FQCMAC: A Multi-Resolute Cerebellar Learning Memory Model

S. D. Teddy[1], E. M-K Lai[2] and C. Quek[3]
School of Computer Engineering, Nanyang Technological University, Singapore.
[1]sdteddy@pmail.ntu.edu.sg
[2,3]{asmklai,ashcquek}@ntu.edu.sg

Abstract

The CMAC neural network is a simple yet powerful modeling tool for various type of applications, such as in control. It is well-established that the network size in a CMAC based system increases exponentially with the number of input variables. In this paper we propose an adaptively quantized CMAC (i.e. FQCMAC) as a brain inspired learning memory model whereby the quantization step size is autonomously adapted based on the characteristics of the data to be modeled. The proposed approach employs a simplified version of the DIC partitioning technique to capture the input data distribution. We subsequently demonstrate through a highway traffic flow modeling application that FQCMAC outperforms its CMAC counterpart.

1 Introduction

The Cerebellar Model Arithmetic Computer (CMAC) neural network [**?**, **?**] has the advantages of simple computation, fast training, local generalization and ease of hardware implementation that render it as particularly useful and effective in control applications [**?**, **?**, **?**, **?**, **?**, **?**]. The proof of learning convergence of the CMAC network has also been established in a number of literatures [**?**, **?**, **?**].

The development of the CMAC neural network was inspired by the neurophysiology of the cerebellum, a part of the human brain which is primarily involved with the control of movements, in particular those of the limbs, hands, and eyes. The human cerebellum functions by storing and retrieving information required to regulate thousands of muscles in the production of controlled behaviors as a function of time [**?**]. As a functional model of the cerebellum, CMAC manifests as an associative memory neural network and computes by a table lookup operation.

Due to this table lookup nature of CMAC, its network size increases exponentially with each addition of a new input variable. Hence, an efficient memory allocation scheme is required to optimize the memory usage in CMAC. The basic CMAC has a static structure in which the resolution of the memory cells is solely dependent on the rigid (even) partitioning of the input space. And since this resolution is not *adapted* according to the characteristics of the input training data, there is no guarantee that the CMAC network employs an optimal memory allocation scheme.

On the other hand, neuro-psychology has established that the human brain organizes information in a highly non-linear manner. In this paper, we propose a multi-resolute cerebellum-inspired learning memory model, where the memory resolution of the network is autonomously adapted according to the information distribution of the input training data. The objective of such an approach is to formulate a memory mapping mechanism to achieve non-linearity in storage allocation, so as to enhance the efficiency of the CMAC memory allocation based on the characteristics of the data to be modeled. The advantages of having such a mapping scheme are two-folded: (1) to provide improved accuracy and fidelity in the stored and computed outputs of the CMAC network, and (2) to maintain a low computational complexity of the CMAC based system with increased data complexity.

The rest of the paper is organized as follows. In Section 2, the anatomical structure of the cerebellum is highlighted to support the use of non-uniform quantization in CMAC. Section **??** presents the description of the proposed FQCMAC architecture. An experiment on traffic trend prediction is provided in Section **??** to demonstrate the performance of the proposed FQCMAC. Section **??** concludes this paper.

2 Cerebellum and its Multi-Resolution Organization

CMAC is a functional model developed to emulate the information-processing characteristics of the

human cerebellum. The cerebellum, which in Latin means *little brain*, is located at the bottom rear of the head (the hindbrain) directly above the brainstem and is highly recognizable for its structural regularity. The cerebellum is divided into several distinct regions, each of which receives projections from different portions of the brain and spinal cord and projects to different motor systems, suggesting that regions of the cerebellum perform similar computational operations but on different inputs [**?**].

Together with the cerebral cortex and striatum (part of the basal ganglia formation), the cerebellum constitutes the brain's procedural memory system. All the brain subsystems involved, including the cerebellum, modify their circuitries in the service of the formation of procedural memory. Changes in the cerebellum's circuitry occur in two ways. Firstly, the synaptic transmissions are modified through a cellular mechanism called *Long Term Potentiation* (LTP), which results from the comparisons of the goals, commands and feedback signals associated with a particular movement. Secondly, cerebellar plasticity (which refers to the alteration of the synaptic connections) is achieved as a result of training and repeated exposures.

Cerebellar plasticity studies by William Greenough and his colleagues inspired the development of a multi-resolution CMAC architecture [**?**]. In his studies, rats were given acrobatic training by challenging them to acquire complex motor skills necessary to traverse a series of obstacles. It is discovered that rats with such training developed an increased volume of the parallel fiber layer in the cerebellar cortex, and this increases the number of synapses onto the Purkinje cells without an increase in synaptic density. This suggests that the cerebellum organizes its learned knowledge in a non-linear manner, where repeated training yields more synaptic connections as well as a finer resolution in the neural circuitry, and resulting in more precise control.

Such observations provided the motivations for our proposed FQCMAC architecture, a multi-resolution variant of the CMAC network which manages the memory (receptive field) allocation based on the information distribution of the input training data.

3 FQCMAC: A Learning Memory Model

The basic CMAC neural network is an associative memory model of the cerebellum, where it simply performs a mapping of the multi-dimensional input-output data tuples. The CMAC memory is visualized as a hypercube array of storage cells. These cells are employed to store sets of *weight* values, which constitutes the computed outputs of the CMAC network. The elements in the input vector are used as indices to activate a particular set of storage cells, and the summation of the stored values in these active cells forms the computed output of the CMAC. In the CMAC network, the computing cells are organized as a multi-dimensional memory array, and the resolution (receptive field) of these cells are computed through an even quantization of the input space along each of the input dimensions. Each of the computing cells therefore cover a region of similar size in the input surface.

However, certain parts of the input space tend to be more important or significant as compared to the rest. Depending on the underlying dynamics and characteristics of the system to be modeled, some regions of the input surface will contain more information than others. In such cases, adopting an uniformly quantized memory resolution throughout the entire problem surface may not be a preferable option, as it could potentially result in many unused memory cells. Drawing inferences from the findings of neuroscience, where it has been demonstrated that the human brain allocates more synaptic connections to cortical areas associated with parts of the body such as fingers or tongue, where there are higher throughput of sensory information as compared to the arms or legs, we proposed a novel method to maximize the memory utilization of the CMAC model to enhance the resolution of its computed output. The resulting architecture, named Fuzzy-Quantized CMAC (FQCMAC), employs variable quantization of the training input values based on the observed information distribution in each of the input dimensions. In the proposed FQCMAC model, more memory cells are allocated to *significant* areas in the input space which are perceived as containing more information than the rest. These regions generally corresponds to those densely populated areas of the input surface.

Figure 1: 2D FQCMAC input surface

Figure **??** depicts the 2D illustration of the concept of the multi-resolute allocation of memory cells in the proposed FQCMAC network.

3.1 Adaptive Memory Allocation

The initial step to create a variable quantization memory mapping in the FQCMAC network is to identify areas which potentially contain more information as compared to the rest, and subsequently allocating more memory cells to these regions. This is achieved by partitioning the input space based on their information content. In particular, we are looking for densely populated areas in which large amount of data points are clustered together within close proximity. This is because densely populated area potentially contains more entropy or information content which justifies the use of more memory cells to efficiently model and capture the inherent characteristics of the data points in the region.

The partitioning of the input space is performed on a per-dimension basis. A simplified version of the Discrete Incremental Clustering (DIC) [**?**] method is employed to partition the overall input space into a set of fuzzy clusters. DIC was selected as the partitioning method due to its high efficiency and low computational complexity; and it does not require the prior specification of the number of partitions to be formed before the algorithm commences.

Figure 2: A sample partitioning of the DIC technique

An illustration of a sample result of the DIC partitioning algorithm is given in Figure **??**. The DIC algorithm computes a set of trapezoidal-shaped fuzzy sets to denote the partitions identified, where the densely populated areas in the input space are represented by the kernels of the fuzzy sets. From the computed partitioning result, we obtain two types of regions: the kernels and the overlapping regions. Based on our notion, the kernel areas are the important or significant regions that would be allocated more memory cells in the proposed FQCMAC network. We introduce a parameter λ to regulate the overall proportion of memory cells allocation to the two types of regions.

Let the total number of memory cells per dimension be M. The memory allocation process for the two types of regions is formulated as follows:

$$M_k = \left\lfloor \frac{\lambda \cdot A_k}{A} \cdot M \right\rfloor \quad (1)$$

$$M_o = M - M_k \quad (2)$$

$$A = (\lambda \cdot A_k) + A_o \quad (3)$$

$$\lambda \geq 1 \quad (4)$$

where:

M_k	total number of memory cells for kernel regions
M_o	total number of memory cells for overlap regions
A_k	total area of kernel regions
A_o	total area of overlap regions
A	total conceptualized area
λ	the memory cell allocation constant

A large value of λ gives more weightage or emphasis to the kernel regions of the fuzzy sets (partitions); and when λ equals to unity, memory cells are allocated based on the arithmetic ratio of the two region types.

Subsequently, the allocation of memory cells to each of the individual areas in a region type is computed as

$$M_i = (A_i / A_{type}) \times M_{type} \quad (5)$$

where A_i is the size of the i-th region and *type* refers to the particular region type to which region i belongs.

Figure 3: Variable memory distribution in a kernel

Subsequently, in order to introduce non-linearity to the memory cell allocation process, the quantization step sizes inside the kernel regions are non-linearly spaced and symmetrical about the centers of the kernel regions, while the quantization step sizes in the overlapping regions are linearly spaced. The degree of non-linearity in the kernel regions is governed by a parameter μ, and the μ-law quantization technique is subsequently employed to vary the distribution of the memory cells in these regions. Figure **??** illustrates the variable memory quantization inside a kernel region.

The distribution of the memory cells is denser towards the center of a kernel area. Assuming a particular input *in* that appears within the i-th region of the corresponding input dimension, the memory quantization equations are given by:

$$\mathcal{F}(in) = S_i + \left[\frac{M_i}{2} \pm \left(\frac{y}{A_i/2} \right) \cdot \frac{M_i}{2} \right] \quad (6)$$

$$y = \frac{\frac{M_i}{2} \cdot \left(1 + \frac{\mu \cdot |in - cp_i|}{M_i/2} \right)}{\log(1+\mu)} \quad (7)$$

where:

S_i	the starting index of the i-th region
$\mathcal{F}(in)$	the indexing function for input in
i	the region within which in appears
cp_i	the center (mid-point) of the i-th region
μ	the degree of nonlinearity
M_i	the number of cells allocated to the i-th region
A_i	the total size of the i-th region

3.2 Network Learning

The proposed FQCMAC network adopts a modified form of the Widrow-Hoff learning rule to implement the computations for *Weighted Gaussian Neighborhood Output* and *Weighted Gaussian Neighborhood Update*. The learning equations are:

$$\mathbf{Z}^i_{\mathbf{V}_j} = \frac{\sum_{\mathbf{k}\in\mathbf{K}} (G_{\mathbf{k}} \cdot \mathbf{W}_{\mathbf{k}})}{\sum_{\mathbf{k}\in\mathbf{K}} G_{\mathbf{k}}} \quad (8)$$

$$\mathbf{W}^{i+1}_{Q[\mathbf{V}_j]} = \mathbf{W}^i_{Q[\mathbf{V}_j]} + \mathbf{\Delta W}^{i+1}_{Q[\mathbf{V}_j]} \quad (9)$$

$$\mathbf{\Delta W}^{i+1}_{Q[\mathbf{V}_j]} = \alpha \frac{G_{Q[\mathbf{V}_j]} \mathbf{E}^{i+1}_{\mathbf{V}_j}}{\sum_{\mathbf{k}\in\mathbf{K}} G_{\mathbf{k}}} \quad (10)$$

$$\mathbf{E}^{i+1}_{\mathbf{V}_j} = \mathbf{Z}^{i+1}_{\mathbf{V}_j} - \mathbf{D}_{\mathbf{V}_j} \quad (11)$$

where:

i	training iteration number
$\mathbf{V}_j$	the multidimensional input vector
$Q[\cdot]$	the quantization function
$\mathbf{Z}^i_{\mathbf{V}_j}$	the output at training iteration i
$\mathbf{D}_{\mathbf{V}_j}$	the expected output given the input vector
$\mathbf{W}^i m, n$	the content of the memory cell at index $[m, n]$
$G_{\mathbf{k}}$	the gaussian weighting factor of cell $\mathbf{k}$ in $\mathbf{K}$
$\mathbf{K}$	the set of neighborhood cells

Neighborhood output retrieval is employed to smoothen the computed output of the proposed FQCMAC network due to the quantization error while a Gaussian weighting factor ($G_{\mathbf{k}}$) is applied to distribute the learning error among the cells in the neighborhood. The Gaussian weighting function is defined as:

$$G_{\mathbf{k}} = (1 - d_{\mathbf{k}}) e^{-d^2_{\mathbf{k}}/2\gamma^2} \quad (12)$$

where $d_{\mathbf{k}}$ is the normalized Euclidean distance between the cell $\mathbf{k}$ and the applied input, and γ is the parameter that controls the width of the Gaussian weighting function.

4 Experiments and Results

This simulation is conducted to evaluate the performance and effectiveness of the proposed FQCMAC network in approximation and data modeling using a set of highway traffic flow data obtained from [**?**].

The data were collected at a site (denoted as Site 29) located at exit 15 along the eastbound Pan Island Expressway (PIE) in Singapore (see Figure **??**) using loop detectors embedded beneath the road surface. There are a total of five lanes at the site, two exit lanes and three straight lanes for the main traffic. For this experiment, only the traffic flow data for the three straight lanes were considered. The traffic data set has four input attributes, namely the time and the traffic densities of the three lanes. The purpose of this simulation is to model the traffic flow trends at the site using the FQCMAC network. The trained FQCMAC network is then used to obtain predictions for the traffic density of a particular lane at a time $t + \tau$ where $\tau = 5, 15, 30, 45$ and 60 minutes.

For the simulation, three cross-validation groups of training and test sets are used. The square of the Pearson product-moment correlation value (denoted as $\mathcal{R}^2$) [**?**] is used to denote the accuracy of the computed predicted traffic trends. Two different sets of simulations are performed based on a neighborhood size of 0 (i.e. 1-point update and retrieval only) and a neighborhood size of 0.1 (i.e. neighborhood update and retrieval) respectively for both the FQCMAC and CMAC networks. Each network uses 8 cells per dimension, with a learning constant of 0.1 and γ equals 0.5. The performances of the FQCMAC network are subsequently benchmarked against those of its CMAC counterpart. The average accuracy of the predictions (denoted as $Avg\mathcal{R}^2$) by the FQCMAC and CMAC networks across the three cross-validation groups as τ increases from 5 to 60 minutes for all the three lanes for a neighborhood size of 0 and 0.1 are depicted as Figure **??** and Figure **??** respectively.

Figure 5: Avg $\mathcal{R}^2$ against time for N = 0

Figure 6: Avg $\mathcal{R}^2$ against time for N = 0.1

Two other indicators are used for the comparison. The first indictor is *Var* (The change in $Avg\mathcal{R}^2$ value from $\tau = 5$ to $\tau = 60$ minutes expressed as a percentage of the former) and the second is *AvgVar*, the mean *Var* values across all the three lanes. These two indicators reflect the consistency of the predictions made by the benchmarked systems over the time interval as τ changes from 5 to 60 minutes across the three lanes.

Table **??** provides the comparison between the FQCMAC and CMAC networks. One can observe that FQCMAC consistently outperforms its CMAC counterpart. This is because FQCMAC allocates more memory cells to the areas which potentially contain more information as compared to CMAC where the memory cells are uniformly distributed across the entire input space. This feature of FQCMAC enhances

Figure 4: (a) Location of Site 29 along PIE (Singapore); and (b) Actual site at exit 15

the fidelity of the outputs by increasing the memory resolution in the important areas and smoothening the outputs by reducing the amount of empty cells.

5 Conclusions

In this paper we have presented the FQCMAC network, a brain-inspired learning memory system, as an enhancement to the original CMAC model. FQCMAC outperforms the CMAC network by enhancing the memory utilization as well as increasing the resolution of the computed output. This is achieved by the autonomous adaptation of the quantization step size in FQCMAC based on the characteristics of the training data distribution. The proposed FQCMAC is evaluated with a set of highway traffic flow data. Simulation results have demonstrated that the FQCMAC outperforms CMAC for the same network parameter settings. Further research in this direction includes a more detailed evaluation of the algorithm as well as exploring experimentations in other application areas.

Table 1: Comparison of results from FQCMAC and CMAC testing

Networks	**Lane 1** ***Var*** **(%)**	**Lane 2** ***Var*** **(%)**	**Lane 3** ***Var*** **(%)**	**Avg** ***Var*** **(%)**
CMAC (N = 0)	17.65	17.37	18.88	17.97
FQCMAC (N = 0)	14.17	16.63	23.38	18.06
CMAC (N = 0.1)	10.04	12.38	8.61	10.34
FQCMAC (N = 0.1)	7.70	7.52	12.81	9.34