CHAPTER 3

Soft Computing Approaches

This chapter presents a brief discussion on some of the prominent soft computing approaches and their applications in pattern recognition. The chapter also presents a demonstration of optical character recognition of printed and handwritten digits as a type of image recognition using Artificial Neural Networks.

3.1 Overview

Soft Computing is a term that was coined by Lotfi Zadeh [6] to describe a collection of methodologies that aims to simulate human like decision making by exploiting the tolerance for impression, uncertainty, approximate reasoning and partial truth to achieve tractability, robustness and low-cost solutions. The principal notion of soft computing is that, precision and certainty carry a cost and as such whenever possible imprecision and uncertainty should be exploited as in the case of human intelligence. The role model for soft computing is the human mind and it intends at a formalization of the cognitive processes humans employ so effectively in the performance of daily tasks. Thus the soft computing approaches provide a strong foundation for the conception and design of high Machine Intelligence Quotient systems and therefore form the basis of future generation computing systems [7] [9].

As per wide acceptance, the most prominent components of soft computing are Fuzzy Logic (FL), Artificial Neural Networks (ANN), Genetic Algorithms (GA) and Probabilistic Reasoning (PR) with the latter subsuming belief networks, chaos theory and parts of learning theory [42] [6], where FL deals with imprecision arising from
3.2 Fuzzy Logic

vagueness, ANN provides the machinery for learning and adaptation, GA deals with optimization and searching and PR provides the means for dealing with uncertain information. Recently Rough Sets (RS) have also been proposed as a soft computing technique which provides means for dealing with uncertainty arising from limited discernibility of objects [7]. It is important to note that although there are substantial areas of overlap between FL, ANN, GA and PR, in general the components are complementary rather than competitive. For this reason, it is at times quite advantageous to use them in combination rather than exclusively.

In the present research the main focus was given on two of the prominent soft computing approaches mentioned above, namely Fuzzy Logic and Artificial Neural Networks. The two approaches are now presented.

3.2 Fuzzy Logic

Fuzzy logic is logic system, which is based on the Fuzzy Set Theory introduced by Lotfi Zadeh in 1965 [43]. In fuzzy set theory the membership of elements in a set is described with the help of a membership function valued in the real unit interval [0, 1]. Thus the elements of a set in fuzzy set theory can have degrees of membership. For example, if \( X \) denotes a set of group of people and \( A \) denotes a fuzzy set of old people in \( X \) then the elements of the set \( A \) (which will be a subset of \( X \)) is characterized by assigning to each element \( x \) of \( X \) a real value, such as 0.62, in the range of 0 to 1, as the degree of membership of \( x \) in \( A \). A membership degree of 0 will indicate that the person \( x \) in \( X \) is 'not at all' old and a value of 1 will indicate that \( x \) is a 'perfectly old' person, any value in between 0 and 1 will indicate the varying degree of oldness of \( x \). In contrast, in classical set theory the membership of elements in a set is assessed in binary terms according to a bivalent condition – an element either belongs or does not belong to the set. Fuzzy sets thus generalize classical sets, since the indicator functions (a function defined on a set \( X \) that indicates membership of an element in a subset \( A \) of \( X \), having the value 1 for all elements of \( A \) and the value 0 for all elements of \( X \) not in \( A \)) of classical sets are special cases of the membership functions of fuzzy sets, as the latter takes only the values 0 or 1 [44] [45]. In fuzzy set
theory the classical bivalent sets are usually called *crisp* sets.

The fuzzy logic, based on the concept of the fuzzy sets, is thus very appropriate to deal with vague (imprecise) propositions like "this person is old", "this woman is very beautiful", "this boy is too tall" etc which are very difficult to deal with, using classical logic.

![Fuzzy set representation of an old person](image)

**Figure 3.1: Fuzzy set representation of an old person**

Figure 3.1 illustrates the use of fuzzy sets to represent the notion of an old person and also depicts the granulation of age. The number, $\mu_A(x)$, represents the degree of membership of $x$ in the fuzzy set $A$. A membership function to calculate $\mu_A(x)$ is of the form:

$$\mu_A : X \rightarrow [0, 1]$$

According to Zadeh, fuzzy logic is used in two different senses [6] [46] namely *broad* and *narrow*. Fuzzy logic in the *broad* sense serves mainly as an apparatus for fuzzy control and analysis of vagueness in several application domains. In this sense it is one of the major techniques of soft computing. Its use in this sense offers tolerance to imprecision (vagueness) and suboptimality, so that quick, simple and sufficiently good solutions are obtained for problems involving real-life ambiguous situations.

Fuzzy logic in the *narrow* sense is a logical system that aims at a formalization of approximate reasoning. In this sense fuzzy logic may be viewed as a generalization of
3.3 Artificial Neural Networks

the traditional multivalued logical systems, e.g., Lukasiewicz's logic. However, its agenda remains quite different from the traditional logic systems because the concept of use of a logic system for approximate reasoning is not a part of traditional multivalued logic systems.

Fuzzy sets and fuzzy logic constitute the oldest and most reported soft computing paradigm [47]. Fuzzy logic has been applied to many fields, from control theory to artificial intelligence. Research on application of fuzzy logic and fuzzy set theory in the field of supervised pattern recognition was first reported in the year 1966 in the seminal note of Bellman et al. [48]. After that fuzzy logic has been increasingly used to improve many conventional methods for pattern recognition. An example of this is the introduction of fuzzy logic based clustering algorithms like the popular Fuzzy c-means clustering algorithm. Experiments on the application of fuzzy logic based pattern recognition on numeric and image data is presented in chapter 4. The experiments employ the Fuzzy c-means clustering algorithm.

3.3 Artificial Neural Networks

An Artificial Neural Network is a computational system that tries to mimic the learning process of biological neurons in order to achieve human-like ability of information processing. They offer a remarkable ability to derive meaning from complicated or imprecise data. Thus, they can be used to extract patterns and detect trends that are too complex to be noticed by other computer techniques. Artificial neural networks are made up of a set of simple processing units called artificial neurons assembled in the form of a closely interconnected network which offer a surprisingly rich structure in exhibiting some features of the biological neural network [49]. The concept of artificial neurons was first proposed in 1943 by Warren McCulloch [50].

Figure 3.2 shows the McCulloch-Pitts model of an artificial neuron. In this model the activation value $y$ is given by a weighted sum of its $d$ input values $x_i$ and a bias term $\theta$. The output signal $s$ is typically a nonlinear function (like binary function, ramp function or the sigmoid function) $f(y)$ of the activation value $y$. 
3.3 Artificial Neural Networks

The following equations describe the operation of a McCulloch-Pitts model of an artificial neuron:

**Activation:**

\[ y = \sum_{i=1}^{d} w_i x_i - \theta \]

**Output signal:**

\[ s = f(y) \]

Like human beings, ANNs learn by example. Thus an ANN can be configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true for ANNs as well. In case of an ANN the interconnections are assigned weights which are a measure to reflect the strength of the connection between the units and the amount of information flow between the connected units. ANNs can learn using both supervised and unsupervised learning paradigms.

A trained neural network can be thought of as an ‘expert’ in processing the category of information it is given to analyze. This expert can then be used to provide projections, given new situations of interest and to answer ‘what if’ questions. Other advantages of ANNs include:

(i) **Adaptive learning:** An ability to learn how to do tasks based on the data given for training or initial experience.

(ii) **Self organization:** An ANN can create its own organization or representation of the information it receives during learning time.
(iii) **Real time operation:** ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.

(iv) **Fault tolerance via redundant information coding:** Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

The most common type of ANNs consists of three groups or layers of processing units: *input layer*, *hidden layer* and *output layer*. The units in input layer are connected to the units in hidden layer, which are again connected to units in the output layer. An example is shown below:

![Figure 3.3: An artificial neural network with three layers of processing units](image)

The raw information to be processed by an ANN is fed to the input units. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units. This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights
between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

Artificial neural networks can have a variety of neural network architectures to accomplish different types of tasks. Some of these architectures include the feedforward (loop-free) ANNs, recurrent ANNs, Radial basis function ANNs, Kohonen self-organizing ANNs, stochastic ANNs, modular ANNs, Physical ANNs, Holographic associative memory, Neuro-fuzzy ANNs etc. Details on these architectures can be obtained in [49] and [51].

3.4 An Experiment on Digit Recognition using ANN

In this section an experiment on the application of a special type of neural network called a Hamming neural network for digit recognition from distorted images of printed and handwritten digits, is presented.

3.4.1 Hamming Network

A Hamming Network is a maximum likelihood classifier that can determine which of several exemplar pattern vectors is most similar to an input pattern vector [52]. The network has a very good ability to recognize noise corrupted patterns [53]. The network is used to solve pattern recognition problems which involve the use of binary vectors for pattern representation, with only two possible values, 0 and 1 for the individual elements of the pattern vectors. The hamming network implements a classifier that can determine which of the several stored exemplar vectors or templates is most similar to an input pattern vector. For doing this the network does a correlation or template matching between the input and the stored templates by calculating the distance between the binary input pattern vectors and the stored templates. Usually this distance measure is the Hamming distance. The hamming distance is defined as the number of differing bits between two corresponding, fixed-length input vectors.
The hamming network uses a matrix called the *prototype data matrix* which stores a set of noiseless pattern vectors as templates. The patterns are not learned by the system rather they are stored as exemplar vectors in the prototype data matrix. The objective of the hamming network is to decide which exemplar pattern vector in the prototype matrix is closest to the input vector. For doing this it calculates the similarities between all the vectors of the prototype matrix with the input vector. The input pattern vectors supplied to the network generally come with noise because of distortion by real-world events.

A diagrammatic representation of a hamming network is shown below:

![Diagram of a Hamming Network](image)

**Figure 3.4:** A Hamming network. The network input and output are represented by \(x\) and \(y\) vectors, respectively

As can be seen from Figure 3.3 the hamming network is divided into two parts, the *similarity subnet* and the *winner-take-all subnet*. The similarity subnet comprises of the \(d\)-neuron input layer and \(K\)-neuron memory layer. Each neuron \(k\) in the memory layer is connected to all \(d\) input layer neurons. The winner-take-all subnet consists of a fully connected \(K\)-neuron topology. An exemplar pattern vector \(\xi^k\) is stored in the network by letting the values of the connection weights between memory neuron \(k\) and the input layer neurons \(i\) \((i=1, 2, \ldots, d)\) to be:
3.4 An Experiment on Digit Recognition using ANN

The values of the weights \( W_{ij} \) in the winner-take-all subnet are chosen in such a way such that the output neurons inhibit each other. That is:

\[
W_{ij} = \begin{cases} 
1, & \text{for } i = j \\
-\varepsilon, & \text{for } i \neq j
\end{cases}
\]

where \( 0 < \varepsilon \leq 1/K \). \( \varepsilon \) is same for all \( W_{ij} \)'s for \( i \neq j \) and is usually set to \( 1/K \).

When a binary input pattern vector \( x \) is presented to the network for classification then the computation is the network takes place in two steps:

Step 1): Each neuron \( k \) (\( 1 \leq k \leq K \)) in the memory layer computes its similarity \( Z_k \) with the input pattern vector \( x \) using the following equation:

\[
Z_k = \frac{1}{2} \left( \sum_{i=1}^{d} a_{ki} x_i + d \right)
\]

Step 2): Each neuron \( k \) in the memory layer transfers its similarity value \( Z_k \) to the corresponding neuron \( k \) in the winner-take-all subnet (the connection weight between a neuron \( k \) in the memory layer and the corresponding neuron \( k \) in the winner-take-all subnet is set to 1). The winner-take-all subnet then finds the pattern vector \( i \) with the maximal similarity. Each neuron \( k \) in the winner-take-all subnet sets the initial value \( y_k(0) = Z_k/d \) and then computes \( y_k(t) \) iteratively \( (t = 1, 2, \ldots) \) by using the following:

\[
y_k(t) = \Theta_T \left( \sum_{i=1}^{d} W_{ki} y_i(t-1) \right)
\]

where \( \Theta_T \) is the threshold logic function defined as:

\[
\Theta_T(u) = \begin{cases} 
u, & \text{if } u \geq T, \\
0, & \text{otherwise.}
\end{cases}
\]

The two steps are repeated until the activity levels of the neurons in the winner-take-all subnet no longer change and only one neuron \( k \) in the winner-take-all subnet remains active, i.e., with a non-zero positive value for \( y_k \). This neuron is declared as the winner and it represents the exemplar pattern vector that is closest to the input pattern vector \( x \). The input pattern vector \( x \) is thus classified as the closest
match to the exemplar pattern vector $\xi^k$ corresponding to the winner neuron $k$ [54].

3.4.2 Digit Recognition using a Hamming Network

In the subsection, the experimental results and discussion of an experiment on digit recognition using a hamming network are presented. The experiment was carried out for illustration of a pattern recognition task by neural networks as a part of research on soft computing approaches.

For the experiment, two sets, each consisting of 10, noiseless, monochromatic, 32×48 pixels sized images of the 10 decimal digits, were considered as templates. Let such a set be called here as a template image set. A digit in a monochromatic image is represented as a grid of black and white pixels. The two template image sets are shown below:

![Template Image Set 1](image1.png)

Figure 3.5(a): The first template image set of noiseless monochromatic images of the 10 decimal digits

![Template Image Set 2](image2.png)

Figure 3.5(b): The second template image set of noiseless monochromatic images of the 10 decimal digits

The similarity subnet of the hamming network used for the task of classification consists of an input layer with 1536 (32×48) neurons, a memory layer with 10 neurons and a winner-take-all subnet also with 10 neurons (as the memory layer has 10 neurons).

To recognize an image of a digit, after reading it into a 32×48 matrix (comprising of all 0's and 1's), as one of the 10 digits, the matrix is reshaped to a vector of size 1×1536 by concatenating each row of the 32×48 matrix, one after another, in a single row. This input pattern vector of size 1×1536 is then applied to the input layer
of the hamming network and after processing of this input vector (as described in the previous subsection) one of the 10 neurons in the winner-take-all subnet gives an output value of ‘1’, that corresponds to one of the 10 decimal digits.

For testing the hamming network for recognition of digits from noisy images, the following 5 sets of input images of the 10 decimal digits were considered:

0123456789

Figure 3.6(a): First input image set of distorted monochromatic images for the 10 decimal digits

0123456789

Figure 3.6(b): Second input image set of distorted monochromatic images for the 10 decimal digits

0123456789

Figure 3.6(c): Third input image set of distorted monochromatic images for the 10 decimal digits

0123456789

Figure 3.6(d): Fourth input image set of distorted monochromatic images for the 10 decimal digits

0123456789

Figure 3.6(e): Fifth input image set of distorted monochromatic images for the 10 decimal digits

The experimental results carried out on the above 5 sets of images are now presented.
The following results were obtained from the hamming network for the 5 input image sets using the first template image set:

Table 3.1(a): Digit recognition obtained for first input image set using the first template image set

<table>
<thead>
<tr>
<th>Input Images</th>
<th>Identified As</th>
</tr>
</thead>
<tbody>
<tr>
<td>0123456789</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
</tbody>
</table>

Table 3.1(b): Digit recognition obtained for second input image set using the first template image set. Misclassifications are highlighted in gray colour

<table>
<thead>
<tr>
<th>Input Images</th>
<th>Identified As</th>
</tr>
</thead>
<tbody>
<tr>
<td>0123456789</td>
<td>0 1 2 5 9 5 5 7 0 9</td>
</tr>
</tbody>
</table>

Table 3.1(c): Digit recognition obtained for third input image set using the first template image set. Misclassifications are highlighted in gray colour

<table>
<thead>
<tr>
<th>Input Images</th>
<th>Identified As</th>
</tr>
</thead>
<tbody>
<tr>
<td>0123456789</td>
<td>0 1 2 3 9 5 5 7 5 9</td>
</tr>
</tbody>
</table>

Table 3.1(d): Digit recognition obtained for fourth input image set using the first template image set. Misclassifications are highlighted in gray colour

<table>
<thead>
<tr>
<th>Input Images</th>
<th>Identified As</th>
</tr>
</thead>
<tbody>
<tr>
<td>0123456789</td>
<td>0 1 2 3 1 5 5 7 8 9</td>
</tr>
</tbody>
</table>
3.4 An Experiment on Digit Recognition using ANN

Table 3.1(e): Digit recognition obtained for fifth input image set using the first template image set. Misclassifications are highlighted in gray colour

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>5</th>
<th>8</th>
<th>7</th>
<th>8</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

The following results were obtained from the hamming network for the 5 input image sets using the second template image set:

Table 3.2(a): Digit recognition obtained for first input image set using the second template image set. Misclassifications are highlighted in gray colour

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 3.2(b): Digit recognition obtained for second input image set using the second template image set

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 3.2(c): Digit recognition obtained for third input image set using the second template image set

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>
Table 3.2(d): Digit recognition obtained for fourth input image set using the second template image set. Misclassifications are highlighted in gray color.

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0123456789</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0100156769</td>
</tr>
</tbody>
</table>

Table 3.2(e): Digit recognition obtained for fifth input image set using the second template image set. Misclassifications are highlighted in gray color.

<table>
<thead>
<tr>
<th>Input Images:</th>
<th>0123456789</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identified As:</td>
<td>0108566789</td>
</tr>
</tbody>
</table>

For the above results it can be seen that out of 50 images from the input image sets 38 images were correctly recognized by the hamming network using the first template image set and for the second template image set, 40 images were correctly recognized. The average percentage of performance obtained is thus \( \left( \frac{38}{50} + \frac{40}{50} \right) \times 100 = 78 \), which is a good performance. However it should also be noted that the classification accuracy of a hamming network is fully dependant on the fact that how well the templates stored by it match the input pattern vectors. For more sophisticated recognition, neural networks with dynamic learning ability should be used.

3.5 Chapter Summary

In this chapter, a discussion on some of the prominent soft computing approaches like fuzzy logic and artificial neural networks is presented. On the part of fuzzy logic, the chapter presented a brief overview of a fuzzy set and its representation. It also presented a discussion on how fuzzy logic can be used to deal with vagueness encountered in natural language statements. The two broad senses of fuzzy logic are
also discussed. On the part of artificial neural networks, the chapter presented discussions on overview of ANNs, concept of an artificial neuron and its computational model, advantages of ANNs and structure of an ANN. Further the chapter also presented a short discussion on hamming neural network based pattern recognition and presented an experiment on the use of hamming network in recognition of digits from distorted monochromatic images of printed and handwritten digits.