Receptive Fields for CMAC. An Efficient Approach

Walter S. Mischo

Fachgebiet Regelelemente und Regelungstechnik, Technische Hochschule Darmstadt

Abstract

This article will report on an extension to AMS – a CMAC-like associative memory – which has already been successfully applied e.g. in Learning Control Systems ([1]). The algorithm proposed here removes a well known disadvantage of CMAC: the discontinuous behaviour of its input processing receptive field and therefore considerably defuses the conflict – typically CMAC – between fast training but coarse resolution and fine resolution but slow training.

1 Association in CMAC

A point \( \vec{s} \) of the input space \( S \), which normally is a multidimensional interval in \( \mathbb{R}^n \):

\[
S = [s_{1\min}, s_{1\max}] \times \ldots \times [s_{n\min}, s_{n\max}]
\]

(mostly we consider: \( s_{1\min} = \ldots = s_{n\min} = 0, s_{1\max} = \ldots = s_{n\max} = \max \)) is discretized by certain \( \epsilon_i > 0 \) (without losing generality, here: \( \epsilon_i \equiv \epsilon \equiv 1 \)) and represented then by \( \varrho \) different hypercubes with the edge length \( \epsilon_c \equiv \epsilon \). Each hypercube is displaced against each other by a constant vector. For instance\(^1\) let us assume a displacement in each dimension which agrees an ordinal number of each hypercube, i.e. for an \( \vec{s} \) with sufficiently small positive components hypercube 1 has its "lower left" corner in point \( (0, \ldots, 0) \), hypercube 2 in \( (1, \ldots, 1) \) and so on. By extending this scheme in each dimension we get \( \varrho \) coordinate systems and can calculate a new (coarse grid) representation \( \vec{a}_j \) of a point \( \vec{s} \in S \) in coordinates of the coordinate system \( j \) ((\( s_i \))\( \equiv \) modulo \( \varrho \)):

\[
a_{ji} = \begin{cases} \lfloor \frac{s_i}{\epsilon} \rfloor, & \text{if } (s_i)_{\varrho} > j -1 \\ \lfloor \frac{s_i}{\epsilon} \rfloor - 1, & \text{otherwise} \end{cases}
\]

which can be seen simply as a rounding operation. Normally we transpose the indices to \( a_{ij} \) thus obtaining the association matrix \( A = (a_{ij})_{i=1}^{\varrho} \times \). The columns of \( A \) – the "lower left corners" of the \( \varrho (\varrho \times \ldots \times \varrho) \)-hypercubes mentioned above – are the so called association cells, of which the intersection is a \( (1 \times \ldots \times 1) \)-hypercube representing \( \vec{s} \) at discretization \( \vec{\epsilon} = (1, \ldots, 1) \).

\(^1\)Although this is not the best possibility, c.f. [3] and fig. 1 (left)
The code for each of the \( \rho \) association cells is a vector of \( n \) entries, which is transferred via hash coding to a memory address, where a related value, a (synaptic) weight, is stored. The weights for all \( \rho \) association cells are summed up to yield the output. Learning rules for the adaptation of the weights can be found e.g. in [2]. For a two dimensional case and \( \rho = 50 \) a possible configuration of association cells (squares) is shown in fig. 1 (left).

Figure 1: Left side: two dimensional association area with \( \rho = 50 \) and a hypercube displacement of \((1,7)\) instead of \((1,1)\) (as in the text); right side: resulting superposition of receptive fields in the new algorithm

2 The New Algorithm

As is clear from above and fig. 1 (left) the sphere of influence of one synaptic weight is a \( \rho \times \ldots \times \rho \)-hypercube in the input space. Each time an input falls into this area, the weight is updated following the associated output target with a learning rate generally depending on time but not on the location of the input point in the hypercube. So each weight tends to learn the average of all output targets falling into it’s hypercube. Thus the resolution decreases with increasing \( \rho \). On the other hand a large \( \rho \) causes fast initial training convergence, because only few training points (input points + associated output targets) sparsely scattered over the input space result in a rough overall "idea" of the function to be learned, whereas in the case of a small \( \rho \) the function is unknown on almost the whole input space (therefore the fundamental parameter \( \rho \) is called the
amount of generalization).

In [4] and [5] one can find first thoughts on introducing receptive fields in CMAC - but mainly theoretical considerations. Now the idea is to introduce a location dependent learning factor under the restriction that the additional calculations are to be minimized. At first distances are defined between an input point and the association cells, as indicated in fig. 1 (left), e.g. as simple "city block" distance (sum of component distances) by

\[ d_{ij} = \sum_{l=1}^{n} d_{il}, \]

\[ d_{ij} = \begin{cases} 
(s_i)_o - (j-1) - \lfloor d/2 \rfloor, & \text{if } (s_i)_o > j - 1 \\
(s_i)_o - (j-1) + \lfloor d/2 \rfloor, & \text{otherwise.}
\end{cases} \]

Then a receptive field function ("fire rate" function) is applied on each of these distances, e.g. a gaussian distribution (other "radial basis" functions are possible and yield similar results):

\[ \phi_j = \phi(d_{ij}) = A \cdot \exp(-(ad_{ij}/d_{\text{max}})^2), \quad \text{where } d_{\text{max}} = \begin{cases} 
\frac{n^2}{2} & \text{if } \rho \text{ is even} \\
\frac{n^2(n-1)}{2} & \text{otherwise}
\end{cases} \]

is a constant only depending on the actual configuration. The factors \( a, A \) are selected e.g. as \( a = 2.8 \) or \( 5.1, A = 255 \). As the value range of \( d_{ij} \) is small and discrete, a table for \( \phi \) is sufficient.

As a result, the superposition of the "fire rates" of the association cells responding to one input point (\( \rho = 50 \)) is shown in fig. 1 (right). These fire rates are involved in summing up the output by

\[ p^{(k)} = \frac{\sum_{i=1}^{\rho} \alpha^{(k)}_i \phi^{(k)}_i w^{(k)}_i}{\sum_{j=1}^{\rho} \alpha^{(k)}_j \phi^{(k)}_j} \]

where the index \( k \) stands for the actual access step and \( \alpha^{(k)}_j = 1 \) or 0 denotes whether a weight \( w_j \) is already trained in step \( k \), or not. The \( \phi^{(k)}_j \) are used in training mode as additional learning factor with appropriate scaling, e.g. by dint of:

\[ \Delta w^{(k)}_j = \phi^{(k)}_j \frac{\sum_{i=1}^{\rho} \phi^{(k)}_i}{\sum_{j=1}^{\rho} \phi^{(k)}_j} \left( p^{(k)} - p^{(k)} \right) \]

where \( p^{(k)} \) means the training value for access step \( k \), i.e. a mapping \( \bar{s}^{(k)} \mapsto \bar{y}^{(k)} \) is required.

3 Results

The effect of the new algorithm on the mean absolute error (judged against the value range) and the training index (number of already trained weights: \( 1/\sum_{j=1}^{\rho} \alpha^{(k)}_j \) over the number of training points for a two dimensional test function (square sine surface: \( \sin^22\pi \frac{x}{y_{\text{max}}} \cdot \sin^22\pi \frac{y}{y_{\text{max}}} \)) is shown in fig. 2 for two generalization levels (\( \rho = 16, 128 \)), the latter being realized in the old and the new algorithm.

This shows clearly the bridge between fast initial learning and final convergence inherent in the new algorithm. The computational cost is here about 40% more of calculation
Figure 2: Left side: mean relative errors in % versus $10^3$ training points; right side: mean training indices in % versus $10^3$ training points.

time per access step (nearly independent on $\rho$ and training/read mode) on a standard workstation (SUN 4/60). In an existing software simulation only some code additions were needed leaving the association algorithm principally unchanged.

Applied to an existing simulation of a learning control system it has been shown ([6]), that the new algorithm yields better results especially in the initial on-line learning phase, when the behaviour of an unknown process has to be learned.

4 Conclusion and Future Aspects

It has been shown that it is possible to implement "numerically" smooth receptive fields in CMAC in an efficient and simple way. This leads to faster convergence which is desirable for control applications.

Actual work on AMS deals with the support of optimization strategies, which also is of interest for use in the field of control – and a hardware implementation of a simple AMS version for high speed applications e.g. controlling a multi-fingered robot gripper.

Acknowledgements

The author wishes to thank Prof. P.C. Parks from the Royal Military College of Science Shrivenham, UK, for his remarks and critique during the formulation of the new algorithm.

The work on AMS is supported by the Deutsche Forschungsgemeinschaft (DFG), Bonn, FRG as part of the Sonderforschungsbereich 241 IMES, "Integrated Mechanical-Electronic Systems".

\footnote{engl. \approx special research initiative}
References


