Analyzing Images Containing Multiple Sparse Patterns with Neural Networks

Rangachari Anand, Kishan Mehrotra, Chilukuri K. Mohan and Sanjay Ranka

School of Computer and Information Science
Syracuse University, Syracuse, NY 13244-4100

Abstract

We have addressed the problem of analyzing images containing multiple sparse overlapped patterns. This problem arises naturally when analyzing the composition of organic macromolecules using data gathered from their NMR spectra. Using a neural network approach, we have obtained excellent results in using NMR data to analyze the presence of various amino acids in protein molecules. We have achieved high correct classification percentages (about 87%) for images containing as many as five substantially distorted overlapping patterns.

1 Introduction

Currently known image analysis methods are not very effective when applied to images containing large multiple overlapped sparse patterns. Such patterns consist of a small number of features dispersed widely in the image. The features are usually small in size: possibly no larger than a single pixel. Such a classification problem is encountered when analyzing images obtained by certain types of Nuclear Magnetic Resonance (NMR) spectroscopy.

Neural networks offer potentially promising techniques for such problems, but few successful results have been reported in the literature on the application of neural networks to such complex image analysis tasks. One possible approach is to use Strong and Whitehead's physiological model [10] which describes how humans can sequentially focus on each pattern contained in a complex image. Their model is a discrete-event simulation of activities within human neurons. Due to the complexity of human neurons this model has only been tested with small input images.

The selective-attention neural network of Fukushima presents another approach for classifying overlapped patterns [3]. The main problem in applying Fukushima's approach for large images is the huge size of the required network. As many as 41000 cells are needed for classifying patterns in a 19×19 image. Since practical applications require processing considerably larger (256×256) images, the computational requirements using Fukushima's model are too high.

We have developed a modular analyzer for the problem of analyzing images containing multiple sparse patterns. Each module detects the presence of patterns that belong to one class in the input image. Each module has two stages. The first stage is a feature detector based on clustering [5]. For each class of patterns, cluster analysis is used to identify those regions of the input image where features of the patterns belonging to that class are most likely to be found. The second stage of each module is a backpropagation-trained feed-forward neural network [9] that performs the tasks of thresholding and classification. With this approach, we have been able to obtain very high correct classification performance (87%) on 256×256 images with noisy test data.

In the next section, we discuss the problem of analyzing multiple sparse patterns, describe some details of the NMR analysis problem, and discuss previous work on this topic. In section 3, we describe details of our system. Experiments and results are presented in section 4. Section 5 contains concluding remarks.

2 The problem

The images we analyze may contain many different 'patterns'. Each pattern consists of several 'features'. A feature may be a group of neighboring pixels, or perhaps just a single pixel. The locations of pixels may vary within a range determined by the feature. Hence the pattern-matching process has to allow for variability of pixel locations.

Figure 1 shows three images, each containing one pattern (of the same class) which consists of three features. Each feature consists of a single pixel (indicated by a '+' symbol), which must occur somewhere within a known region (delineated by dashed ellipses in the figure).

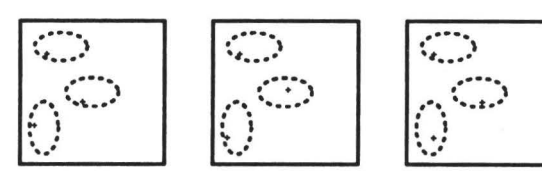

Figure 1: Three sparse patterns which belong to the same class.

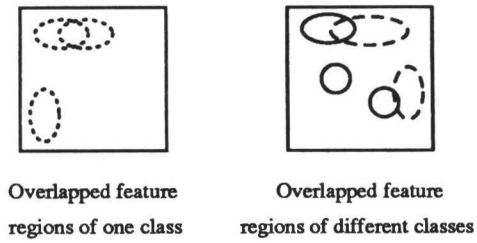

Figure 2: Overlap of feature regions

In the applications that we are interested in, feature-regions for different classes *do* overlap, as shown in Figure 2. Consequently, a feature may lie within feature-regions of several classes. Such a feature partially constrains the classification, although it does not permit us to decide unambiguously whether a particular class of patterns is present in the image. As noted by Rumelhart and McClelland [9], such problems are ideal candidates for neural network solutions.

A particular instance of this problem arises in the classification of NMR spectra. NMR spectroscopy is a powerful method for the determination of the three-dimensional structure of complex organic macromolecules such as proteins [11]. Proteins are long chains of smaller molecules called amino acids. Approximately 18 different types of amino acids are commonly found in proteins. The first step in analyzing the structure of a protein is to determine its constituent amino acids. One type of NMR spectroscopy used for this purpose is called Correlational Spectroscopy ('COSY').

The COSY spectrum of a protein is the result of the combination of the spectra of its constituent amino acids. The task of determining the constituent amino acids of a protein is therefore equivalent to the task of analyzing an image containing multiple sparse patterns. The training set for our analyzer consists of a number of sample spectra for each type of amino acid. These spectra were generated from information about the distributions of peaks for each type of amino acid, tabulated in [4].

2.1 Definitions

Image representation: An input image is a two-dimensional array of non-negative integers called 'intensities'. We will represent an image by a set $P = \{P_1, P_2, \ldots, P_N\}$ of triples, where each triple $P_i = (P_{i,x}, P_{i,y}, P_{i,z})$ for $1 \leq i \leq N$. The first two components $(P_{i,x}, P_{i,y})$ of each triple identify the location of a non-zero element in the input image, while the third $(P_{i,z})$ represents the intensity of that element. Chemists refer to each such triple as a *peak*.

An image P may contain several patterns (disjoint subsets of P). Each pattern is a collection of peaks associated with a certain amino acid class. The number of patterns contained in an input image is not known *a priori*. Hence each image contains an unknown number of peaks (N).

Pattern Description: In some cases, the same class may be identified by one of many different images. For instance, an amino acid c may give rise to $t_{(c,1)}$ or $t_{(c,2)}$,

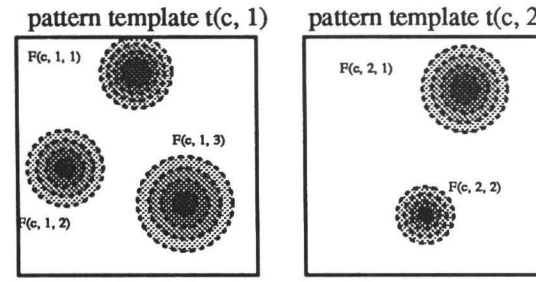

Figure 3: A class of patterns with two pattern templates.

which are two different configurations. Therefore, we define a set T_c of *pattern-templates* for each class c, where each pattern-template $t_{(c,i)}$ characterizes one configuration:

$$T_c = \{t_{(c,1)}, t_{(c,2)}, \ldots, t_{(c,M_c)}\}.$$

Each pattern-template is a set of *feature-templates*:

$$t_{(c,j)} = \{F_{(c,j,1)}, F_{(c,j,2)}, \ldots, F_{(c,j,S_{(c,j)})}\}.$$

A feature-template $F_{(c,j,k)}$ contains a complete specification for a feature which could occur in a pattern belonging to class c. Feature-templates determine which features (peaks) are present in an input image:

$$F_{(c,j,k)} = (\mathbf{r}_{(c,j,k)}, \lambda_{(c,j,k)}),$$

where $\mathbf{r}_{(c,j,k)}$ is the center of a feature region and $\lambda_{(c,j,k)}$ is used to define how far the feature region extends around the center. As described in section 3, we obtain the values of $\mathbf{r}$ by cluster analysis and implicitly compute the values of λ when a neural network is trained.

2.2 Classification procedure

In this section, we describe our procedure for analyzing images with multiple patterns from C classes.

Matching a feature-template: This is the first step in pattern recognition. We must determine whether a peak in the input image matches a feature-template.

We say that a peak P_i 'matches' a feature-template $F_{(c,j,k)}$ if:

$$\frac{g(|\mathbf{r}_{(c,j,k)} - (P_{i,x}, P_{i,y})|)}{P_{i,z}} < \lambda_{(c,j,k)}$$

where g, the 'error function', is chosen to increase with distance $|\mathbf{r} - (P_{i,x}, P_{i,y})|$. Peaks with high intensity values $(P_{i,z})$ match a feature template even when they are positioned far from the location of the feature template $(\mathbf{r}_{(c,j,k)})$. This is the reason for depicting feature templates with varying grey levels in Figure 3.

Matching a pattern-template: We say that an input image P 'matches' a pattern-template $t_{(c,j)}$ if for each feature-template $F_{(c,j,k)} \in t_{(c,j)}$, there exists a unique peak $P_i \in P$, such that P_i matches $F_{(c,j,k)}$.

Classification: If an input image P matches a pattern-template $t_{(c,j)}$, a pattern of class c is defined to be

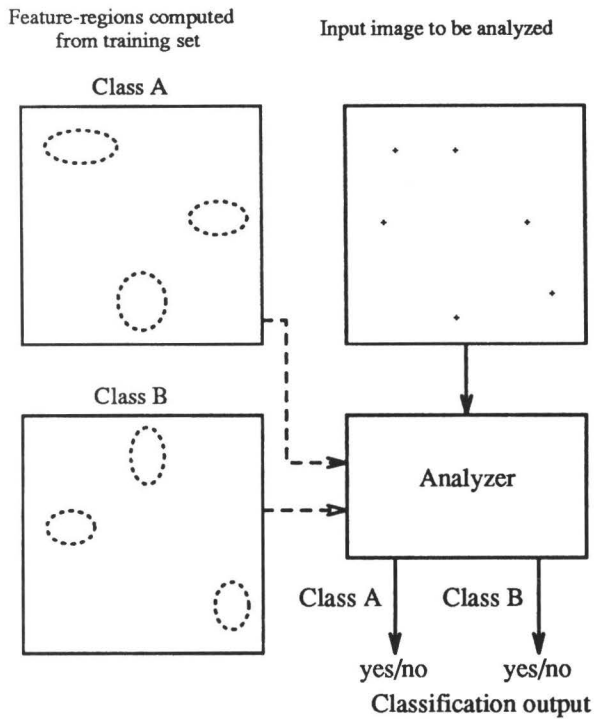

Figure 4: Overview of the classification process.

present in the input image. The overall analysis task is to determine all the classes whose features are present in the input image, hence the above procedure is repeated for every class.

An overview of the classification process is depicted in Figure 4. In the example shown, the feature-regions for two classes are known to the analyzer. Using this knowledge, the analyzer can determine whether a pattern of either class is present in the input image.

3 System description

In this section, we describe a modular analyzer for the analysis of images containing multiple sparse patterns. An important aspect of our system is the use of a clustering algorithm to train feature detectors for each class. Our use of the term 'feature detection' to identify *spatial* features in the patterns is to be distinguished from statistical parlance, where 'feature detection' may refer to the process of choosing the best set of variables to characterize a set of items [1].

3.1 Overview of the analyzer

A block diagram of the analyzer is shown in Figure 5. Each module detects the presence of one class of patterns. The modules work in parallel on an input image (presented as a list of peaks).

Each module consists of two stages. The first stage, called a *clustering filter*, transforms the image into a 'feature vector'. The second stage is a perceptron-like feed-forward neural network. The clustering filter computes the values of the matching functions, while thresholding is done by the neural network.

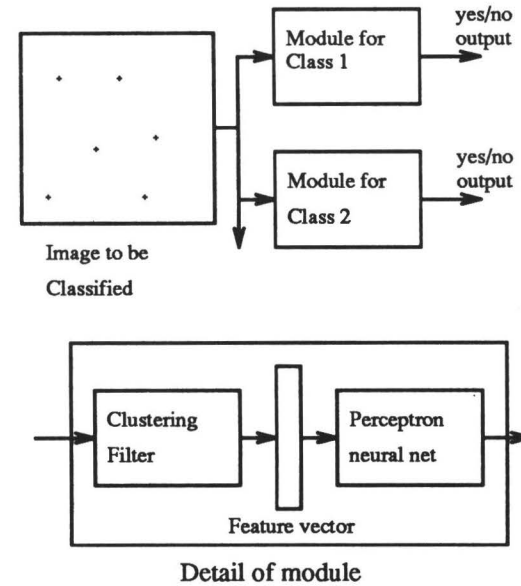

Figure 5: Block diagram of part of the modular analyzer.

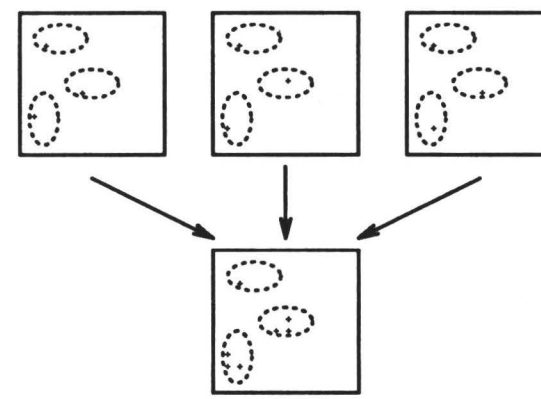

Figure 6: Use of clustering to find the expected locations of features.

3.2 Clustering

In machine vision systems, clustering is often used for image segmentation [8]. In our system, clustering is used to find the expected locations of features. We illustrate this with an example. Let the training set for some class c consist of the three images in Figure 1. Each image contains one pattern, which is known to belong to class c. These images have been superimposed in Figure 6. Clearly, the features occur in three clusters. The center of each cluster is the expected location of a feature.

The procedure may be summarized thus: for each class c, create a set R_c containing the locations of all features in an image created by superimposing all the training set images for class c. By applying a clustering algorithm to R_c, we determine the expected location of each feature, i.e., the cluster center.

We have investigated two clustering algorithms: the K-means clustering algorithm [5] and the LVQ (Learning Vector Quantizer) [6]. We have found that the LVQ performs better for our problem. A benchmarking study

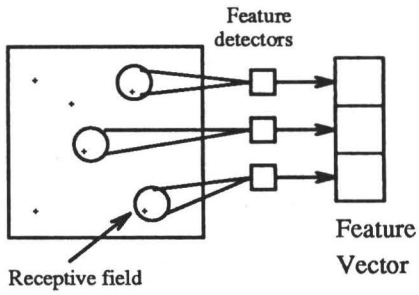

Figure 7: Clustering filter

by Kohonen et al. [7] also found that the LVQ produced better results than K-means clustering on several classification tasks.

3.3 The clustering filter

The role of each clustering filter (shown in Figure 7) is to extract relevant information from the input image for one class of patterns. A clustering filter consists of a number of *feature detectors*. A feature detector is a processing unit activated by the presence of a feature (peak) in its *receptive field*, a specific region of the input image. The output of a feature detector is a real value which depends on the number of peaks present within its receptive field, the intensities of those peaks and their distances from the center of the receptive field. The output of a clustering filter is the 'feature vector', each element of which is the (real-valued) output from one feature detector in the filter.

In a filter for class c, the receptive fields of the feature detectors should coincide with the feature regions of class c. For simplicity, we use feature detectors with *fixed* size receptive fields in our system. Consequently, if a feature region is larger than the receptive field, several feature detectors are required to cover it. We use the LVQ learning procedure to determine the position of each feature detector.

The output from a feature detector located at coordinate $\mathbf{r}$, when presented with an image $P = \{P_1, P_2, \ldots, P_N\}$ is:

$$Output(\mathbf{r}) = \sum_{i=1}^{N} \frac{P_{i,z}}{g(|\mathbf{r} - (P_{i,x}, P_{i,y})|)}.$$

The kernel function chosen is $g(x) = \exp^{\tau x^2}$ where τ is a constant between 0.1 and 0.5. Although all peaks in the image are fed to the feature detector, it will actually respond only to those peaks which are very close to $\mathbf{r}$. This is because for the values of τ that we have chosen, the reciprocal of the kernel function $g(x)$ drops to almost zero when $x > 6$. Therefore the feature detector only responds to peaks which lie within a radius of 6 pixels around its center, $\mathbf{r}$.

As we noted previously, there are cases where two peaks sometimes occur very close together. We do not need to make any special provision for this situation. We use only one set of feature detectors to cover the *combined* feature region. The output from these feature detectors will be higher, but this is easily handled by the neural network.

3.4 The neural network

Although a clustering filter is trained to respond most strongly to patterns of a particular class, it is possible (due to overlap of feature-regions) that some of the detectors of one class may be activated when patterns of another class are presented. We use a feed-forward neural network to determine implicitly the appropriate thresholds for each pattern detector. This neural network is trained after the clustering filters of the first stage have been trained and set up.

For each class c, the neural network (of the corresponding module) must be taught to discriminate between feature vectors obtained from images containing a pattern of class c and feature vectors produced from images which do not contain patterns of class c.

We use backpropagation [9] to train the network. Backpropagation is a supervised learning algorithm in which a set of patterns to be learnt are repeatedly presented to the network together with their target output patterns. At the outset of a training session, the weights and biases are randomly initialized. For each pattern presented, the error backpropagation rule defines a correction to the weights and thresholds to minimize the square sum of the differences between the target and the actual outputs. The learning process is repeated until the average difference between the target and the actual output falls below an operator-specified level.

4 Results

In this section, we describe our experiments in training and testing the sparse image recognition system, and report the results obtained.

4.1 System parameters

To substantiate our approach, seven modules were trained for the NMR protein analysis problem. Each module can detect patterns corresponding to one amino acid. The final output from each module is a yes/no answer about whether the respective class (amino acid) is judged to be present.

From among 18 possible amino acids, we trained modules for seven amino acids whose spectra appeared to be the most complex, with more peaks than the others. But in training as well as testing the modules, we used data which included peaks from the other 11 amino acids as well, and obtained good results in analyzing the presence of the seven amino acids for which modules were trained. This shows that an incremental approach is possible: upon building modules for the other 11 amino acids, we expect that our results will continue to hold.

Table 1a lists the parameters of each module. The names of the amino acids along with their one-letter codes are listed in the first column. The second column is the number of feature detectors in the first stage of each module. The third column shows the number of hidden layer nodes in the neural network which comprises the second stage of each module. These were approximately the smallest number of nodes required for convergence

of the network training procedure, obtained by experimenting with various values.

Module	Number of detectors	Hidden layer size
Alanine (a)	4	10
Cystine (c)	10	20
Aspartic acid (d)	19	20
Glutamic acid (e)	14	10
Phenylalanine (f)	26	10
Isoleucine (i)	19	20
Valine (v)	15	20

Table 1a: Module Parameters.

The training set consists of a total of 90 single-class images, with 5 for each of the 18 amino acids. The equation indicating how weights are changed using the error back-propagation procedure [9] is:

$$\Delta w_{ji}(n+1) = \eta \delta_{pj} o_{pj} + \alpha \Delta w_{ji}(n).$$

In each module, we used a value of $\eta = 0.1$ for the learning rate parameter, and a value of $\alpha = 0.05$ for the momentum coefficient. The target mean squared error (to terminate the network training procedure) was set to be 0.01. During training, the target output for the networks was set to be 0.1 when the required answer was 'no' and 0.9 when the required answer was 'yes'. Weights in the network were updated only at the end of each 'epoch' (one sequence of presentations of all training inputs). Table 1b shows the number of epochs needed to train the LVQ's and the feedforward (backpropagation) neural networks, for each module.

Module	LVQ	Backpropagation
a	10000	717
c	10000	2653
d	20000	452
e	20000	149
f	30000	3128
i	20000	503
v	20000	328

Table 1b: Number of epochs required for training.

To investigate the effect of varying the receptive field sizes of detectors, we trained two versions of each module with different values for τ, the constant in the kernel function. When $\tau = 0.5$, detectors have small receptive fields, whereas when $\tau = 0.1$, detectors have large receptive fields.

4.2 Experimental results

The goal of the experiments was to measure the correctness of overall classification when the system was presented with composite images containing several patterns of different classes. Various experiments were performed to test our sparse image analysis system on composite images consisting of:
(i) different numbers of patterns;
(ii) with and without perturbations; and
(iii) for detectors with different receptive fields ($\tau = 0.5$ and $\tau = 0.1$).

To illustrate the testing method, consider a composite image created by superimposing two images with patterns that belong to classes *c* and *d*. Correct classification implies that this image should be classified 'NO' by modules *a, e, i, f* and *v* and 'YES' by modules *c* and *d*. We measure the percentages of correct classification, testing the modules in this manner on various composite images.

In the *first* set of experiments, we generated 1000 examples of each case: composite images containing 2, 3, 4 and 5 patterns respectively. In each set, the composite images were created by superimposing a randomly chosen set of images (each of a different class) drawn from the training set. The percentages of these images correctly classified by each module under different conditions are reported in table 2, for $\tau = 0.5$ and $\tau = 0.1$.

From table 2, it is clear that error rates increase with the number of images (patterns) in the input image. This is because the receptive fields of different classes of patterns overlap. Hence patterns of one class may partially activate feature detectors for other classes. As the number of patterns in the input image increases, it becomes increasingly likely that a feature detector may respond to artifacts arising from a fortuitous combination of patterns belonging to other classes. This problem is further aggravated by increasing the size of the receptive fields, as shown in table 2.

Module	$\tau = 0.5$				$\tau = 0.1$			
	No. of patterns in image				No. of patterns in image			
	2	3	4	5	2	3	4	5
a	100	100	100	100	99.1	99.3	98.9	97.7
c	100	99.8	99.9	99.7	100	100	100	100
d	100	100	99.8	99.5	100	99.9	99.1	98.5
e	100	100	100	100	96.4	94.4	90.3	86.3
f	99.9	99.5	99.3	98.5	99.3	95.1	91.0	86.4
i	98.0	96.4	96.4	95.7	99.9	99.6	99.6	99.3
v	100	100	100	99.6	99.4	99.2	98.7	94.4

Table 2: Percentages correctly classified, when test images were random combinations of training set images.

Module	$\tau = 0.5$				$\tau = 0.1$			
	No. of patterns in image				No. of patterns in image			
	2	3	4	5	2	3	4	5
a	98.0	97.8	97.8	95.8	99.7	99.6	98.6	96.6
c	97.1	95.0	95.0	94.0	99.7	99.3	99.0	99.2
d	98.6	97.1	95.5	94.8	100	99.6	99.4	98.2
e	99.4	99.2	98.6	98.3	97.0	95.0	88.6	87.7
f	96.1	93.2	93.6	90.1	98.9	97.0	89.8	84.6
i	96.4	94.6	93.7	92.6	99.7	99.6	99.7	99.1
v	98.3	97.0	96.4	95.1	99.8	99.4	96.7	95.5

Table 3: Percentages correctly classified, with low noise. Test images were random combinations of training set images with peak locations randomly translated by [-1, +1].

Module	$\tau = 0.5$				$\tau = 0.1$			
	No. of patterns in image				No. of patterns in image			
	2	3	4	5	2	3	4	5
a	92.0	85.8	82.5	80.7	94.6	90.9	86.8	86.1
c	94.0	91.0	89.0	83.7	95.3	94.0	92.7	89.7
d	89.2	85.7	79.0	76.4	95.2	94.6	92.0	89.8
e	89.4	83.8	79.0	76.2	96.6	95.2	91.5	86.8
f	87.4	82.2	75.8	70.6	94.4	89.8	84.8	76.8
i	89.8	82.8	77.5	74.4	95.5	93.6	93.2	89.7
v	90.0	85.8	80.8	75.5	94.1	91.3	87.7	88.6

Table 4: Percentages correctly classified, with high noise. Test images were random combinations of training set images with peak locations randomly translated by [-3, +3].

It is desirable to perform correct classification even in the presence of small errors or corrupted data. Hence,

we tested our system with composite images produced by superimposing *distorted* versions of the training set images to the system.

Two series of experiments were performed, varying the amount of distortion in the test data, for small and large receptive fields ($\tau = 0.5$ and $\tau = 0.1$). In one set of experiments, distortions were introduced by adding a random integer in the range [-1, +1] to the coordinates of the peaks. The results of these are summarized in table 3. In another set of experiments, the distortion was increased by adding random integers in the range [-3, +3] to the coordinates of peaks. These results are summarized in table 4.

With the small receptive field system ($\tau = 0.5$), the combined effect of distortion and multiple patterns causes classification accuracy to deteriorate substantially. On the other hand, classification capabilities of the large receptive field system ($\tau = 0.1$) are less affected and degrade more gracefully with noise. This phenomenon may be contrasted with the observation that the small receptive field system performs marginally better on uncorrupted test data.

5 Concluding Remarks

In this paper, we have addressed the problem of analyzing images containing multiple sparse overlapped patterns. This problem arises naturally when analyzing the composition of organic macromolecules using data gathered from their NMR spectra. Using a neural network approach, we have obtained excellent results in analyzing the presence of various amino acids in protein molecules. We have achieved high correct classification percentages (about 87%) for images containing as many as five substantially distorted overlapping patterns.

The architecture of our system is modular: each module analyzes the input image and delivers a yes/no output regarding the presence of one class of patterns in the image. Each module contains two stages: a clustering filter, and a feedforward neural network. An unconventional aspect of our approach is the use of clustering to detect *spatial* features of patterns.

We performed a number of experiments to measure the correctness of overall classification when the system was presented with composite images containing several patterns of different classes. We tried two versions of the system, one with small receptive field detectors and the other with large receptive field detectors. In both cases, we observed that the rate of correct classification decreased as the number of patterns in the image was increased. To determine the ability of the system to cope with variations in the patterns, images with random perturbations to the patterns were presented to the system in another series of experiments. In this case, we observed that the classification abilities of the large receptive field system are less affected and degrade more gracefully.

The classification process described in this paper is only the first step in the analysis of NMR spectra. It is of considerable interest to chemists to determine the precise association of the peaks in in the input image with different patterns. We are currently working on an extension to the system described in this paper to perform this task. We plan to refine the clustering algorithm to enable the use of feature-detectors with variable size receptive fields. We expect to improve performance by combining the evidence from multiple input sources, as is done in other NMR analysis methods.

Acknowledgments

We gratefully acknowledge the assistance received from Sandor Szalma and Istvan Pelczer of the NMR Data Processing Laboratory, Chemistry Department, Syracuse University. The images for the training set were generated by a program written by Sandor Szalma.

References

[1] Chen, C. "Statistical Pattern Recognition". Spartan Books, 1973.

[2] Fu, K. S. and Mui, J. K. "A Survey on Image Segmentation". *Pattern Recognition*, vol. 13, pp. 3-16, 1981.

[3] Fukushima, K. "Neural Network Model for Selective Attention in Visual Pattern Recognition and Associative Recall". *Applied Optics*, vol. 26, no. 23, pp. 4985-4992, 1987.

[4] Gross, K-H., Kalbitzer, H. R. "Distribution of Chemical Shifts in 1H Nuclear Magnetic Resonance Spectra of Proteins". *Journal of Magnetic Resonance*, vol. 76, pp 87-99, 1988.

[5] Jain, A. K. and Dubes, R. C. "Algorithms for Clustering Data". Prentice Hall, 1988.

[6] Kohonen, T. "Self-organized Formation of Topologically Correct Feature Maps". *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.

[7] Kohonen, T., Barnes, G., and Chrisley, R. "Statistical Pattern Recognition with Neural Networks: Benchmarking Studies". *Proc. IEEE Int. Conf. on Neural Networks*, vol. 1, pp. 61-68, 1988.

[8] Nagao, M. and Matsuyama, T. "A Structural Analysis of Complex Aerial Photographs". Plenum Press, 1980.

[9] Rumelhart, D. E., and McClelland, J. L. "Parallel Distributed Processing, Volume 1". MIT Press, 1987.

[10] Strong, G. W. and Whitehead, B. A. "A Solution to the Tag-Assignment Problem for Neural Networks". *Behavioral and Brain Sciences*, vol. 12, pp. 381-433, 1989.

[11] Wuthrich, K. "NMR of Proteins and Nucleic Acids". John Wiley & Sons, New York, 1986.