Cerebellar Model Articulation Controller

Homework 2 Report

Robot Learning

Nikhil Lal Kolangara (116830768)

GitHub: https://github.com/nikhilkolangara/ENPM-690—Robot-Learning-Homework-2

Date : 02/12/2020

Contents

1 Problem 1

Question 1 *Program a Discrete CMAC and train it on a 1-D function (ref: Albus 1975, Fig. 5) Explore effect of overlap area on generalization and time to convergence. Use only 35 weights for your CMAC, and sample your function at 100 evenly spaced points. Use 70 for training and 30 for testing. Report the accuracy of your CMAC network using only the 30 test points.*

Approach 1 *For the purpose of building a CMAC algorithm, the 100 evenly spaced sample points considered were from the range -100 to 100 with step size of 2 based on the 1-D function:*

$$y = x^2$$

The Data Points plotted on the graph forms a Concave-Up Parabola. The line is fitted across the Data Points

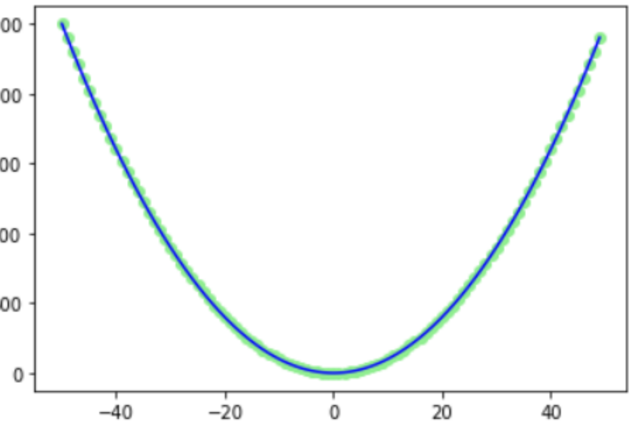


Figure 1: Data Points and Curve Fitted

In the discrete CMAC architecture, for each input there consists feature detecting neurons. Each association neurons are connected to one input from the input space with an equal weight assigned to each input.

Steps Followed:

1. The 1-D function mentioned above is used to to generate corresponding Y values from the X values.

2. The data points are plotted which forms a concave up parabola. The total number of points are 100.

3. The dataset is split into 70 for training and 30 for testing with randomization value as 100.

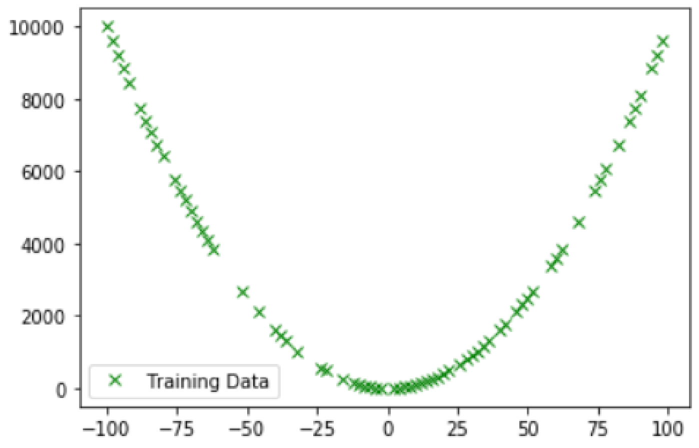


Figure 2: Training Data

4. The parameters are initialized and functions are declared and initialized to calculate and map the weights and the indices associated with it to the inputs.

5. A function for error minimization is declared. The function calculates Root Mean Square Error.

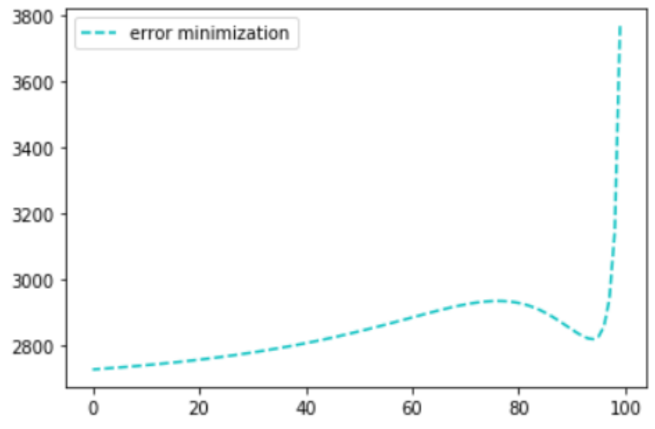


Figure 3: Error Minimization

6. Looping is instantiated, the weights are updated and the errors are minimized.

7. Finally, testing is performed by considering the updated weight, number of repetitions and calling the error function to predict the Accuracy.

Results using Discrete CMAC:

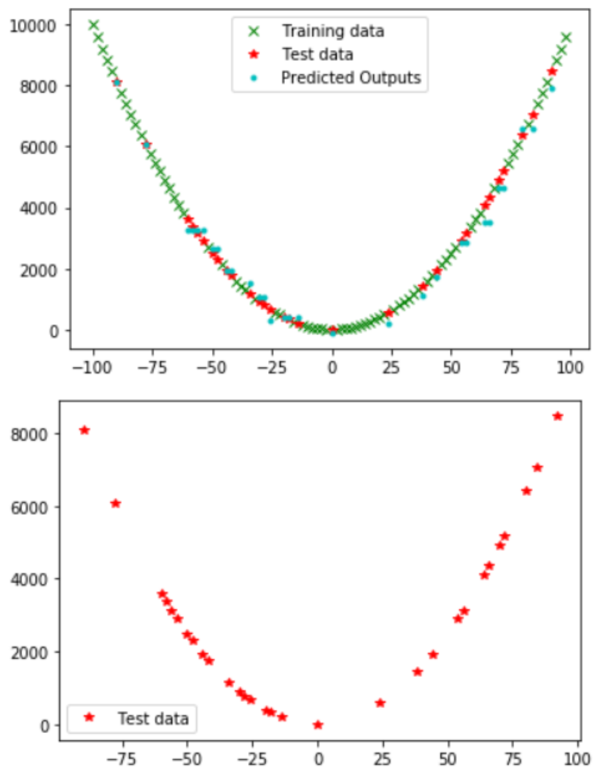


Figure 4: Outcome compared with the Test Data

2 Problem 2

Question 2 *Program a Continuous CMAC by allowing partial cell overlap, and modifying the weight update rule accordingly. Use only 35 weights weights for your CMAC, and sample your function at 100 evenly spaced points. Use 70 for training and 30 for testing. Report the accuracy of your CMAC network using only the 30 test points. Compare the output of the Discrete CMAC with that of the Continuous CMAC.*

Approach 2 *For the purpose of building a CMAC algorithm, the 100 evenly spaced sample points considered were from the range -5o to 50 based on the 1-D function:*

$y = 2 * x$

Based on the function, the Data Points are plotted on the graph which forms a Concave Up Parabola. The line is fitted across the Data Points.

The Data Points plotted on the graph forms a linear straight-line. A line is fitted across the Data Points

In the continuous CMAC architecture the sliding window concept is used.

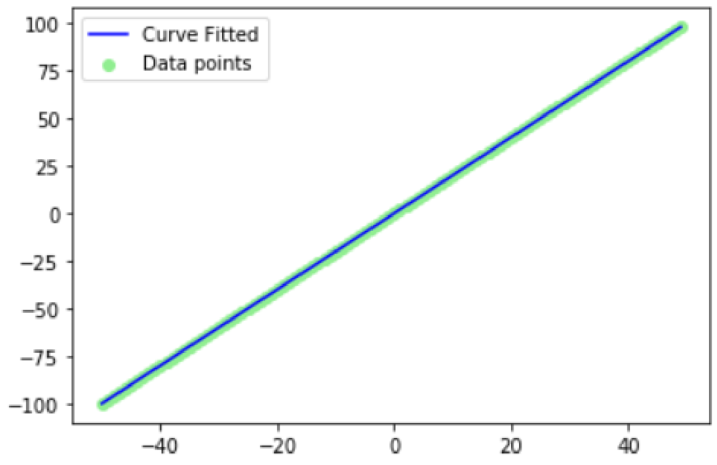


Figure 5: Data Points and Line Fitted

Steps Followed(The steps followed are same as Discrete):

1. The 1-D function mentioned above is used to to generate corresponding Y values from the X values.

2. The data points are plotted which forms a linear straight line. The total number of points are 100.

3. The dataset is split into 70 for training and 30 for testing with randomization value as 100.

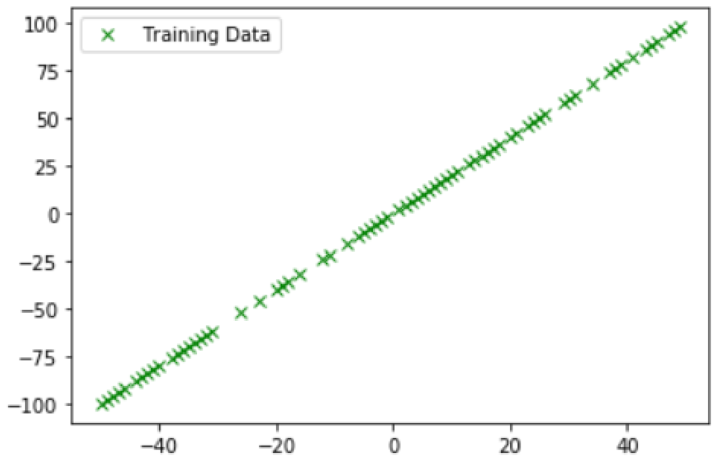


Figure 6: Training Data

4. The parameters are initialized and functions are declared and initialized to calculate and map the weights and the indices associated with it to the inputs.

5. The generalization factor value is 20.

6. A function for error minimization is declared. The function calculates Root Mean Square Error.

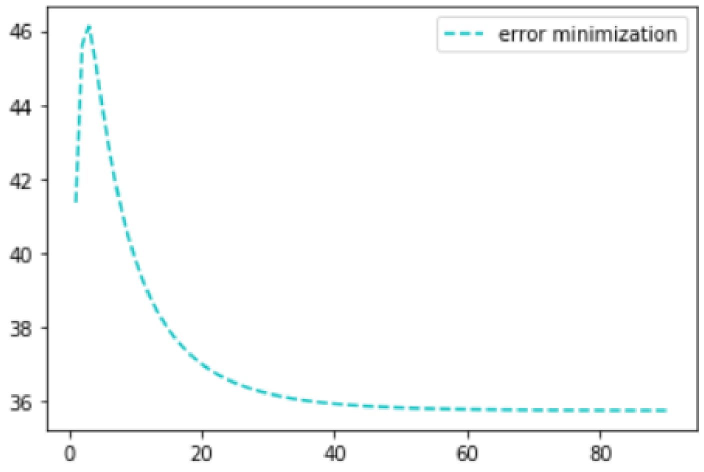


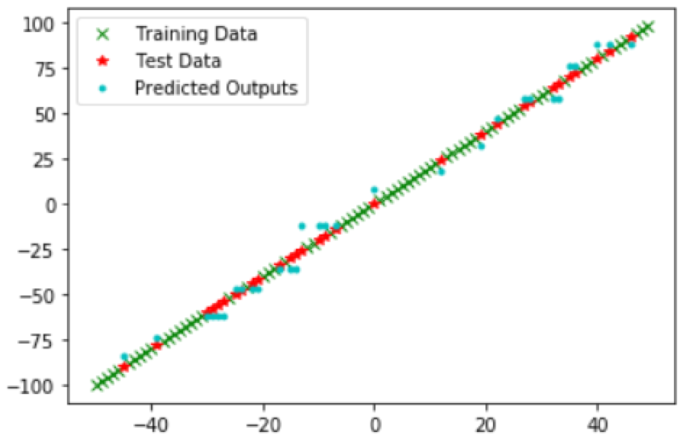
Figure 7: Error Minimization

7. Looping is instantiated to update the weights as per the previous error.

8. Inside the loop, the weights are calculated in a such a way that the window is determined using the highest, lowest and middle block by updating the corresponding indices.

9. Finally, testing is performed by considering the weight window, number of repetitions, and calling the error function to predict the Accuracy.

Results using Discrete CMAC:



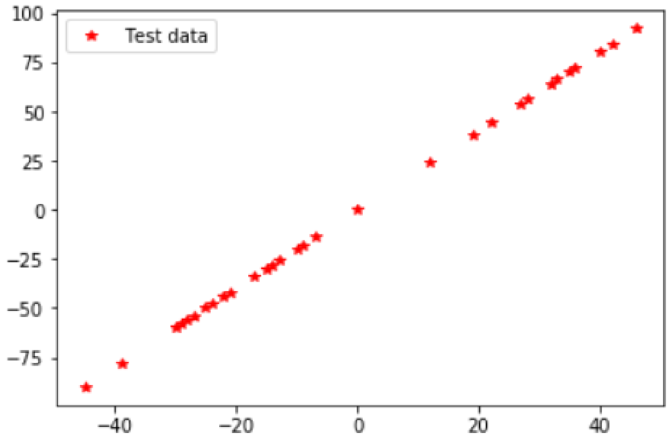


Figure 8: Outcome compared with the Test Data

3 Problem 3

Question 3 *Discuss how you might use recurrent connections to train a CMAC to output a desired trajectory without using time as an input (e.g., state only). You may earn up to 5 extra homework points if you implement your idea and show that it works.*

Approach 3 *Recurrent connections basically forms a type of neural networks which has recurring layers of neurons. These layers are hidden layers and the inputs of one layers are the outputs from the previous layers. These types of neural networks are popularly known as RNNs and few examples of such neural networks are: Feed Forward Neural Networks, Back Propagation, etc.*

Since we know that a 2D CMAC can have 3 layers where 2 inputs first goes through a layer of feature detecting neurons which quantize the inputs to the next layer which contains association neurons. A recurrent network can be implanted into the CMAC at the very first layer. Here, the feedback connections, for example consider a Back propagation Algorithm where the error signal is given as feedback to the first layer, similarly, this technique can be used to implement temporal relations in the network.

Referring to the paper proposed by Jinzhu Peng, Yaonan Wang and Wei Sun on Trajectory-Tracking Control for Mobile Robot Using Recurrent Fuzzy Cerebellar Model Articulation Controller

Controlling of the velocity and the angular measurement in spherical coordinate system is achieved by manipulating the torques for the left- and the right-wheels. Hence time as an input is not considered and the velocity error and its rate, and the azimuth error and its rate are considered as the inputs, and the driving torque required for controlling the two wheels are considered as the output. Since the neural network is a 4 input and 2 output network, the error minimization technique used helps reduced and calculate the approximate value required for the Mobile Robot to track its trajectory.

It consists of 5 layers, where the 4th layer is the Associate layer and the 5th layer is the Output layer. The output signal generated by the 5th layer is fed back to the 1st layer which is the Feedback layer.