionality that is not in the base script. The base script will only have examples of the actions that are required to be made. This will be based off the car driving. The base module will only have 4 outputs for forward, backwards, left and right. If the user wants to add more outputs, they also need to provide the functionality for it themselves. The user will not have access to the hidden layers. This will give the user a large influence on how the network learns by predetermining which inputs the outputs will rely on. The weights and connections will then be saved and can be accessed during runtime.

The goal is to have the user be able to set the inputs and outputs of the neural network. For now, it will be a single layer perceptron that will only include one hidden layer.

To make this modular, many of the variables must be freely set in the inspector. To do this, the number of inputs and outputs and what they are will be freely set by the user. This makes it applicable to most use cases. This module is more tailored towards programmer as the actions must be created by the user. This module only includes the neural network.

Deepanshi (2021) *Beginners Guide to Artificial Neural Network*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2021/05/beginners-guide-to-artificial-neural-network/ (Accessed: May 2, 2023).

Geeks For Geeks (2023) *Genetic algorithms*, *GeeksforGeeks*. GeeksforGeeks. Available at: https://www.geeksforgeeks.org/genetic-algorithms/ (Accessed: May 2, 2023).

Mallawaarachchi, V. (2020) *Introduction to genetic algorithms - including example code*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3 (Accessed: May 2, 2023).

Nicholson, C.V. (no date) *A beginner's Guide to Neural Networks and deep learning*, *Pathmind*. Available at: http://wiki.pathmind.com/neural-network (Accessed: May 2, 2023).

Sim, V. (2021) *Using genetic algorithms to train neural networks*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/using-genetic-algorithms-to-train-neural-networks-b5ffe0d51321 (Accessed: May 2, 2023).