[Mini Project 2] **Project Code : DSNN**

2D Self Driving Car Simulation using Artificial Neural Network

Project Duration : 15-Feb-2021 ~~ 01-Mar-2021
Submission Information : (via) CSE-Moodle.

Objective:
In this assignment, you need to control a car in a 2D racetrack environment using an evolutionary algorithm. The car has four controls as follows:

FA : Forward Acceleration
BA : Backward Acceleration
TL : Turn Left
TR : Turn Right

You are asked to develop a neural network model which you need to train using the given evolutionary algorithm in order to navigate the car on the given tracks.

Neural Network Structure:
The intermediate or hidden layer structures of the neural network may be as follows:

- layer1: Dense Layer 7 neurons to 10 neurons
  Rectified Linear Unit (ReLU)
- layer2: Dense Layer 10 neurons to 10 neurons
  Rectified Linear Unit (ReLU)
- layer3: Dense Layer 10 neurons to 10 neurons
  Rectified Linear Unit (ReLU)
- layer4: Dense Layer 10 neurons to 4 neurons
  Binarize operation

You need to modify the following files:

1. **net.js**
   You need to implement the following functions in this file:

   a. **weightInitializer(input_neurons, output_neurons):** You need to implement this function which should return the weight matrix 2D list) of shape (**output_neurons x input_neurons**). Each weight value is taken from a Gaussian distribution of mean 0 and standard deviation **sqrt(2/input_neurons),** which is known as **kaiming initialization**.

   b. **biasInitializer(neurons):** Should return the bias vector (1D list) of length neurons. Initialized as all zeros.

   c. **relu(x):** This function takes two types of arguments.
      i. Type1 is a list (which is an object in js): Should return relu applied on each element of the list
      ii. Type2 is a number: Should return relu of that number

   d. **binarize(x):** This function takes two types of arguments.
      i. Type1 is a list (which is an object in js) : Should return binarize applied on each element of the list
      ii. Type2 is a number: Should return binarize operation applied on that number
   The binarize operation on a number x is the following:
      if x is negative, return 0

else return 1

e. **forward(X) in class Layer:** Implement the forward function of a feedforward layer of a neural network. Should return the matrix multiplication of this.weight and X after which this.bias has been added : (WX + b)

f. **mutate() in class Layer:** Adds values sampled from a Gaussian Distribution of mean 0 and standard deviation mutationParameter to all the weight and bias values

g. **forward(X) in class Network:** Implement the forward function of this neural network by calling forward functions of the intermediate layers and calling relu activation functions in between. The final output is binary and hence the output of the last layer must be passed through the binarize function.

2. **evolution.js**
   You need to implement the following function in this file:

   a. **select():** Tasks in this function are as follows:
      i. this.fitness contains the fitness values of all the cars in the population. Find the probability of selection of the i-th car in the population is given by:
      ```
      Probability_selection = (Normalizing_factor) x
                          [ exp(this.fitness[i] / total_fitness)
                    ]
      ```
      where the normalizing factor ensures that the sum of all probabilities is equal to 1 and total_fitness is the sum of all fitness values.
      ii. Find the index of the mostfit individual and assign it to this.mostfit
      iii. Push the maximum fitness value of the population to the list this.maxfitvals
      iv. Sample the car index from the probability_selection distribution's cumulative density function.
      v. Create a new list newpop and add a cloned version of the fittest car to it. This won't be mutated because as you can see in the mutate function, i starts from 1.
      vi. Using (D) and (E), generates the new population. Note that chooseFittest parameter should also be used, in which we choose the fittest car with a probability of "chooseFittest". Ultimately, store the new population back in this.pop array. Note that objects are assigned by reference in JavaScript and hence you need to use the given clone function to assign by value.

Already existing functions that you might need to call in this code:

- **sqrt(num)** returns square root of a float
- **randomGaussian(μ, σ)** returns a value from a Gaussian distribution with mean μ and std σ
- **matrixmultiplication(M1, M2)**: returns the matrix multiplication of M1 and M2, here M1 and M2 are 2D lists in JavaScript
- **matrixaddition(M1, M2)**: returns the matrix addition of M1 and M2, here M1 and M2 are 2D lists in JavaScript
- **random():** returns a uniform random number between 0 and 1
- **clone(car):** (implemented in evolution.js file) returns a new car with the same neural network as that of car
- **exp(number)**: returns e raise to the power of that number

Please feel free to change the parameters mutationParameter and chooseFittest during evolution to achieve better fitness scores rapidly (change them in console, which opens after you press Ctrl + Shift + C).

You can easily change the following variable in the file ‘metadata.js’ to switch between different types of racetracks.

Details of Metadata.js:

num: takes the value 0 to 4 (both included). Each num is a different type of racetrack.

Your Tasks:

1. Write code in the files net.js and evolution.js in the specified function definitions.
2. Prepare a report which should include the plots between maxfitness and generation number for each of the racetracks, and your observations on controlling the mutationParameter and chooseFittest parameters during evolution of the population of cars.

Submission Details: (to be submitted under the specified entry in CSE-Moodle)

A zipped file containing the following files:

1. File net.js
2. File evolution.js
3. A brief (2-3 page) report/manual of your work.

Submission Guidelines:

1. You need to write code in JavaScript in this mini project. JavaScript is very much like C++ and Python and you can refer to the following link to have a quick look at the language: https://www.codecademy.com/learn/introduction-to-javascript/modules/learn-javascript-introduction/cheatsheet
   For any other questions about understanding the language syntax, feel free to contact the assigned TA (mentioned below).
2. Your programs should be able to run on a Linux Environment.
3. You should name your file as <GroupNo_DSNN.FileExtension>.
4. The submitted files should have the following header comments:
   // Group Number
   // Roll Numbers : Names of members (listed line wise)
   // Project Number
   // Project Title
5. Don't modify any other code in any of the attached library files or in any location where it is not specified.

You should not use any code available on the Web. Submissions found to be plagiarised or having used ML libraries (except for parts where specifically allowed) will be awarded zero marks.

For any questions about the assignment, contact the following TA:
Aditya Rastogi (r.aditya0824@gmail.com)