A New Approach to Modifying Fuzzy ARTMAP Systems

Mu-Chun Su  
Department of Computer Science & Information Engineering, National Central University, Taiwan, R.O.C.  
muchun@csie.ncu.edu.tw

Wei-Zhe Lu  
Department of Computer Science & Information Engineering, National Central University, Taiwan, R.O.C.

Chen-Chiung Hsieh  
Laboratory, Institute for Information Industry, Taiwan, R.O.C.

Abstract: A fuzzy ARTMAP system is a system for incremental supervised learning of recognition categories and multidimensional maps in response to arbitrary sequence of analog or binary input vectors. The original fuzzy ARTMAP system incorporates two fuzzy ART modules and an inter-ART module. Many different approaches have been proposed to modify fuzzy ARTMAP systems. In this paper, we proposed a new approach to modifying a fuzzy ARTMAP system. We referred to the new system as the Modified and Simplified Fuzzy ARTMAP (MSFAM) system. Two data sets were used for demonstrating the performance of the proposed MSFAM systems.

Keyword: neural networks, learning algorithm, ART

I. Introduction

Recently, neural networks have been successfully used in many pattern recognition problems. Each class of neural networks has its own considerations. Multi-layer perceptions incorporated with the backpropagation algorithm are based on the minimization of the mean-squared error. Usually, they can take from hundreds to thousands to sometimes millions of epochs through the data set. On the contrary, the probabilistic neural networks require only one pass of learning [1]. Some neural networks focus on extracting either crisp rules [2]-[4] or fuzzy rules [5] directly from numerical input-output data for pattern recognition. The focus of nested generalized exemplar (NGE) theory is the ability to represent hierarchical relationships between data items through overlapping and nested hyperrectangles [6]. While many popular neural networks require a complete retraining of the networks with both the old and the new information whenever new information is added to the networks, the fuzzy min-max neural networks [7] and fuzzy ARTMAP systems [8] are able to learn new classes and refine existing classes quickly and without destroying old class information. This property is referred to as on-line learning and it is a very appealing property for an efficient pattern recognition system.

The first ARTMAP system was used to classify inputs with binary features in an incrementally supervised learning way [9]. The fuzzy ARTMAP system is a more general ARTMAP system that learns to classify inputs with analog features. Fuzzy ARTMAP systems have been benchmarked against a variety of machine learning, neural networks, and genetic algorithms with considerable success [10]. Owing to so many appealing properties, fuzzy ARTMAP systems provide a natural basis for many researchers.

The proposed system, referred to as an MSFAM system, is a new approach to modifying fuzzy ARTMAP systems. This paper is organized as follows. Section II gives a short description of the original fuzzy ARTMAP system. In section III, the detailed description of the proposed MSFAM system is given. Simulation results are given in section IV. Finally, the conclusions are given in section V.

II. Brief Review of Fuzzy ARTMAP Systems

Each fuzzy ARTMAP system includes a pair of fuzzy ART modules, ARTa and ARTb that are linked together via an inter-ART module, Fab, called a map field [8]. During supervised learning, ARTa receives an input in the complement code form \( I^c = A^c = (a^T, u^T) \), and ARTb receives an input, \( I^c = B^c = (b^T, v^T) \). Note that each component in \( I^c \) is in the interval \([0,1]\) and complement coding is a normalization rule that preserves amplitude information. The map field is used to form predictive associations between categories and to realize the match-tracking rule. It ensures autonomous system operation in real time and works by increasing the vigilance parameter \( \rho_a \) of ARTa. Parameter \( \rho_a \) calibrates the minimum confidence that ARTa must have in a recognition category, or hypothesis, activated by an input \( A \) in order for ARTa to accept that category, rather than search for a better one through an automatically controlled process of hypothesis testing. The lower the value of \( \rho_s \), the larger the number of categories is. A predictive failure of at ARTb increases \( \rho_s \) by the minimum amount needed to at ARTb, using a mechanism called match tracking. Hypothesis testing leads to the selection of a new ARTa category, which focuses attention on a new cluster of \( A \) that is better able to predict B. Owing to these mechanisms, fuzzy ARTMAP systems becomes one of a rapidly growing family of on-line learning pattern recognition systems.

Many researchers have proposed different modified models of fuzzy ARTMAP systems. One of the most appealing improvements is the simplified Fuzzy ARTMAP (SFAM) proposed by Kasuba [11]. The SFAM system is a vast simplification of fuzzy ARTMAP, which reduces the computational overhead and architectural redundancy of fuzzy ARTMAP without sacrificing recognition rates. An
SFAM system is essentially a two-larger neural network. The category layer is just a layer to hold the names of the J number of categories (or classes) that the network has to learn. All SFAM input values must be within the range 0 to 1.

III. Modified and Simplified Fuzzy ARTMAP Systems

In this paper, we propose an approach to not only simplifying but also modifying the original fuzzy ARTMAP system. The new system is referred to as the Modified and Simplified Fuzzy ARTMAP (MSFAM) system. Before we give a detailed description of the architecture of the MSFAM system, we first introduce a class of neurons with hyperrectangular neural-type junctions. Similar definitions of a hyperrectangular neuron can be found in [2]-[4]. The output function of a hyperrectangular neuron is computed as follows:

\[
Out_j(x) = \frac{(P_{max} - P_j(x))}{\alpha + (P_{max} - P_j)}
\]  

where

\[
P_{max} = \sum_{j=1}^{n} (M_j - m_j)
\]

\[
P_j = \sum_{j=1}^{n} (M_j - m_j)
\]

\[
P_j(x) = \sum_{j=1}^{n} \max(M_{j-x}, M_j - x_j)
\]

\[
M_j = \max(x_j)
\]

\[
m_j = \min(x_j)
\]

The vector, \(x = (x_1, \cdots, x_n)\), is an n-dimensional data pattern selected from a data set with total N patterns. The parameter \(\alpha\) is a pre-specified real-valued positive constant. The parameters \(M_j\) and \(m_j\) represent the adjustable weights of the \(j\)th hyperrectangular neuron and \(Out_j(x)\) is the output function of the neuron. Obviously, \(M_j\) and \(m_j\) represent the highest bound and the lowest bound of the data set at the \(j\)th dimension, respectively.

Fig. 1 The architecture of the proposed MSFAM system.

After we have introduced the definition of a hyperrectangular neuron, we may proceed to the introduction of the training algorithm for constructing an MSFAM system. The schematic view of the MSFAM architecture is shown in Fig. 1. While input vectors have to be normalized by the complement coder in fuzzy ARTMAP systems and SFAM systems. We can directly present raw data into an MSFAM system. The training algorithm starts with no committed hyperrectangular neuron and then incrementally adds neurons into the system to produce correct predictions.

The training algorithm is given as follows:

**Step 1:** Choose a baseline vigilance \(\rho\), specify the values of the parameters \(\Delta\rho\) and \(\alpha\), and initialize the index \(J\) to be zero. Compute the values of the parameters, \(M_j\) and \(m_j\), for \(i=1,\ldots,n\), from the training data.

**Step 2:** Present an input pattern \(x\) into the competition layer of the system. Suppose the class label of the input pattern is \(C_x\). Set the counter \(N_d\) and the maximum output \(Out_{max}\) to be zeros. Note that the symbol \(C_j^\dagger\) is used to denote the class label of neuron \(j\).

**Step 3:** If there is no committed neuron in the competitive layer then go to Step 9 to generate a new neuron to respond to the first data pattern \(x\). Otherwise, compute the outputs of the committed neurons in the competition layer using Eq. (1).

**Step 4:** Find the winner \(j^*\) by selecting the neuron with the largest output:

\[
Out_j(x) = \arg\max_{j=1,\ldots,J} Out_j(x)
\]

Here we assume the present number of neurons in the competition layer is \(J\). If all committed neurons were all disabled then go to Step 9, otherwise, go to Step 5.

**Step 5:** Check whether the output of the winner is larger than the vigilance value associated with the winning neuron:

\[
Out_j(x) \geq \rho_j^\dagger
\]

If this condition is met, go to Step 6. Otherwise, go to Step 9.

**Step 6:** Check whether the data pattern \(x\) is contained within the hyperrectangle, \(HR_j^\dagger\), defined by the winning neuron \(j^*\):

\[
m_{j^*} \leq x_i \leq M_{j^*} \quad \text{for} \quad 1 \leq i \leq n
\]

Go to Step 7.

**Step 7:**

Case 7.1: the data vector is contained within the hyperrectangle \(HR_j^\dagger\)

**Condition 7.1.1:** \(C_{j^*} = C_k\)

If \(N_d = 0\) then we do not need to update the weights of the winning neuron and can directly go to Step 2. Otherwise, we disable the winning neuron and go to Step 4.

**Condition 7.1.2:** \(C_{j^*} \neq C_k\)

We set \(\rho_j^\dagger = 1.0\) and \(N_d = N_d + 1\). Go to Step 9 to add a new neuron.

Case 7.2: the data vector is not contained within the hyperrectangle \(HR_j^\dagger\)

**Condition 7.2.1:** \(C_{j^*} = C_k\)

If \(\rho_j^\dagger = 1.0\), we disable the winning neuron and go to Step 4. If \(\rho_j^\dagger \neq 1.0\) and \(N_d = 0\) we go to Step 8.
to update the weights of the winning neuron. Otherwise, we tentatively use Eq. (10)-(11) to update the weights of the winning neuron and check whether the adjustment can make \( \text{Out}_j(\mathbf{x}) \) be greater than \( \text{Out}_{\text{max}} \). If yes, we go to Step 8 to really update the weights, otherwise, the original weights are restored and go to Step 4.

**Condition 7.2.2**: \( C'_j \neq \mathcal{C}_k \)

We slightly raise the value of the vigilance parameter by adding a small amount to it (e.g. \( \rho'_j = \min[1.0, \rho_j + \Delta \rho] \)). Set \( N_d = N_d + 1 \). If \( N_d = 1 \) we set \( \text{Out}_{\text{max}} \) to be \( \text{Out}_j(\mathbf{x}) \). Then we disable the winning neuron and go to Step 4.

**Step 8**: Update the winner’s weights by expanding the hyperrectangle defined by the weights of the winner to include the data \( \mathbf{x} \):

\[
M_{j,i} = \max(x_i, M_{j,i}) \quad \text{for} \quad 1 \leq i \leq n \quad (10)
\]

\[
m_{j,i} = \min(x_i, m_{j,i}) \quad \text{for} \quad 1 \leq i \leq n \quad (11)
\]

Go to Step 2.

**Step 9**: A new neuron is generated and initialized as follows:

\[
\rho_{j+1} = \rho + \Delta \rho \times N_d \quad (12)
\]

\[
M_{j+1,i} = x_i \quad \text{for} \quad 1 \leq i \leq n \quad (13)
\]

\[
m_{j+1,i} = x_i \quad \text{for} \quad 1 \leq j \leq n \quad (14)
\]

\[
C'_{j+1} = \mathcal{C}_k \quad (15)
\]

\[
J = J + 1 \quad (16)
\]

Go to Step 2 to learn next pattern.

**IV. Simulation Results**

We use two data sets, one artificial data set and one real data set, to test the performance of the proposed MSFAM system.

**Example 1: 2-D artificial data set**

Fig. 2(a) illustrates the 2-D artificial data set consisting of 579 data points that are labeled into 3 different classes. Table I tabulates the simulation results in the data set. The parameter \( \alpha \) was set to 0.00001 for all the simulations. Obviously, when \( \rho = 1.0 \), there are 579 neurons generated in the MSFAM system. That is, each neuron corresponds to a data point. As the baseline vigilance decreases, the number of neurons also decreases. Fig. 2(b) shows the 11 rectangles generated for the case \( \rho = 0.9 \). Fig. 2(c) shows the decision regions achieved by the constructed MSFAM system. Several observations can be made form the simulations:

1. Rectangles corresponding to different classes can overlap each other.
2. Small rectangles can be generated inside large rectangles in order to produce correct predictions for those data points lying inside the larger rectangles that correspond to different classes.
3. When a data point lies inside several rectangles corresponding to different classes the data point is finally assigned to the smallest rectangle and classified to be the class label associated with the smallest rectangle.

**Example 2: The Landsat data set**

This data set is taken from the ftp anonymous “UCI Repository of Machine Learning Database and Domain Theories” and was in use in the European statLog project [12]. It is for classification of the multi-spectral values of an image of the Landsat satellite. There are 6435 data patterns in the data set. Each data pattern contains the pixel values in four spectral bands of each of the 9 pixels in a 3 x 3 neighborhood and a number indicating the classification label of the central pixel. The classification labels correspond to 6 types of soil: (1) red soil, (2) cotton crop, (3) gray soil, (4) damp gray soil, (5) soil with vegetation stubble, and (6) very damp gray soil. We randomly partitioned the data set into a training set consisting of 80% of the data points and a testing set consisting of the remaining data patterns. The confusion matrix is given in Table II. The correct classification rates are 100% and 82.98% for the training set and the testing set, respectively. There are total 205 neurons generated in
V. Conclusions

In this paper we propose an MSFAM system. Several major differences between the proposed MSFAM systems and the original fuzzy ARTMAP systems or the SFAM systems can be summarized as follows:
1. In our MSFAM system, raw data can be directly presented to the system without the need of the complement coder.
2. The MSFAM system can use only one criterion to find winners while the original fuzzy ARTMAP system and the SFAM system need to use two criteria.
3. In our MSFAM system, each hyperrectangular neuron is associated with its own vigilance parameter.

Acknowledgement

This work was supported by the MOE Program for Promoting Academic Excellent of Universities under the grant number EX-91-E-FA06-4-4.

References:

<table>
<thead>
<tr>
<th>ρ</th>
<th>Number of neurons</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>579</td>
</tr>
<tr>
<td>0.99</td>
<td>314</td>
</tr>
<tr>
<td>0.95</td>
<td>29</td>
</tr>
<tr>
<td>0.90</td>
<td>11</td>
</tr>
<tr>
<td>0.80</td>
<td>5</td>
</tr>
<tr>
<td>0.70</td>
<td>5</td>
</tr>
<tr>
<td>0.60</td>
<td>5</td>
</tr>
<tr>
<td>0.50</td>
<td>5</td>
</tr>
<tr>
<td>0.40</td>
<td>5</td>
</tr>
<tr>
<td>0.30</td>
<td>5</td>
</tr>
</tbody>
</table>

Table II: The confusion matrix for the Landsat data set for the case of 205 neurons.