1 Discrete Cmac

1.1 Description

1. **Define input space** - In the implementation, the input space was defined over the range[0,10]

2. **Association Cells** - This is a 1D cmac, so the number of association cells was chosen to match the number weights.In the implementation,35 weights with 35 association cells.

3. **Quantize input** - The input space is quantized so as to perform a table lookup. The quantization is achieved using the formula given below.

$$q_x = \left\lfloor resolution * \frac{x - xmax}{xmin - xmax} \right\rfloor \tag{1}$$

   so in the implementation, we have,

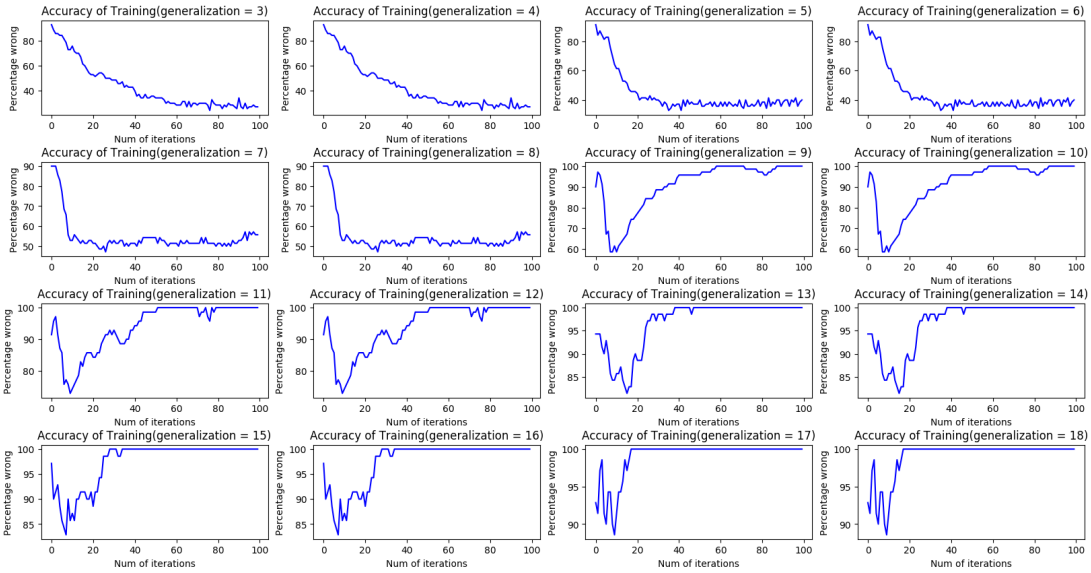
$$q_x = \left\lfloor 35 * \frac{x - 0}{10 - 0} \right\rfloor \tag{2}$$

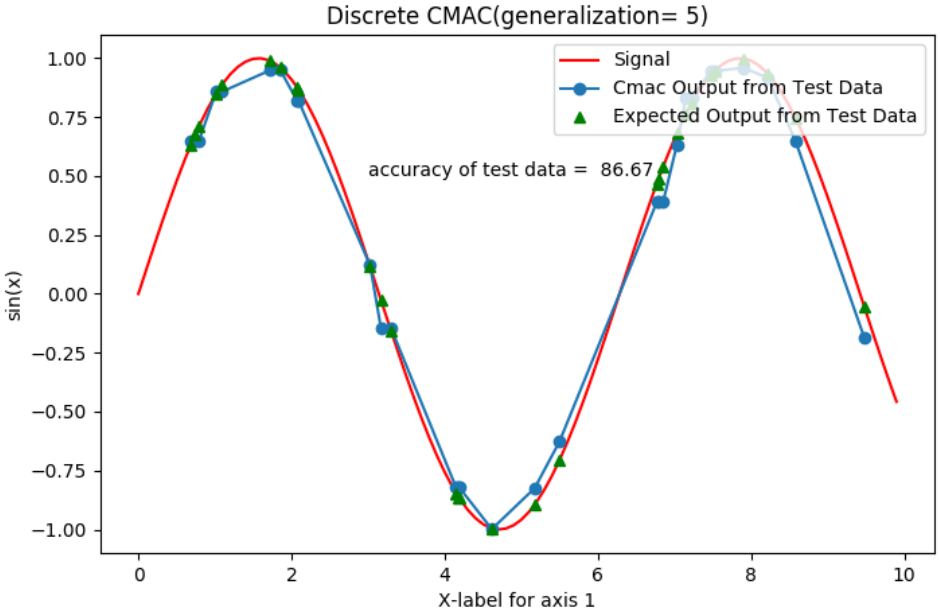
4. **Generalization Number, g** - This number indicates the degree of similarity between tasks. Odd numbers was utilized as centering weights around quantized input was preferable. When an input is received, the association cells activated are centered around the quantized input. For example, suppose the quantized input of 3 is 3 with a generalization number of 5, this means that 2 cells before 3(0 and 1) and 2 cells after 3(4 and 5) are activated.

1.2 Results

Generalization Vs Convergence: Clearly from the graph shown below,it can be noted that as the generalization number increases, the model has a hard time converging.This can be attributed to the fact that the generalization number,g indicates to a degree the similarity between tasks. As g increases, the model becomes weak(wrong model) as it tends to consistently learn the wrong thing by categorizing unrelated tasks as similar,hence a high bias. Ideally, g is chosen so that tasks which are related have similar values whiles tasks which are different have clear cut distinct values.

Accuracy: Below is a graph that depicts the accuracy of the Discrete Cmac. After running a couple of times, it can be noted that the accuracy hovers between 70-85%





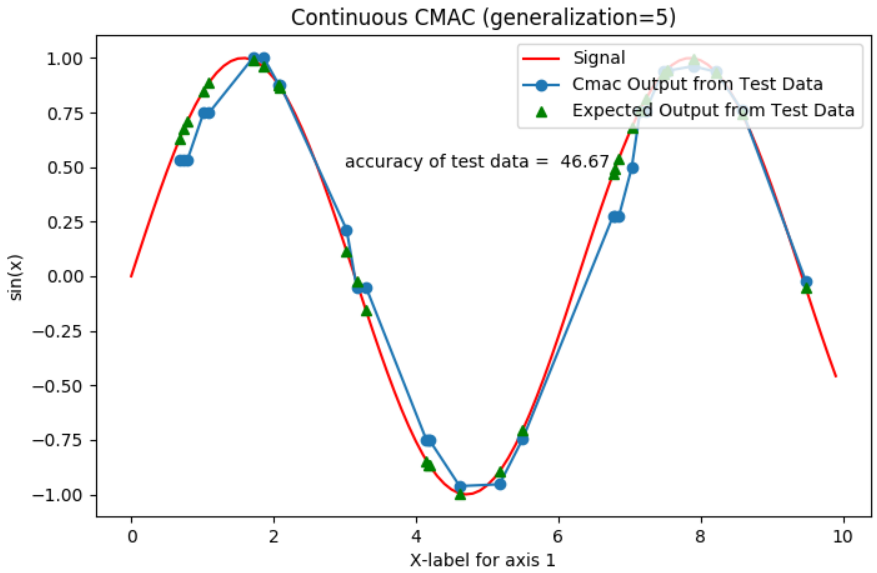
2 Continous Cmac

2.1 Description

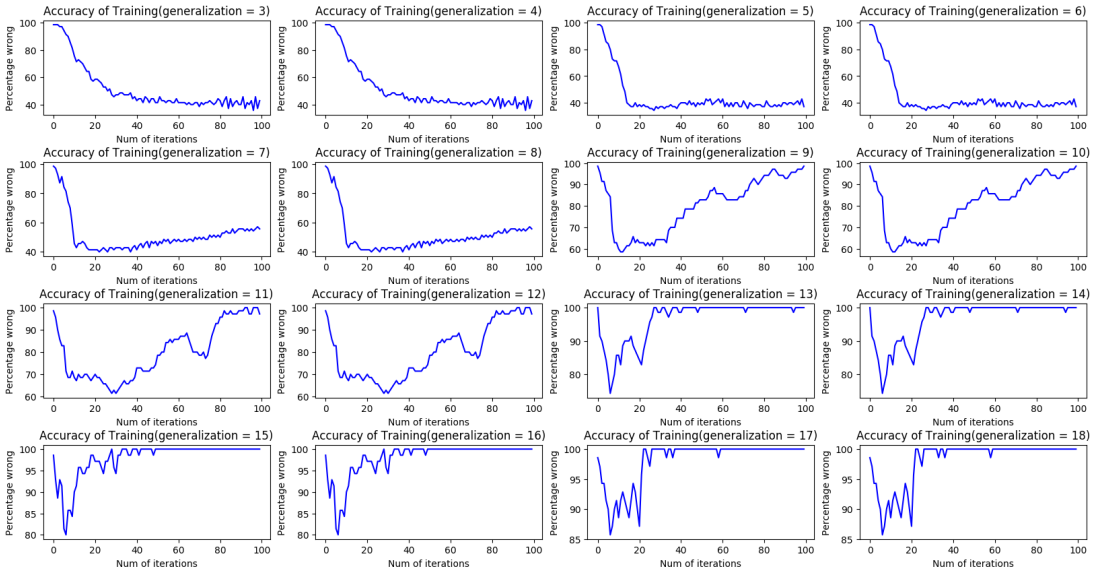
2.2 Results

Accuracy: Below is a graph that depicts the accuracy of the Continous Cmac. After running the the code a couple of times, it can be noted that the accuracy

hovers between 40-60%



Discrete Vs Continous Cmac: The "convergence Vs generalization" graphs for both continous cmac and discrete cmac are relatively similar in particular for generalization numbers g = 3 to 6. However,the accuracy of the test results are clearly different. For the same parameters used in training the Discrete Cmac(generalization number, number of iterations in training,accuracy used in training,learning rate), it can be noted that the continous Cmac has a significant drop in accuracy as the model clearly seems to overfit.This is because the continous cmac slides over more weights. Therefore, although a generalization number of 5 is used, more than 5 weights are changed during each update and this means that more inputs are categorized as similar and that is undesirable. To rectify this the generalization number can be reduced and number of iterations increased to improve perfomance. Also, the type of sliding window utilized has an impact on the results.



3 Recurrent Connections with Cmac

Problem: Cmac can be particular useful in modelling connnections that are only state dependent. Most recurrent connections such as recurrent neural networks employed in speech recognition, music generation etc. are temporally dependent. So if a given task is paused midway, all the relevant information is lost as the history of the information seen previously matters. Although changes can be made to such networks by storing certain information, this is usally intractable as it means a different logic to determine where the current function is located at. Therefore, a model that is only state dependent becomes necessary. Cmac can be used to model a state dependent recurrent connections and this is shown below.

Modelling Cmac to output desired trajectory: The output of cmac is now fed as an input to P(logic depends on the particular problem), then P decodes the appropriate weights to excite. This is illustrated by the diagram below.

What if two different inputs have the same ouput value. How do we know which input to excite next?
The decoding function **P** must take this into account. Therefore, P is modified to be $P(\sum, weights)$. Therefore, different inputs with same output weight sum are treated as different. The diagram is now modified to be the following.

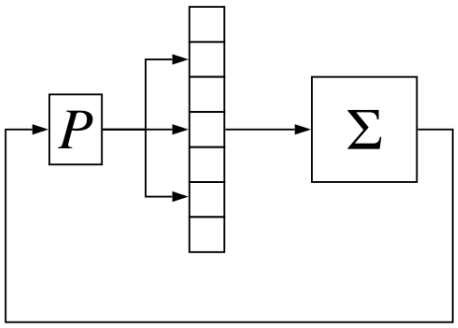


Figure 1: Recurrent Cmac

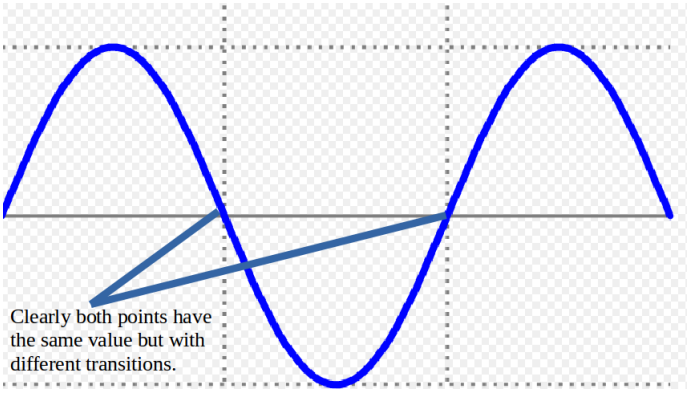


Figure 2: Trajectory

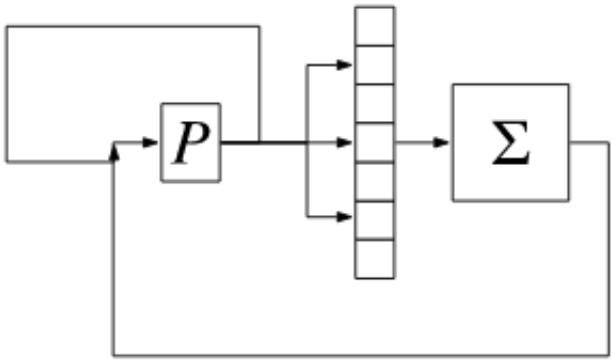


Figure 3: Modified model