Using Fuzzy Filters as Feature Detectors

Chuen-Tsai Sun Tsuey-Yuh Shuai Guang-Liang Dai

Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 30050
E-mail: ctsun@cis.nctu.edu.tw

Abstract— **A neuro-fuzzy model of adaptive learning and feature detection is presented. The model, called the *fuzzy filtered neural network*, was first introduced in a previous publication, which showed its validity in the domain of plasma analysis. Here we extend the model to another problem, the recognition of hand-written numerals, to demonstrate its generality. We propose three versions of the architecture, which use one-dimensional fuzzy filters, two-dimensional fuzzy filters, and genetic-algorithm-based fuzzy filters, respectively, as feature detectors. All three versions smoothly handle such issues of a real-world pattern recognition problem as drifting and noise. Simulation results show that the proposed model is an efficient architecture for achieving high recognition accuracy.**

I. Introduction

The banner of fuzzy logic, as pointed out by its inventor, Professor Lotfi Zadeh, is to exploit the tolerance for imprecision [10]. Because high precision entails high cost and low tractability, reasonable solutions to problems encountered in daily life usually employ knowledge at a compromised degree of precision granularity. In this sense, feature detection can be defined as an effort to extract essential attributes from a massive amount of information so that a pattern recognition problem can be solved efficiently.

The importance of feature detection is also evident when we consider neural network architecture. The complexity and limitations of traditional neural networks are largely due to the lack of an effective way of extracting meaningful information from the learned configuration. This problem becomes more intractable when the number of physical sensors used for measurement increases. For example, an image to be processed contains thousands of pixels, which is much more than the number usually needed for pattern recognition or computer vision. In addition, the drifting of sensory equipment and variations in samples would cause any fixed-position feature detector to miss the resulting altered signal. To cope with these two factors together with the general problem of background noise, a form of signal filtering is needed. In this paper we introduce a mechanism for *fuzzy filtering* to cope with the complexity of feature extraction.

Fuzzy filtering is the task of partitioning a massive amount of physical data channels into a much smaller number of *fuzzy channels*. These channels, adaptable during a training process, are employed for both noise filtering and feature detection. In the proposed neuro-fuzzy model, system parameters such as the *membership functions* defined for each fuzzy channel and the weights in the feedforward network are calibrated with backward error propagation. We have successfully applied this mechanism in plasma analysis [9]. In this paper we will explore its validity in another important domain, recognition of hand-written numerals, to illustrate its generality in terms of feature detection.

If we follow the scientific paradigm of simulation, a positional feature detector should preferably behave as a *localized receptive field*, which is supported by biological evidence [5]. Mathematically, localized receptive fields can be represented as *radial basis functions*. We have shown that there exists a type of functional equivalence between radial basis function networks and fuzzy inference systems [2]. Furthermore, localized-receptive-field-based architectures are more efficient than standard neural networks in terms of learning [7]. The above background justifies the choice of using a fuzzy neural network as a feature extraction tool for problems with a massive amount of sensory input.

In the next section, we describe the basic architecture of fuzzy-filtered neural networks and their application to hand-written numeral recognition. Simulation results are presented to validate the use of fuzzy channels to identify important features. In Section III, we generalize the model so that it can handle two-dimensional inputs, such as those encountered in numeral recognition, more smoothly than before. In Section IV, the method is further extended to include a genetic algorithm (GA)-based mechanism, which makes it even more flexible. Some useful techniques, such as sample superimposition, are briefly discussed. The paper closes with concluding remarks and suggestions for future work.

II. Fuzzy-Filtered Neural Networks

A *fuzzy filtering* mechanism assumes that the boundary between two neighboring *meaningful* channels is a continuous, overlapping area in which a physical channel has partial membership in both fuzzy channels. A fuzzy channel defines a range of input signal intensity characterized by an

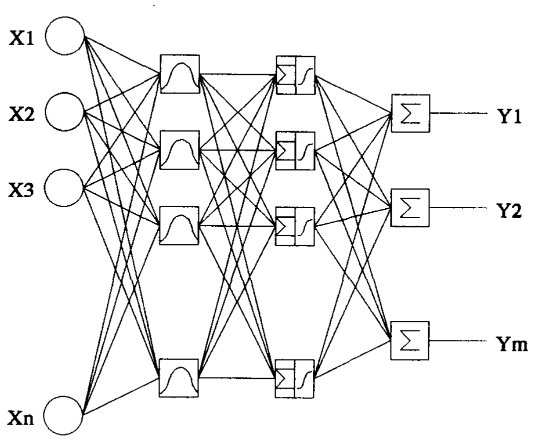

Figure 1: *Fuzzy Filtered Neural Network.*

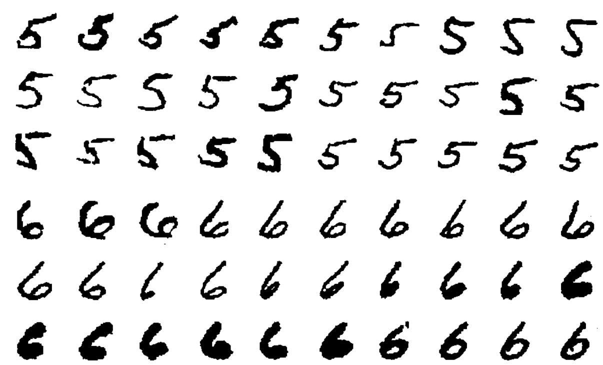

Figure 2: *Testing examples.* Some of the 5's and 6's in the testing set.

appropriate *membership function* [11]. The position and shape of this membership function are adjusted through a learning process so that the system error is minimized. At the end of training, these fuzzy channels are expected to hook onto salient physical channels so as to provide a meaningful interpretation of the qualitative aspects of patterns.

To implement this idea, we use a multi-layer feed-forward *adaptive network* in which each node performs a particular function (*node function*) in response to incoming signals and a set of parameters pertaining to that node. Figure 1 depicts a fuzzy-filtered adaptive network, in which the x_i's are inputs and the y_j's are outputs. Nodes in the same layer have the same type of node function.

Layer 1 is the input layer. Each node in *Layer 2* is associated with a parameterized bell-shaped membership function represented by

$$\mu_A(x_i) = \frac{1}{1 + [(\frac{x_i - c_i}{a_i})^2]^{b_i}}, \qquad (1)$$

where x_i is one of the input variables, A is the linguistic term associated with the node function, and $\{a_i,\ b_i,\ c_i\}$ is the parameter set. The node output is a normalized weighted sum:

$$\frac{\sum_i \mu_A(x_i) f(x_i)}{\sum_i \mu_A(x_i)}, \qquad (2)$$

where $f(x_i)$ is the intensity of input channel x_i.

The initial values of the parameters are set in such a way that the membership functions satisfy ϵ *completeness* [4] (ϵ = 0.5 in our case), *normality*, and *convexity* [3]. Although these initial membership functions are set heuristically and subjectively, they do provide an easy interpretation that parallels human thought processes. The parameters are then tuned using backpropagation in a learning process based on the training data set.

Each node in *Layer 3* performs in the same way as a node in a standard network: it takes the weighted sum of inputs and produces a transferred output through a sigmoidal function. *Layer 4* is similar, except that the transfer function is neglected because our output values are usually pre-scaled.

To verify the proposed model, we applied it to a handwritten numeral recognition problem. We used the f13 numeral data set in the public domain as a benchmark. This set contains 3,471 numerals from 49 different writers. The numerals have been spatially normalized to a 32×32 frame of pixels. We used 3,000 numerals for training and the remaining 471 for testing. In this case most of the testing numerals were written by persons other than those who wrote the numerals in the training set. We believe that this arrangement closely simulates real applications, such as zip-code recognition. Some examples from the testing set are shown in Figure 2.

For comparison, we initially tested the data set on a standard three-layer (1024-30-10) feedforward neural network. The network examined the 3000 training numerals and used backpropagation to adjust the weights. After 250 training epochs, we tested the network using the remaining 471 numerals. The recognition rate was 85%. Moreover, because of the network's complicated architecture, its learning efficiency was poor.

To use the fuzzy filtering mechanism proposed above, we have to transform the numerals to spectrum-like signals by taking their projection on both the x-axis and y-axis. Hence, each 32×32 image was transformed into a 64-channel spectrum, with each channel representing the number of black pixels in a column or row of the image. A 64-20-10-10 fuzzy filtered network was used to learn the training patterns. The recognition rate was 90%, a better result than that yielded by the pure neural network.

The proposed fuzzy filtering mechanism also simplifies the neural network architecture, because far fewer system parameters need to be adjusted. For instance, the 1024-30-10 network mentioned above has $1024 \times 30 + 30 \times 10 = 31020$ weights to fine-tune; in contrast, the 64-20-10-10 fuzzy filtered network has only $3 \times 20 + 20 \times 10 + 10 \times 10 = 360$

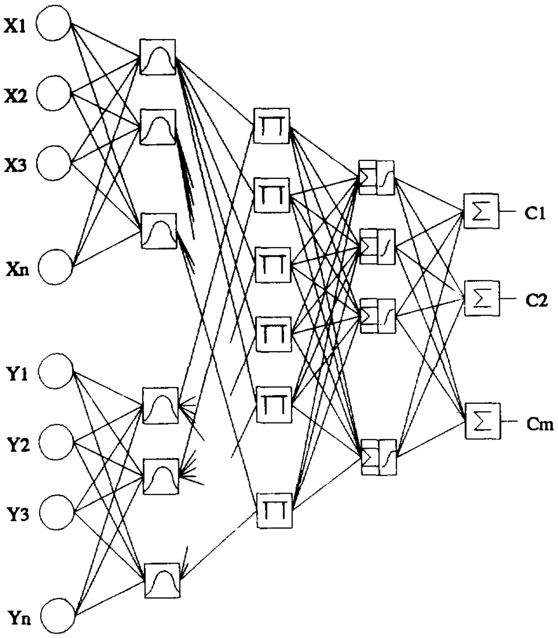

Figure 3: *A fuzzy-filtered neural network for plasma analysis.*

parameters. This simplicity increases learning efficiency. Another important point is that the fuzzy approach provides a meaningful interpretation of the training results, which helps us to better understand the salient features in a pattern recognition problem.

III. Two-Dimensional Fuzzy Filters

Image patterns are different in nature from the one-dimensional plasma spectra that were the subject of our earlier study. To deal with image patterns, we need to develop fuzzy filters to handle two-dimensional data. Thus, we can generalize the proposed model to a 2-D version, which is shown in Figure 3. In this architecture, x-membership and y-membership are integrated by a t-norm operator, e.g., multiplication. The combined membership is then used as the filtering weight.

The initial distribution of membership functions is represented by the grid in Figure 4(a). The training process produces a picture like that shown in Figure 4(b) to capture important pieces of information.

We used 36 two-dimensional fuzzy filters in the experiment. The recognition rate was increased to 92%, compared to 90% with the 40 one-dimensional filters described in the previous section. The resulting contour of the adjusted fuzzy filters is shown in Figure 5.

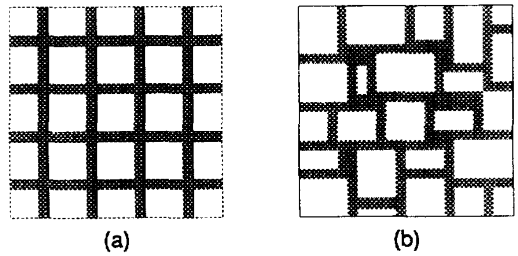

Figure 4: *2-D Fuzzy Filters.* (a) Before training, (b) After training.

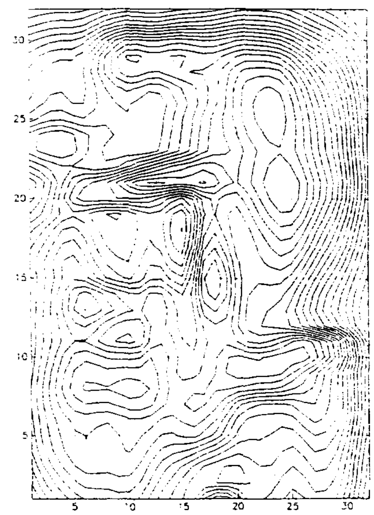

Figure 5: *Contour of Fuzzy Filters.*

IV. GA-Based Fuzzy Channels

Backpropagation, the gradient descent method employed for adapting weights in the last two sections, occasionally suffers from the problem of local optimization. Consequently, in this section we further extend the proposed model into an even more flexible version that employs genetic algorithms.

Genetic algorithms (GAs) have been used in classification problems because of their ability to identify the weights of importance among features. GAs have been employed for feature selection in hybrid models with K-nearest neighbor algorithms [8], and with feature partitioning [1].

For pattern recognition problems, the important positional features are typically those having the highest intensity, and these features change dramatically with different numerals. As in the previous sections, we want to use only this type of low-level, positional feature because for the machine to learn in a realistic sense, high-level human

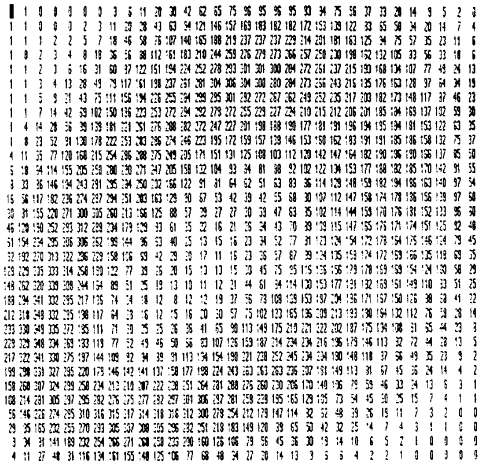

Figure 6: *Superimposed Image.* Each position is denoted by a value that represents the superimposed intensity on that position.

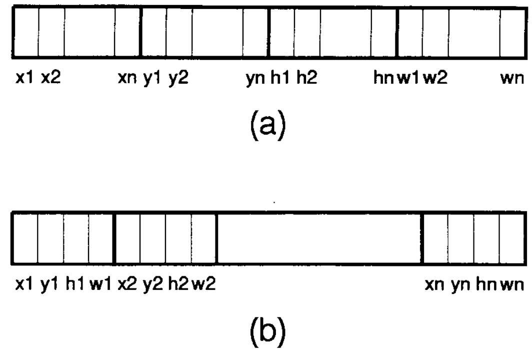

Figure 7: *Gene Encoding Schemes.* (a) All x coordinates of the lower-left corners of the rectangles are grouped together, then the y coordinates, the heights, and the widths. (b) The four features of a rectangle, its x coordinate, y coordinate, height, and width, are grouped together.

involvement should be avoided.

We use a GA to determine ten rectangular regions for each digit. The selection procedure of the GA is guided by the following evaluation formula:

$$\sum_{numerals} \sum_{regions} \frac{b(region)}{s(region) \times z}, \qquad (3)$$

where $b(region)$ is the number of black pixels in a region and $s(region)$ is the size of that region. Because we do not want rectangles to nearly overlap one another, we introduce the denominator term z, which is defined as one plus the total number of overlapped pixels of the rectangles represented by a chromosome string. In other words, the higher the degree of overlapping, the lower the evaluation score. This approach to GA evaluation dramatically improves the training efficiency of GAs, which have traditionally been slow, because we can use superimposed numeral images instead of feeding the GA individual numerals one by one. Figure 6 shows the superimposition of all the 0 digits in our training set.

We tried two gene encoding schemes for this problem, as shown in Figure 7. In the first scheme we grouped together all x coordinates of the lower-left corners of the rectangles as the first section of the chromosome. The sections of y coordinates, heights, and widths follow, in that order. In the second scheme we took the four features of a rectangle as one unit. We found that the first encoding scheme yielded better performance. A possible explanation for this is that the first encoding scheme better fits the one-point crossover mechanism that we used.

When the GA converged, it gave us ten rectangular positional feature detectors for each digit, as shown in Figure 8. We then fuzzified the rectangles by using two Gaussian functions to approximate the height and the width of each area. In other words, we created a membership function with its mean at the middle point of one side and its deviation equal to half of the height or the width of a certain rectangle. Finally, we placed this GA-generated fuzzy feature detector in front of a standard neural network for classification training. The mechanism of this integrated GA-fuzzy-neural approach is shown in Figure 9.

Figure 8: *Rectangular Feature Detectors.* Upper, left to right: 0, 1, 2, 3, 4; lower, left to right: 5, 6, 7, 8, 9.

We found that the above architecture gave us the best performance of all of the architectures we studied. Its recognition rate was 95.7%; in other words, of the 471 numerals in the testing set, only 20 were misclassified. In the past, this level of performance has only been achieved by using a great deal of human knowledge, as discussed in [6].

V. Concluding Remarks

We have investigated the use of fuzzy filtered neural net-

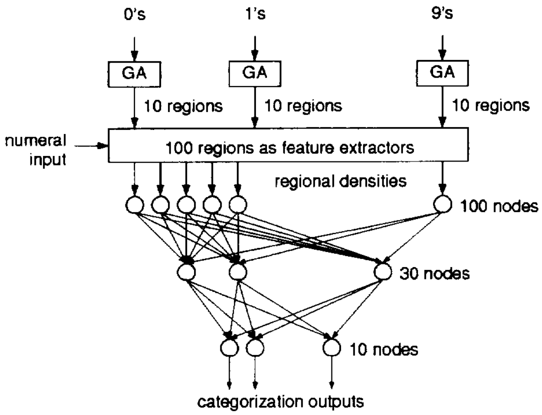

Figure 9: *GA-Determined Fuzzy-Filtered Neural Network.*

works to identify important positional features for recognition of hand-written numerals. We have explored three different architectures: one-dimensional fuzzy filters, two-dimensional fuzzy filters, and GA-based fuzzy filters. The neuro-fuzzy model proposed in this paper has not only the capability to learn and adapt to changes in operating conditions, but also the advantage of pertinent feature identification.

Experiments on a large-scale data set substantiated the effectiveness of fuzzy channels. The location and shape of membership functions were used to identify positional features for complicated pattern recognition problems with a very low level of human involvement. This helped give us an idea of what kind of features the machine could use to simulate human decision making. Once the features were identified, neural network learning became much easier and more efficient.

This technique can also be used for pattern analysis in other fields, such as medical diagnosis or global change, in which the explanation of the detected features plays a role as important as the applicability of the working model. In brief, complicated patterns are not self-explanatory; they require a good deal of interpretation. Fuzzy filtered neural networks successfully serve this purpose.

In summary, the network described here actually learns on its own from the training data and selects the most pertinent positional features without any high-level guidance from human experts. This ability to detect features is the primary advantage of the fuzzy filtering mechanism. This capability is significant, because human experts can actually learn from the network and gain a better understanding of the problem it treats so as to produce better remedies.

References

[1] H. Altay Güvenir and İzzet Şirin. A genetic algorithm for classification by feature partitioning. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 543–548, 1993.

[2] Jyh-Shing Jang and Chuen-Tsai Sun. Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Trans. on Neural Networks*, 4(1):156–159, 1993.

[3] Arnold Kaufmann and Madan M. Gupta. *Introduction to Fuzzy Arithmetic: Theory and Applications.* Van Nostrand Reinhold Co., 1985.

[4] C.C. Lee. Fuzzy logic in control systems: Fuzzy logic controller. *IEEE Trans. on Systems, Man, and Cybernetics*, 20(2):404–435, 1990.

[5] John Moody and Christian Darken. Learning with localized receptive fields. Technical Report YALEU/DCS/RR-649, Department of Computer Science, Yale University, 1988.

[6] Christine Nadal and Ching Y. Suen. Applying human knowledge to improve machine recognition of confusing handwritten numerals. *Pattern Recognition*, 26(3):381–389, 1993.

[7] Stephen M. Omohundro. Geometric learning algorithms. Technical Report TR-89-041, International Computer Science Institute, 1989.

[8] W.F. Punch, E.D. Goodman, Min Pei, Lai Chia-Shun, P. Hovland, and R. Enbody. Further research on feature selection and classification using genetic algorithms. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 557–564, 1993.

[9] Chuen-Tsai Sun, Jyh-Shing Jang, and Chi Jung Fu. Neural network analysis of plasma spectra. In *Proceedings of the 1993 International Conference on Artificial Neural Networks*, pages 968–972, 1993.

[10] Lotfi A. Zadeh. Unpublished lectures. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

[11] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.