Object Recognition Using Reflex Fuzzy Min-Max Neural Network with Floating Neurons

A.V. Nandedkar and P.K. Biswas

Electronics & Elec. Communication Engg. Department,
Indian Institute of Technology, Kharagpur - 721302, India
{avn, pkb}@ece.iitkgp.ernet.in

Abstract. This paper proposes an object recognition system that is invariant to rotation, translation and scale and can be trained under partial supervision. The system is divided into two sections namely, feature extraction and recognition sections. Feature extraction section uses proposed rotation, translation and scale invariant features. Recognition section consists of a novel Reflex Fuzzy Min-Max Neural Network (RFMN) architecture with "Floating Neurons". RFMN is capable to learn mixture of labeled and unlabeled data which enables training under partial supervision. Learning under partial supervision is of high importance for the practical implementation of pattern recognition systems, as it may not be always feasible to get a fully labeled dataset for training or cost to label all samples is not affordable. The proposed system is tested on shape data-base available online, Marathi and Bengali digits. Results are compared with "General Fuzzy Min-Max Neural Network" proposed by Gabrys and Bargiela.

1 Introduction

Object recognition is an important component in computer vision. Object recognition broadly involves two steps namely, feature extraction and pattern classification. Efficient object recognition demands rotation, translation and scale invariant (RTSI) features. Pattern classification extracts the underlying structure in the data and performs the recognition. Fuzzy interpretation of patterns is very natural in cases where precise partitions of data are not known. Zadeh [1] elaborated the importance of fuzzy logic for pattern classification in his seminal paper. The merge of fuzzy logic and neural network for pattern classification can be found in "Fuzzy Min Max Neural Network" (FMNN) proposed by Simpson [2][3]. Gabrys and Bargilela [4] proposed a merge of FMNN classification and clustering algorithms called as "General Fuzzy Min-max Neural network" (GFMN). This hybridization allowed learning under partial supervision. Semi-supervised learning is of high importance for the practical implementation of pattern recognition systems, as it may not be always feasible to get a fully labeled dataset for training or cost of labeling all the samples is not affordable.

The proposed Object Recognition System (ORS) uses a new set of RTSI features. Recognition is carried out using proposed "Reflex Fuzzy Min-Max Neural Network with Floating Neurons" (RFMN). RFMN is trainable by means of partial supervision.

P. Kalra and S. Peleg (Eds.): ICVGIP 2006, LNCS 4338, pp. 597 – 609, 2006.

It uses aggregation of fuzzy hyperbox sets (called as hyperbox neurons) [2][3] to represent classes or clusters. A variety of ORS methods are available such as, boundary based analysis via Fourier descriptors [5], neural networks models [6] and invariant moments [7]. However, most of these methods are too computationally expensive or are not invariant under the three types of transformations i.e., rotation, translation and scaling (RTS). An inexpensive ORS was proposed by Torres-Mendez et al [8] based on radial coding technique.

The proposed RFMN with floating neurons exploits use of reflex mechanism inspired from human brain for the pattern classification and clustering. It uses Compensatory Neurons (CN) to overcome the hyperbox overlap and containment problems [9] [10]. CNs are inspired from the reflex system of human brain [11]. CNs maintain the hyperbox dimensions and control the membership in the overlapped region. During the training RFMN tries to label the unlabeled data, thus it is possible to learn from mixture of labeled and unlabeled data. The unlabeled hyperbox neurons created during training are kept floating and are restrained from contributing to the classification. This approach has improved performance of RFMN compared to GFMN [4]. Gabrys and Bargiela advocated the use of a new activation function [4] for FMNN based algorithms. But we observed that their activation function can lead to errors and is discussed in section 3.

The main contribution of this work is development of a new architecture for semi-supervised learning and new set of RTSI features for object recognition. Rest of the paper is organized as follows. Section II elaborates new RTSI features. The proposed new RFMN architecture is explained in section III. Detailed learning algorithm and recall procedure is explained in section IV. Section V shows the experimental results on real datasets. Section VI concludes with summery.

2 RTSI Features

Feature can be defined as quantitative description of input within a lower dimensional space [12]. It plays an important role in object recognition systems (ORS) since the information related to an object is contained within the extracted features. In an ORS, pre-processing is required to extract the features. This may include image enhancement, filtering, segmentation [13] etc. Object segmentation is a must to recognize it. For an invariant ORS feature extraction must be invariant to translation, rotation and scale. Here we propose a new set of RTSI features for object recognition. This includes 1) normalized moment of inertia, 2) max to average ratio, 3) average to max-min difference ratio, 4) radial coding [8] and 5) radial angles.

To extract these features one needs to compute centroid of an object. Here we assume that after segmentation a binary image of the object is available for post processing. The centroid (C_x, C_y) of a two-dimensional object is given by,

$$C_x = \frac{\sum_{i=1}^{N} x_i * f(x_i, y_i)}{N} \quad , \quad C_y = \frac{\sum_{i=1}^{N} y_i * f(x_i, y_i)}{N} \tag{1}$$

$$where \quad f(x_i, y_i) = \begin{cases} 1 & if \quad pixel \quad p(x_i, y_i) \in object \\ 0 & otherwise \end{cases} \tag{2}$$

x_i, y_i : co-ordinate values and N: total number of object pixels.
Once the centroid is computed other features are extracted as follows:

1) Normalized moment of Inertia (NMI)
In general the moment of inertia quantifies the inertia of rotating object by considering its mass distribution. The moment of inertia (MI) is normally calculated by dividing the object into N-small pieces of mass $m_1, m_2,..,m_N$, each piece is at a distance d_i from the axis of rotation. MI is given by,

$$\mathrm{I} = \sum_{i=1..N} m_i d_i^2 \tag{3}$$

In case of object in a binary image, we consider pixel as unit pieces (i.e. m=1). Due to the finite resolution of any digitized image, a rotated object may not conserve the number of pixels. So moment of inertia may vary but normalized moment of inertia reduces this problem. Normalized MI is invariant to translation, rotation and scale. This can be observed from Table 1 depicting features for an object shown in Fig. 1(a). The normalized moment of inertia (MI) of an object is [8] computed by,

$$I_N = \frac{1}{N^2}\sum_{i=1}^{N} d_i^2 = \frac{1}{N^2}\sum_{i=1}^{N} ((x_i - C_x)^2 + (y_i - C_y)^2) \tag{4}$$

where $(C_x C_y)$ are centroid co-ordinates and x_i, y_i are object pixel co-ordinates. d_i pixel distance from centroid.

2) Max to average length ratio (MAR)
MAR is a ratio of maximum (d_{max}) of distance of object pixels from centroid to the average pixel distance (d_{avg}) from centroid.

$$\mathrm{MAR} = \frac{d_{max}}{d_{avg}} \tag{5}$$

Note the RTS invariance of this feature from Table 1.

3) Average to Max-Min Difference (AMMD) Ratio
AMMD is a ratio of average pixel distance from centroid d_{avg} to difference between maximum (d_{max}) and minimum (d_{min})of pixel distance from centroid. It is given by,

$$\mathrm{AMMD} = \frac{d_{avg}}{(d_{max} - d_{min})} \tag{6}$$

Table 1 indicates AMMD is a RTS invariant feature.

4) Radial Coding (RC) and Radial Angles (RA)
The radial coding features are based on the fact that circle is the only geometrical shape that is naturally and perfectly invariant to rotation. RC is computed by counting the number of intensity changes on circular boundaries of some radius inside the object. This simple coding scheme extracts the topological characteristics of an object

Table 1. NMI, MAR and AMMD for Fig.1(a) with various rotations, translations and scales

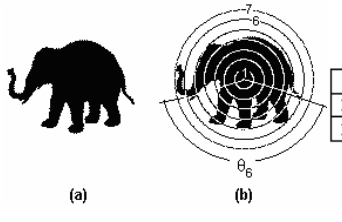
Rotation (Degrees)	NMI	MAR	AMMD
0	0.193	2.161	0.465
25	0.193	2.166	0.464
55	0.193	2.179	0.463
85	0.194	2.176	0.463
105	0.194	2.145	0.468

Size (%)	NMI	MAR	AMMD
120	0.193	2.164	0.466
140	0.194	2.154	0.465
160	0.193	2.174	0.462
180	0.193	2.169	0.463
200	0.193	2.171	0.462

regardless of its position orientation and size. The methodology to obtain the radial coding features of an object can be seen in [8]. Along with RC, proposed radial angles (RA) are found out as follows:

1) Obtain the centroid of the object.
2) Generate K equidistant concentric circles C_i around the centroid. The spacing is equal to the distance between centroid and furthest pixel of the object divided by K.
3) For each circular boundary, count the number of intensity changes (zero to one or one to zero) that occur in the image. These are radial coding features.
4) Find the largest angle (θ) between the two successive intensity changes for every circle. These are called as Radial Angles. If $\theta > \pi$ then take θ as $2\pi - \theta$. This is a necessary step to avoid the dependency of angle measurement on reference point or direction of the measurement. If there is no intensity change then take θ=0.

Fig. 1(b) shows an example of radial coding and angles. Extracted features are shown in Fig.1(c). These features are also rotation, translation and scale invariant and can be noted from Table 2 and 3. We used seven concentric circles to code an object. Thus total feature vector length used in the proposed ORS is 17 (7RC+7RA+ NMI+MAR+AMMD).



Codes/Ring	1	2	3	4	5	6	7
RC	0	0	3	6	10	2	1
RA(Rad)	0	0	1.08	2.68	2.64	2.56	0

(c)

Fig. 1. (a) Object (b) Radial Codes and Angles (c) RC and RA

Table 2. RA for various rotations of Fig 1(a)

Rotation (deg) / Ring	1	2	3	4	5	6	7
0	0	0	1.08	2.68	2.64	2.56	0
25	0	0	1.1	2.78	2.44	2.56	0.06
55	0	0	1.82	2.26	2.02	2.56	0.24
85	0	0	1.82	2.76	2.22	0.46	0
105	0	0	0.74	2.66	2.32	2.56	0

Table 3. RA for various sizes of Fig 1

Size (%)/ Ring	1	2	3	4	5	6	7
120	0	0	1.08	2.6	2.54	2.54	0
140	0	0	1.1	2.7	2.54	2.56	0
160	0	0	1.08	2.7	2.62	2.58	0
180	0	0	1.1	2.7	2.54	2.58	0
200	0	0	1.06	2.7	2.64	2.58	0

3 Reflex Fuzzy Min-Max Neural Network with Floating Neurons

The proposed ORS uses a novel Reflex Fuzzy Min-Max Neural Network (RFMN) with floating neurons for the recognition purpose. RFMN uses aggregation of hyperbox fuzzy sets to represent classes or clusters. It can be trained in two ways i.e. classification (supervised learning) and hybrid mode (semi-supervised learning). During training RFMN tries to accommodate the training samples in the form of hyperbox fuzzy sets. The class overlaps are handled by reflex section. In hybrid mode, RFMN tries to label the unlabeled data using knowledge acquired from available labeled data. After completion of training, many hyperbox fuzzy sets may remain unlabeled due to lack of evidence for these sets. Neurons representing such hyperbox fuzzy sets are restrained from contributing to the output. Such neurons are called as "Floating Neurons". Floating neurons (FN) can be labeled and are allowed to contribute to the output if evidence for a class is found out later on. Since RFMN learns on-line whenever data is made available it can be trained without hampering performance on the earlier acquired knowledge.

3.1 RFMN Architecture

The proposed architecture of Reflex Fuzzy Min-Max Neural Network (RFMN) is shown in Fig. 2.

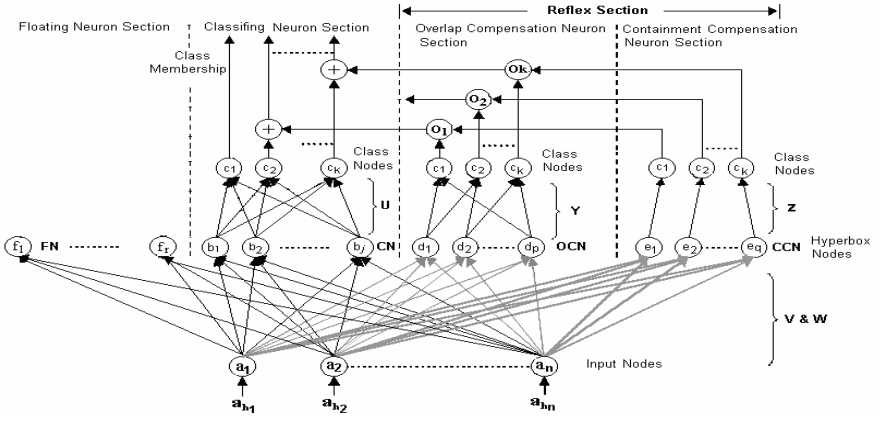


Fig. 2. RFMN Architecture

It is divided into three sections: 1) The classifying neuron section (CL) 2) Reflex section and 3) Floating Neuron section. The classifying section contributes in calculating memberships for different classes. The Reflex section consists of two subsections, Overlap Compensation neuron (OCN) section and Containment Compensation neuron (CCN) section. Reflex section is active whenever a test sample falls in the class overlap area. This action is very similar to the Reflex action of human brain which takes over the control in hazardous conditions. It compensates the output of classifying section and solves the dispute of membership in class overlapped area. Floating neuron section represents hyperbox fuzzy sets whose labels are not confirmed. These neurons are transferred dynamically during training to the classifying neuron section if class evidence is found.

An n-dimensional input $A_h = (a_{h1}, a_{h2}, \ldots a_{hn})$ is applied to the input nodes a_1-a_n. The neurons b_1-b_j are classifying neurons. Classifying section collects output of these neurons at class nodes C_1-C_k. During training hyperboxes belonging different classes do overlap as depicted in Fig3(b), 4(b). These overlaps and containments infer OCNs and CCNs respectively in the reflex section. The nodes d_1-d_p are overlap compensation neurons and e_1-e_q represent the containment compensation neurons. Outputs of OCN & CCN are collected at a class node C_i in respective compensation sections. The output of floating neurons (FNs) f_1 -f_r are not connected to any class node. The activation function of the classifying neuron b_j is given by [3],

$$b_j(A_h, V_j, W_j) = \frac{1}{n}\sum_{i=1}^{n} min\,[(1 - f(a_{hi} - w_{ji}, \gamma)), (1 - f(v_{ji} - a_{hi}, \gamma))] \tag{7}$$

where V, W: min-max point of the hyperbox b_j. γ: Fuzziness controller, $f(x,y)$ is a two parameter ramp threshold function, n- dimension of data.

$$f(x,y) = \begin{cases} 1 & if\ x\gamma > 1 \\ x\gamma & if\ 0 \le x\gamma \le 1 \\ 0 & if\ x\gamma < 0 \end{cases} \tag{8}$$

Eq.7 finds membership for a given input as an average of memberships along each dimension. Membership depends on the distance of applied input from hyperbox min and max point along each dimension. Gabrys and Bargiela [4] modified the above activation function and advocated use of their new activation function given by Eq.9 for FMNN based algorithms. It is stated in [4] that Simpson's activation function (Eq.7) [3] offers a large memberships even though very few features are close to the range specified by the hyperbox min-max points. To solve this problem Eq.9 [4] offers a membership based on the minimum of the memberships (match) along each dimension.

$$b_j(A_h, V_j, W_j) = \min_{i=1..n}(min[(1 - f(a_{hi} - w_{ji}, \gamma)), (1 - f(v_{ji} - a_{hi}, \gamma))]) \tag{9}$$

But we observe that this criterion of offering membership based on minimum membership is not suitable universally. The search for the minimum membership penalizes too heavily and leads to errors in cases where matching of features is more important rather than searching for a minimum match. In case of proposed features for object recognition, the requirement is to see how many features of an input match

to the learned patterns, thus we found that Eq.7 is more suitable than Eq.9. This is supported by our results of experiment 1, in Section 5.

As training progresses hyperbox size goes on increasing to accommodate the applied input. The maximum hyperbox size is controlled by the expansion coefficient

$$\Theta \geq \frac{1}{n}\sum_{i=1}^{n}(max(w_{ji}, a_{hi}) - min(v_{ji}, a_{hi})) \tag{10}$$

As stated earlier while training the network, hyperboxes representing different classes may overlap, or a hyperbox of one class may contain a hyperbox of another class as depicted in Fig. 3(b), 4(b) respectively. The overlap compensation and containment compensation neurons are trained to handle these situations. Fig. 3(a) depicts the details of overlap compensating neuron (OCN), which represents a hyperbox of size equal to the overlapping region between two hyperboxes. OCN is active only when the test sample falls in the overlap region. The activation function is given by Eq.(11) and (12).

$$d_{j_p} = U(b_j(A_h, V, W) - 1) \times (-1 + b_{j1}(A_h, V_p, W_p)) \tag{11}$$

$$\text{where } b_{j1}(A_h, V_p, W_p) = \frac{1}{n}\sum_{i=1}^{n} max\left(\frac{a_{hi}}{w_{p\,ji}}, \frac{v_{p\,ji}}{a_{hi}}\right) \tag{12}$$

p=1,2. d_{j1} and d_{j2} are Class1 and Class2 outputs. V,W: OCN min-max points. V_1, W_1, V_2, W_2: min-max point of overlapping hyperboxes $U(x)$: a unit step function. $b_j()$ is same as Eq.(7).

The unit step function with threshold of '1' allows OCN to be active whenever applied input falls inside the overlap region represented by it. If the test data is outside the OCN region, membership calculated by $b_j()$ is less than one and thus $U(b_j() - 1)$ term is zero. This makes compensatory neurons inactive i.e. no compensation is added. Compensation is produced whenever the test data falls inside overlapped region. If data is contained in OCN region (as shown in Fig.3(b)), its membership is calculated for the respective classes depending on its distance from the min-max points.The activation function of this neuron is such that it protects the class of the min-max point of the overlapping hyperboxes, which improves the learning accuracy. The output of this neuron is connected to the two class nodes of overlapping classes (OCN section Fig. 1).

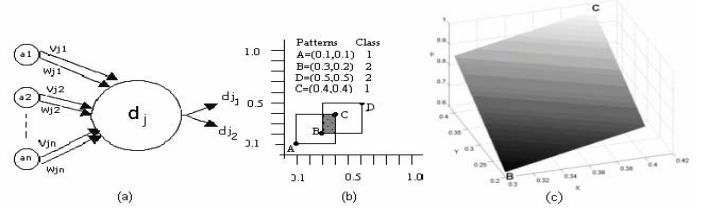


Fig. 3. Overlap Compensatory Neuron

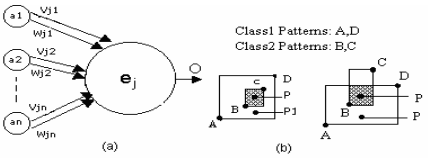


Fig. 4. Containment Compensatory Neuron

Referring to the overlap shown in Fig. 3(b) note that the resulting membership grade after adding compensation decreases gradually from point C to B for class 1 (Fig. 3(c)) and from B to C for class2. Thus activation function tries to give a membership grade for the applied input considering its position in the OCN region.

The containment compensation neuron (CCN) is shown in Fig. 4(a). This represents a hyperbox of size equal to the overlap region between two classes as shown in Fig. 4(b). Similar to OCN activation function the term $U(b_j() -1)$ finds whether the input data falls inside the overlapped region represented by CCN. This neuron is also active only when the test data falls inside the overlapped region. The activation function of CCN is:

$$O_{c_j} = -U(b_j(A_h, V, W) -1)) \tag{13}$$

where O_{cj}: output, V,W: CCN min-max points, $U(x)$: unit step function, $b_j()$: same as Eq.(7)

This activation function allows a hyperbox of one class to be contained in a hyperbox of different class. The output of CCN is connected to the class that contains the hyperbox of other class (CCN Section Fig.2).

In hybrid mode of learning, there may be generation of hyperboxes without labels due to lack of class evidence. These hyperbox neurons are kept in Floating Neuron section without connection to output and thus are not allowed to contribute to the out put. Floating neurons are brought dynamically into the classifying neuron section if evidence of a class is found.

The number of output layer nodes in CL section is same as the number of classes learned. The number of class nodes in the CCN, OCN section depends on the nature of overlap the network faces during the training process. The final membership calculation is given by,

$$\mu_i = \max_{k=1..j}(b_k u_{ki}) + \min\left(\min_{l=1..p}(d_l y_{li}), \min_{m=1..q}(e_m z_{mi})\right) \tag{14}$$

where U,Y,Z are the connection matrices for the neurons in the three sections.
j,p,q are number of neurons in respective sections.

Eq.(14) takes care of multiple class overlaps. It gives maximum grade to a class from the available grades considering its compensation.

3.2 Comparison of RFMN with GFMN and FMNN

FMNN and GFMN use a process called as contraction to solve the class overlaps. Nandedkar and Biswas [9] [10] pointed out that the contraction process causes errors during training. To overcome this problem, contraction process was removed and a

reflex mechanism was added in to the FMNN architecture. It is observed that in case of GFMN hybrid mode of learning, many hyperboxes remain unlabeled after training due to no class evidence for them. Thus any test sample falling in these hyperboxes are not classified. But in situations where we need to take a decision based on existing knowledge, one needs to ignore the output of these neurons. Thus we propose a concept of "Floating Neurons" to overcome this problem. Floating neurons are labeled dynamically and are allowed to contribute to the output if evidence for a class is found.

4 Training Algorithm and Recall Procedure

RFMN Training algorithm creates and expands hyperboxes depending on the demand of the problem. It utilizes the currently learned structure to label the unlabeled data. If there is any overlap, containment created (between hyperboxes of different classes) while expanding labeled hyperboxes, respective compensatory neuron is added to the network. Note that hyperboxes are not contracted in RFMN learning.

a) Training Algorithm

Training algorithm consists of mainly two steps, Data Adaptation and Overlap Test. Assume $\{A_h, C_i\}$ is a training data, $\{b_j, C_j\}$ a hyperbox for class C_j. θ: current hyperbox size θ_{max}: maximum hyperbox size. Initialize the network with b_1 with $V_1=A_h$, $W_1= A_h$ and class C_i for an ordered data pair $\{A_h, C_i\}$, Repeat the following steps 1 and 2 for the all-training samples. Note that for simplicity unlabeled data and hyperboxes are represented by C_0.

STEP 1: Data adaptation

Find a $\{b_j, C_j\}$ for training sample $\{A_h, C_i\}$ such that $C_j=C_i$ or $C_j=C_0$ offering largest membership, $\theta \leq \theta_{max}$ and is not associated with any OCN or CCN. Adjust the min-max points of hyperbox b_j as:

$$V_{ji}^{new} = \min (V_{ji}^{old}, A_{hi}) \quad W_{ji}^{new} = max (W_{ji}^{old}, A_{hi}) \text{ where } i = 1,2\ldots n \tag{15}$$

and If $C_j = C_0$ and $C_i \neq C_0$ then $C_j=C_i$. Take a new training sample.

If no b_j is found, create a new hyperbox with $V_j=W_j=A_h$ and class C_i.

STEP 2: Overlap Test

Assuming that b_j expanded in previous step is compared with b_k with class label $C_k \neq C_i$.

a) ***Isolation Test****:*

If $(V_{ki}<W_{ki}<V_{ji}<W_{ji})$ or $(V_{ji}<W_{ji}<V_{ki}<W_{ki})$ is true for any i, $(i \in 1..n)$ Then (b_k, b_j) are isolated and Check the following:

Case1: if $C_j=C_0$ then assign $C_j=C_k$

Case2: if $C_k=C_0$ then assign $C_k=C_j$

Case3: if $C_j=C_k=C_0$

Stop further expansion of these hyperboxes if any of the above cases is satisfied and go to step1. Else go for Containment test.

b) Containment Test*:*

If $(V_{ki}<V_{ji}<W_{ji}<W_{ki})$ or $(V_{ji}<V_{ki}<W_{ki}<W_{ji})$ is true for any i, $(i \in 1..n)$ then Create a CCN with hyperbox min-max co-ordinates given by,

$$V_{ci}= max(V_{ki},V_{ji}),\ \ W_{ci}=min((W_{ki},W_{ji}) \text{ for } i =1,2\ldots n \tag{16}$$

Else hyperboxes are not facing containment problem go to step (c)

*c) **Overlap compensation neuron creation**:*

Create a OCN with hyperbox min-max co-ordinates given by,

$$V_{oci}=max(V_{ki},V_{ji}),\ W_{oci}=min(W_{ki},W_{ji}) \text{ for } i=1,2\ldots n \tag{17}$$

Avoid further expansion of hyperboxes belonging to different classes, which are facing the problem of overlap and containment, in the next expansion cycles.

b) Recall Procedure

The class nodes in each section calculate the class memberships and respective compensations. The summing node in the classifying neuron does the final grade calculation. The membership grade is computed according to Eq. 14 by adding the compensation to the class membership.

5 Experimental Results

The basic aims for the experiments were to verify 1) Effectiveness of change in activation function on the performance of RFMN, 2) To compare performance of proposed set of RTSI features, 3) To verify performance of proposed ORS on various datasets, 4) To check performance of RFMN under partial supervision.

a) Effect of activation function

Here we used shape database [14] available on line. Fig.5 depicts some examples. It consists of 18 different classes and 12 images for each class, in total 216 images. Proposed RTSI features were extracted and fifty percent samples were selected randomly for training. Performance of RFMN on complete dataset with two different activation functions is compared with GFMN. Results are presented in Table 4.

It is clear from the results that RFMN performance is better than GFMN with activation function Eq.9. But it improves a lot when Eq.7 is used. The reason is that for the proposed RTSI features how many features match to the learned patterns is more important than the mismatches. Hence we recommend using activation functions depending on the nature of features.

Fig. 5. Few Images from Database [14]

Table 4. Activation function comparision

Algorithm	Learning Error (%)	Test Error (%)
RFMN (Eq. 7)	0	1.39
RFMN (Eq. 9)	0	9.26
GFMN	0	19.91

b) Performance on Various Feature Sets

Performance of RFMN, GFMN and K-Nearest Neighbor (KNN) [12] on shape database [14] using various feature sets is compared in Table 5. Training dataset is prepared by selecting 50% samples randomly. Note that a better object recognition is achieved using RFMN and the new set of RTSI features.

Table 5. Feature Set comparison (**LE- Learning Error, TE – Test Error**)

	RFMN		GFMN		K-NN (n=1)		K-NN (n=3)	
Features	**LE**	**TE**	**LE**	**TE**	**LE**	**TE**	**LE**	**TE**
New RTSI Features	0	1.39	0	19.91	0	4.62	22.22	20.83
Invariant Moments[13]	0	3.24	0	20.37	0	7.87	15.74	21.29
Radial Coding [8]	0	7.87	0	11.11	0	7.87	22.22	19.90

c) Test for RTS Invariance

To test the RTS invariance performance of proposed ORS rigorously we tested it on Bengali, Marathi digit database and an expanded shape database created from [14].

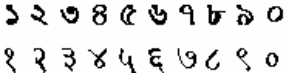


Fig. 6. Bengali (First Row) and Marathi Digits (0-9)

The details of the image database are given in Table 6.

Table 6. Database details

Database	Rotations	Sizes	Classes	Samples /Class	Total
Bengali ,Marathi Digits (0-9)	0, 10 35, 55, 60, 75, 90, 110, 135 Degrees	Font 20,24,28	10	30	300
Expanded Shape Database	0, 45, 90 Degrees	50, 100, 150 % of size in [14]	18	108	1944

For training the system, fifty percent of new RTSI features were selected randomly. Complete dataset is used for the test purpose. Table 7 shows that performance of proposed RFMN classifier is better than GFMN and KNN. A good recognition of proposed ORS i.e. combination of rotation, scale and translation invariance of new RTSI features and RFMN is demonstrated.

d) Semi-Supervised Learning

As stated earlier this mode of learning is suitable for the practical implementation of pattern recognition based systems, since practically it may not be always possible to label every training sample or cost of labeling is very high. To study the performance under partial supervision a mixture of labeled and unlabeled samples is used to train

Table 7. Performances on Bengali, Marathi Digits and Shape Database

Database	RFMN		GFMN		KNN (n=1)		KNN (n=3)	
	LE	**TE**	**LE**	**LE**	**LE**	**TE**	**LE**	**TE**
Bengali Digits	0	4	0	35.67	0	11	15.33	20
Marathi Digits	0	8	0	35.33	0	12.66	1.33	21
Expanded Shape	0	1.75	0	7.87	0	2.82	4.01	6.79

RFMN and GFMN. RFMN tries to apply the acquired knowledge from the labeled sample to label unlabeled and extract the underlying data structure. In this experiment, fifty percent of shape database and expanded shape database were selected randomly for training. Out of the selected training samples 2/3 samples were unlabeled randomly. This mixture was used to train GFMN and RFMN. Testing was carried out on complete dataset.

Table 8 compares results for RFMN, GFMN and KNN on shape and expanded shape database (Table 6). For training KNN we used available labeled data. RFMN and GFMN expansion coefficients were 0.1 and 0.05 for shape, expanded shape dataset respectively. It is clear that RFMN performance is better than GFMN and KNN. Compared to the hybrid mode learning of RFMN, performance of GFMN is poor due to the problems of unlabeled hyperboxes and its activation function. The interference of unlabeled hyperboxes leads to no classification of the input for GFMN.

Table 8. Semi-Supervised Test Error

Database	RFMN	GFMN	KNN(n=1)	KNN(n=3)
Shape [14]	11.57	48.61	21.29	30.55
Expanded Shape	11.98	45.37	18.10	23.14

6 Conclusion

A new rotation, translation and scale invariant object recognition system along with a novel RFMN architecture with floating neurons is presented. The need to select an activation function for neurons depending on the nature of features is discussed. The problem in GFMN hybrid mode learning is elaborated. The concept of floating neurons has improved the performance of RFMN in hybrid mode of learning. It helped to solve the problem due to unlabeled hyperboxes in GFMN. Experimental results show that the proposed object recognition system with RFMN classifier learns efficiently even with very few labeled samples added to unlabeled data. This is an important consideration for the practical implementation of pattern recognition system. Moreover performance of proposed set of RTSI features is found to be better than radial coding and invariant moments feature.

References

1. Zadeh, L.A.: Fuzzy Sets , Information and control, (1965), Vol. 8, 338-353.
2. Simpson, P.K.: Fuzzy Min-Max Neural Network – Part I: Classification, IEEE Tran. on Neural Networks, Vol.3,no.5, Sep. (1992) 776-786.

3. Simpson, P.K.: Fuzzy Min-Max Neural Network – Part II: Clustering, IEEE Tran. Fuzzy System, Vol.1, no.1, Feb. (1993) 32-45.
4. Gabrys, B., Bargiela, A.: General fuzzy min-max neural network for clustering and classification, IEEE Tran. Neural Network, Vol.11, May (2000) 769-783.
5. Kauppinen H,.Seppanen T, and Pietikamen M.,: An experimental comparison of Autoregressive and Fourier Based Descriptors in 2D shape classification, IEEE Trans. Pattern Analysis, Machine Intelligence, Vol.17, Feb. (1995) 207-210.
6. Perantonis S.J and. Lisboa P. J. G: Translation, rotation and scale invariant pattern recognition by high-order neural networks and moment classifiers, IEEE Trans. Neural Net-works, Vol. 4, July (1993), 276-283.
7. The C.H. and Chin R.T.: On image analysis by the methods of moments, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 10, July (1988) 496-513.
8. Torres-Mendez L.A., Ruiz-Suarez J.C., Sucar L.E and Gomez G., "Translation, rotation and scale-invariant object recognition", IEEE Trans. on Systems, Man and Cybernetics, Vol.30, No.1, February (2000) 125-130.
9. Nandedkar, A.V., Biswas, P.K.: A Fuzzy min-max neural network classifier with compensatory neuron architecture, 17th Int. Cnf. on Pattern Recognition (ICPR2004), Cambridge UK , Vol. 4 , Aug (2004) 553-556.
10. Nandedkar A.V., Biswas P.K.: A General fuzzy min max neural network with compensatory neuron architecture, Lecture Notes in Computer Science, Vol. 3683, Aug (2005), 1160-1167.
11. Baitsell, G.A.: Human Biology, second edition, Mc-Graw Hill Book co. inc. NY, (1950).
12. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification, 2nd edition, John Wiley &Sons Inc, Singapore (2001).
13. 13.Gonzalez R. C and Woods R. E.: Digital Image Processing, 2nd Edition, Pearson Education Pvt. Ltd., Delhi (2002).
14. T. Sebastian, P. Klein, B. Kimia: Recognition of shapes by editing shock graphs, 18th Int. Cnf. on Computer Vision (ICCV'01), Vol.1, (2001) 755-762.
http://www.lems.brown.edu/vision/researchAreas/SIID/