A Fuzzy Backpropagation Neural Networks for the Classification of Biological Data

Hiefa Alajail — *Faculty of Educati* — *Misurata University mmmm* — *H.Alajail@edu.misuratau.edu.ly*

Fathi Gasir — *The Libyan Academy* — *for Postgraduate Studies - Tripoli* — *Fathi.Elgasir@Academy.edu.ly*

Abstract:

This paper investigates the effects of applying fuzzy techniques to artificial neural networks (ANN) for the classification of biological data. A fuzzy neural networks (FNNs) model was proposed and evaluated as a system for image classification. This system involved the process of collecting dataset, image processing and image classification. Patch-based technique is used to present images to the neural network. Feed-Forward Backpropagation neural networks are used to build the system. Fuzzy Min-Max Neural Networks (FMNN) approach was used to synthesize Fuzzification and neural networks to generate fuzzy neural networks that can handle imprecision and uncertainty. The approach is evaluated using images from the data portal (Papers with Code) website. Experimental results have shown an improvement in the performance of fuzzy neural networks compared with neural networks.

Introduction:

A neural network (NN) is a collection of interconnected processing units designed to have performance characteristics that model the human brain. This appears clearly in that: The network acquires knowledge by learning from its environment, and synaptic connection strengths are applied between neurons in order to save the knowledge as it is obtained. In other words, NNs attempt to imitate biological neural networks, both in architecture and operation. Neural networks are trained by using knowledge collected (that is described by using statistical methods or fuzzy logic) [1].

Neural networks have several alternative architectures, learning methods and activation functions, each of which has different features that can be used for specific applications [2]. The typical architecture of the network is a characterisation of the neurons' layers, the number of neurons in each layer and the connections between the layers. The artificial neuron consists of a number of inputs, each one representing the output of another neuron and having a weight assigned to it. The sum of all the inputs multiplied by their weight determines the activation level of the neuron. The threshold value of each neuron determines whether the neuron will fire or not, by comparing its value with the net (weighted sum). The output has two values - 1, at which point the net is greater than the threshold and the neuron fires, and 0, where the net is smaller than, or equal to the threshold, and the neuron stays quiet. The NN has several layers of neuron, these being categorised into an input layer, hidden layers and output layers, and in overall terms NNs can be classified by the number of hidden layers which they contain, so they are referred to as 'single layer' if they do not have any hidden layers, and 'multilayer' if there are one or more hidden layers [3].

Fuzzy logic, which measures the likelihood of an event occurring in order to represent imprecise and fuzzy data, is what is described as human reasoning, it can also be described as a tool for representing and utilizing data and information that has non-statistical uncertainty. On the other hand, binary logic explains crisp events - that is, events that either occur or do not. Reasoning on higher semantic or linguistic level issues are often dealt with using fuzzy logic, so it can be said that fuzzy logic is not logic that is fuzzy, but logic that is used to describe fuzziness. Applications of fuzzification have been to soften sharp decision boundaries [4].

Fuzzy neural networks (FNNs) differ from traditional neural networks in their ability to handle imprecise or uncertain data. Traditional FNNs do not explicitly address the reliability aspect of uncertain real-world applications, whereas FNNs incorporate measures of information reliability for rule training and decision-making in the presence of uncertain input data. In other words, FNNs allow for improved handling of imprecise or uncertain data compared to traditional neural networks. [5] [6].

This paper presents a novel framework for the fuzzification of Backpropagation Neural Networks using Fuzzy Min-Max Neural Networks approach and Patch-based technique for presenting the images [7] [8] [9]. The most important characteristic of this technique is that it uses and maintains the features of Backpropagation neural network**s,** Fuzzy logic and Patch-based technique.

The paper is structured as follows: In Section II Artificial Neural Network and Images classification are highlighted. The Fuzzy Neural Networks is described in Section III. The Experimental methodology is described in Section IV. Section V shows the experiments and section VI presents the results and discussion. Finally the paper is concluded in Section VII.

Artificial Neural Network and Images Classification:

Basically, image classification is a pattern recognition problem, and because neural networks are good at pattern recognition, a number of researchers tried applying neural networks to image classification. The system’s accuracy depends on the conditions under which it is evaluated: under sufficiently narrow conditions almost any system can attain human-like accuracy, but it is much harder to achieve good accuracy under general conditions.

Creating a system to classify real world imagery from unclear dataset is a complex task. In general, image data set is affected by a number of factors such as positioning of the camera and the subject, lighting, and location. As a result, two image datasets belonging to the same subject can be very different from each other, such as differences in shape, colour, and texture, and objects can be scaled or translated within an image. Classification usually consists of four steps:

1. Pre-processing; image processing is a collection of techniques for the manipulation of images by computers.
2. Training; selection of the feature, which best describes the pattern.
3. Decision; choice of an appropriate algorithm for comparing the image patterns with the target patterns.
4. Assessing the accuracy of the classification.

The classification system is divided into supervised and unsupervised systems. In supervised classification, examples of the information classes of interest in the image are identified by a teacher. In unsupervised classification, a large number of unknown pixels are examined and divided into a number of classes based on natural groupings present in the image values.

There are a number of techniques that are applied for object recognition, from image-based methods through to a range of approaches that decompose objects into parts. Part-based models have been used for object recognition and in learning models from example images [7] [8] [9].

Particular emphasis has been on the recognition of general classes of objects utilizing models that are learned from specific examples. This approach offers some advantages like the individual parts can be trained separately and modelled more or less independently. As a result, an object that is partly occluded can be classified correctly as long as the visible parts can be recognized.

Because of neural network nature, and the fact that using RGB images makes computation more complex, so images are converted to Grayscale or binary form before introducing them to the system. Histograms and other non-parametric forms have been used in visual tracking. One of the classical approaches is Otsu's method that is used to perform histogram shape-based image thresholding, and the reduction of a graylevel image to a binary image. The classical approach uses the graylevel histogram to select the threshold by using an intensity histogram to find a threshold value for the entire image [10].

Fuzzy Neural Networks:

Different types of fuzzification techniques used in neural networks such as Fuzzification of training data binary class membership values [11], Fuzzification of parameters in a feedforward neural network (FNN) [12], Fuzzification of temperature in short-term load forecasting (STLF) using a Multi-Layered LSTM model [13], Selective fuzzification of the input space in Intuitionistic Semi-Fuzzy Neural Network (ISFNN) [14], Fuzzification of spike neural networks (SNNs) using interval type-2 fuzzy sets (IT2FS) [15] and Fuzzy Min-Max Neural Networks (FMNN) [16].

Fuzzy sets, which are the main concept behind fuzzy logic, were introduced as an extension of set theory for representing and manipulating fuzzy data. This data was described as data that does not have precisely defined criteria of membership [17].

FMNN (Fuzzy Min-Max Neural Networks) technique is one of approaches that is used to synthesize fuzzy logic and neural networks to generate a fuzzy neural network that is able to tolerate imprecision and uncertainty.

This approach uses min-max hyperboxes to define fuzzy sets, with membership functions determining the degree to which an input pattern belongs to a class [16].

Different neural network architectures come with different approaches. One of the architectures proposed was for a neural network based fuzzy logic decision system. This architecture is composed of three layers. The first layer is the input layer, which transmits the inputs to the next layer. The second layer is used for fuzzification, this layer maps inputs to membership functions. The last layer is the output layer. Inputs in this model represent features while outputs represent categories [3]. Another approach uses fuzzy neurons to either perform an operation (t-norms) or an operation (s-norms). The first layer, input layer, consists of neurons whose activation function is the membership function of the fuzzy sets defined for inputs. The second layer is of fuzzy logic neurons. Each neuron performs a weighted aggregation of some of the first layer outputs. Finally, the output layer computes the network output using output layer weights and second layer output [18].

Experimental Methodology:

The structure of experiments will be described in this section. This proposal consists of two main components:

Firstly: Obtaining the optimal neural network construction.

The network structure is chosen, data is prepared, and initial values are provided for weights and biases. The network is then trained and tested with backpropagation learning algorithm, and the error is evaluated. If the error is not satisfactory, the structure is adjusted or weights and biases are modified. When the error is acceptable, weights and biases are fixed, and the network can be used to predict.

Secondly: Fuzzification

Applying fuzzy logic to selected optimal neural network construction. Tuning membership function, evaluating results, retuning if necessary and if it is acceptable, the network can be used for prediction.

Fuzzy Min-Max Neural Networks framework can be divided to three stages (Figure .1). The first is the preparation of data. The second identifies the classifier structure. The third is applying the fuzzification to the optimal neural network.

Stage 1: The preparation of data

1.1 Collecting data

Cheetah images were collected from Papers With Code Repository [19].

1.2 Resizing the image

The images were resized in order to get most of the patterns of objects, which are part of the animal or background. The images have been resized to 330 x 225 pixels.

1.3 Converting images into Grayscale

The pieces of the images are in RGB format. RGB images are composed of three colour channels: red, green and blue. Every pixel is represented by three numbers; each channel has its own value, which leads to an increase in the number of inputs and workload. Consequently, the images were converted to Grayscale.

1.4 Normalizing dataset

Normalization modifies each value of the dataset to fit within the range between 0 and 1. A Max-Min normalisation technique was used [20], and the following formula was used in this respect:

V'(i)=V'(i)min+(V'(i)max- V'(i)min)*(V(i)-V(i)min)/(V(i)max- V(i)min) (1)

where, V(i) is the data value, V'(i) is an input value [V(i)max, V(i)min] is the initial range and [V'(i)max,V'(i)min] is the new range.

1.5 Converting dataset into binary

A binary image is a digital image that has only two possible values for each pixel, 0 or 1. This is done by comparing the pixel's value with a threshold. The threshold is selected based on the histogram method [21].

Stage 2: Identifying the classifier structure

This paper will consider neural networks trained using backpropagation algorithm. The backpropagation network consists of at least three layers; input, hidden and output layer. The activation of the input layer is propagated forward to the output layer through the intervening input to the hidden layer then from the hidden layer to the output [22].

Minimizing the net's output error is the goal of the training algorithm. Error can be defined as:

$$e = t - a \qquad (2)$$

Where: t is output's target, a is actual output.

The error metric for the net is:

$$E = E(w_1, w_2, ..., w_{n+1}) \qquad (3)$$

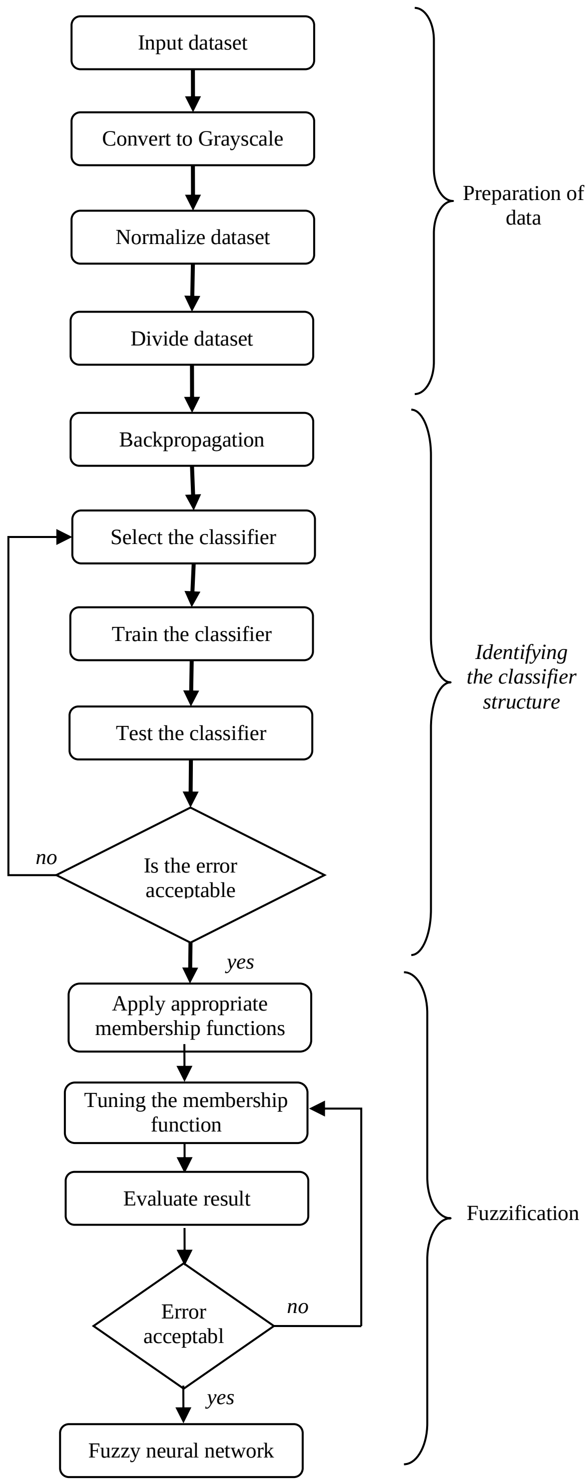

Fig .1 Neural Network Fuzzification Flowchart

By changing and adjusting the weights the error will decrease. To find the optimal weight vector, function (E) should be minimizing by gradient descent, (Figure 2), a method that is used to minimize the total error in the training process.

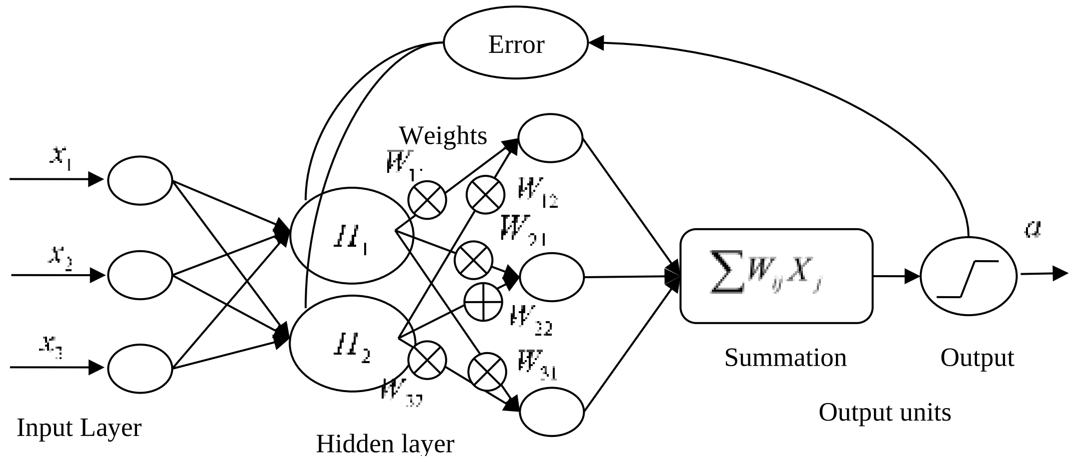

Fig .2 Backpropagation Network Architecture

Mathematically it is obtained by:

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} \qquad (4)$$

Where η is called the *learning rate*. Gurney [23] defines η as "it governs how big the changes to the weights are and, hence, how fast the learning takes place".

Now the error is the average error over all patterns:

$$E = \frac{1}{N}\sum_{\mathrm{P}=1}^{N} e^{\mathrm{P}} \qquad (5)$$

Where N = no. of patterns, and e^{P} is the error per pattern P.
From (2):

$$e^{\mathrm{P}} = t^{\mathrm{P}} - y^{\mathrm{P}} \qquad (6)$$

As mentioned above, backpropagation was derived from the Widrow-Hoff learning rule. Root mean squared error (RMSE) can be used to measure the error over all patterns [24]:

$$E = \frac{1}{N}\sum_{\mathrm{P}=1}^{N} \frac{1}{2}(t^{\mathrm{P}} - a^{\mathrm{P}})^2 \qquad (7)$$

In order to evaluate $\frac{\partial E}{\partial w_i}$, the whole training set is needed.

$$\partial E / \partial w_i = \frac{1}{N}\sum_{\mathrm{P}=1}^{N} \partial e^{\mathrm{P}} / \partial w_i \qquad (8)$$

This term is called *batch training*. The gradient per pattern is:

$$\frac{\partial e^{\mathrm{P}}}{\partial w_i} = -(t^{\mathrm{P}} - a^{\mathrm{P}})x_i^{\mathrm{P}} \qquad (9)$$

It is called *pattern training* [23]. Where: x_i^{ρ} is the i^{th} component of pattern P and $(t^{\mathrm{P}} - a^{\mathrm{P}})$ is referred to as δ *delta*. This term is either known as *delta rule* or pattern training regime.
As mentioned previously, backpropagation is based on gradient descent, considering the equations (7):

$$\Delta w_{ji\,\substack{OUTPUT\\UNITS}} = \alpha\sigma'(a_j)(t_j^{\mathrm{P}} - y_j^{\mathrm{P}})x_{ji}^{\mathrm{P}} \qquad (10)$$

Where j refers to one of the output nodes.

To calculate the ΔW_{ki} (hidden layer):

$$\Delta w_{ki} = \alpha\sigma'(a_K)\delta^k x_{Ki}^{\mathrm{P}} \qquad (11)$$

δ for hidden layer is :

$$\delta_k = \sigma'(a_k)\sum_{j\in I_k}\delta_j w_{jk} \qquad (12)$$

Hidden node may have a number of other output units that connect to it and take inputs from it. Consequently, the delta for hidden unit is:

$$\delta_k = \sum_{j\in I_k}\delta_j w_{jk} \qquad (13)$$

Where I_k is the set of nodes.

The backpropagation learning rule is defined as:

$$\Delta w_{ki} = \alpha\delta_k x_{ki}^{\mathrm{P}} \qquad (14)$$

Briefly, backpropagation aims to reduce the total error for the network by adjusting the error every time during the training process. Mathematically it is shown that by increasing the number of training the E_{total} will approach zero.

$$E_{total} = \lim_{p\to\infty}\frac{1}{N}\sum_{\mathrm{P}=1}^{N}\frac{1}{2}(t^{\mathrm{P}} - a^{\mathrm{P}})^2$$

And the Root mean squared error (RMSE) is then calculated using the following formula [28]:

$$RMSE = \sqrt{\sum_{p=1}^{n}\frac{(d_p - o_p)^2}{n}} \qquad (15)$$

Where:

d_p is the desired output at sample p.

o_p is the network output at sample p.

n is the total number of training samples.

The algorithm structure is shown in (Figure 2). Training aims to reduce the error value. The set of vectors that are presented for the period of training is called Training Set. Each training vector applied causes the weights to be adjusted slightly. This is done each epoch, one full pass through the training set. A measure of network performance during training is a training set classification accuracy, which is the percentage of vectors that are classified correctly.

At the end of training, a set of unseen patterns called a Testing Set is passed through the network in order to test the ability of the network to generalize. The classification accuracy of the testing set indicates how well the network is able to generalize.

One way to improve the neural network's ability to generalize is to train the networks on different training sets. This approach is known as cross-validation [25]. This suggests the following strategy: data set is divided into a number of equal sized divisions. One division is used for the test data, and the others are used for training. It is very important that the test set is not used as part of the training set. This allows the training algorithm to use nearly the whole data set for training, but is clearly very intensive. Hassoun [25] stated that the training is stopped when the error on the test set is at a minimum level, (Figure 3).

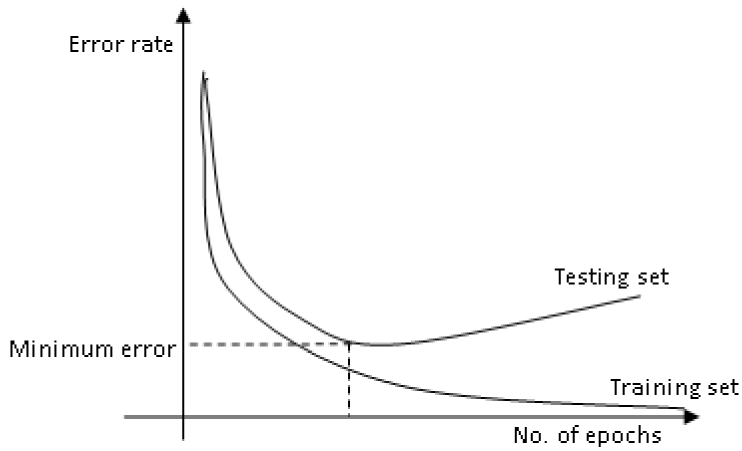

Fig 3 Cross-validation

A feedforward full-connection net was built to enable the classification of biological data. The number of input/output data items and the relation between them determine the network architecture. Because image has a large amount of input data with no clear relation to output, backpropagation neural network might be a good idea.

As mentioned previously, the backpropagation network consists of three units - input, hidden layers and output.

2.1 Input layer structure

The input neurons in input layer receive information from the outside world in the form of patterns or signals [26]. The outputs of this layer are then directly sent to the next layer, which is usually the hidden layer. The number of neurons in the input layers depend on the size of patch sample system. More input nodes mean more characteristics and information to determine the class. The input layer can be expanded by adding more new data sources as neurons. However, this expansion in input layer size will increase the computation time significantly, for example, if the input data is doubled then the training time will be four times more than the initial time. For that reason, adding new data sets should be considered only if they contribute to a significantly improved classification [26]. In $n \times n$ pixels patch sample system, there are n^2 inputs to feed into the first hidden layer.

2.2 Number of Neurons and Hidden layers

An infinite number of network structures may be made for a specific dataset. A backpropagation network with more than one hidden layer is sufficient for some applications, but one hidden layer is also sufficient. The presence of a hidden layer in a neural network will make data linearly separable. The addition of more than one hidden layer increases the distance between the classes of data (27). On the other hand, a higher number of nodes in the hidden layer causes slower convergence with a smaller error

[25]. However, continuing to increase the number of nodes will lead to an increase in the running time and not a decrease in errors. Through experiments, the network structure that gives the best result can be determined. This project focuses on exploring the impact of applying fuzzy techniques to artificial neural networks for biological data classification. The paper utilizes a maximum of two hidden layers in a backpropagation ANN.

2.3 Output layer

The output layer is responsible for producing information and signals to the outside world as a result. There is always one output layer in a neural network. During training, the backpropagation network was presented with binary output data. There was only one output variable in the training data set. In the cheetah recognition system, an output variable value of 1 was assigned to cheetah and a value of 0 to non-cheetah.

2.4 The learning rate

The learning rate is a common parameter in many of the learning algorithms, and affects the speed at which the network reaches the minimum error. In backpropagation, if the learning rate is too high, the system will either fluctuate around the minimum error or it will diverge completely. In contrast, if the learning rate is too small, the system will take a long time to reach the minimum error. For this project, the learning rate was selected to be 1 during all the experiments.

2.5 The network classification accuracy

The outputs from neural network are not binary. The neural network produces real values between 1 and -1, indicating whether or not the input contains the target. A threshold value of 0.5 is used during training to determine whether the output is 0 or 1. If the output is greater than 0.5, it is considered as 1, otherwise as 0.

The performance of the neural networks was evaluated based on the Root Mean Squared Root errors (RMSE) and the Classification Accuracy (CA). The classification accuracy is defined as the percentage of vectors that the network is able to classify correctly.

$$CA= \frac{\text{no.correct classifying vectors}}{\text{no. of whole dataset}} * 100 \tag{16}$$

2.6 The minimum error

The aim of the training network is to reach the minimum error. So, one of the most important parameters is the value of minimum error. 1% error was chosen as an acceptable percentage.

2.7 Stopping criteria

In this work, both the Root Mean Squared Error (RMSE) and Classification Accuracy (CA) are monitored for the testing set. An epoch is considered as good if the (RMSE) is lower than the smallest previous value and the CA value is higher than the largest previous value. When the training of the network has finished, the weights should be saved and then reloaded to use them in the testing session.

2.8 Evaluation stage

The final stage of the classifier system is the evaluation of optimal neural network architecture for each object, this stage can be done by using unseen image. A patch of $n \times n$ pixels is taken form the image in order to test the optimal neural network architecture. The output of black square is assigned to the object and grey square to the non-object. The patch of size, used during training, is applied in turn for the hall image area.

Stage 3: The fuzzification

Fuzzy Min-Max Neural Networks techniques is used for applying Fuzzification to the optimal neural network construction, which is obtained in the previous stage. This approach builds hyperbox fuzzy sets to classify data. Union of fuzzy set hyperboxes is a single fuzzy set class, where hyperboxes range from 0 to 1 along each dimension. By using the fuzzy min-max learning algorithm, the min-max points are determined by an n-dimensional box defined by a min point and a max point with a corresponding membership function.

The fuzzy min-max classification learning algorithm is divided into three steps:

1. Expansion: Determine the hyperbox that can be expanded. A new hyperbox is added if no expandable hyperbox is found.

2. Overlap Test: Determine whether there is any overlap between different types of hyperboxes.

3. Contraction: If there is an overlap between different types of hyperboxes, each hyperbox is adjusted to a minimum to eliminate the overlap.

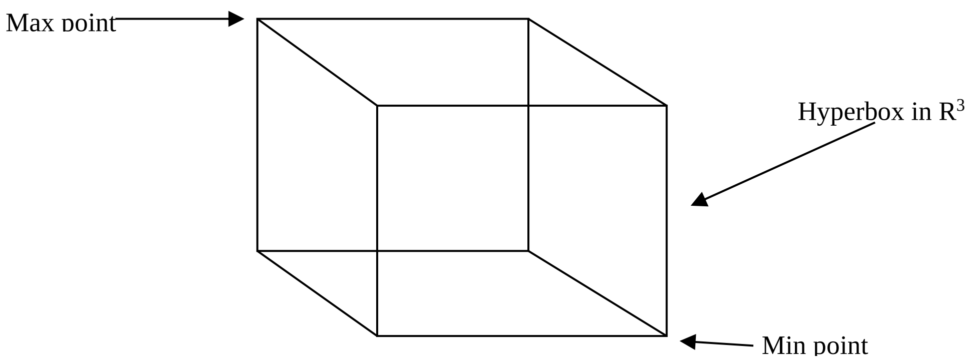

Fig 4 The min and max points

Figure 4 shows the illustration of the min and max points in a three-dimensional hyperbox, where the pattern space will be the n dimensional unit cube I^n.

A collection of hyperboxes forms a pattern class, and a membership function is associated with the hyperbox, it determines the degree to which any point X$\in R^3$ is contained within the box. The membership function for each hyperbox fuzzy set must describe the degree to which a pattern fits within the hyperbox.

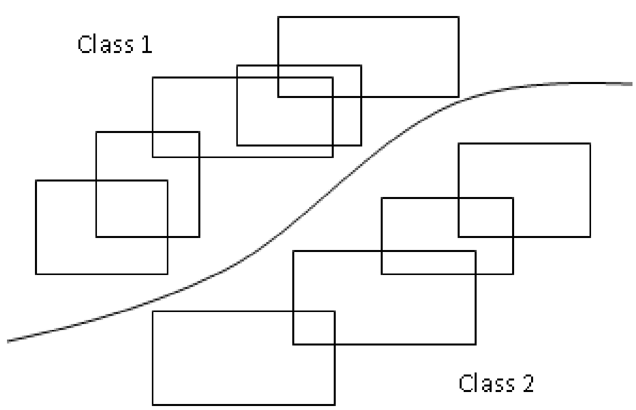

Fig 5 The aggregation of fuzzy min-max hyperboxes

Figure 5 shows the aggregation example of fuzzy min-max hyperboxes placed along the boundary of a two-class problem is illustrated.

The aggregation of several hyperboxes in I^2 is illustrated for a two-class problem.

Let each hyperbox fuzzy set, B_j, be defined by the ordered set:

$B_j = \{X, V_j, W_j, f(X, V_j, W_j)\} \qquad \forall X \in I^n$.

hyperboxes that have a range of values from 0 to 1 along each dimension.

the kth pattern class C_k defined by fuzzy set is defines as:

$$C_{k=\cup_{j\epsilon k} B_j}$$

where K is the index set of those hyperboxes associated with class k.

Fuzzy sets are defined for the inputs, where the first layer consists of neurons whose activation function is the membership function of the fuzzy sets defined for inputs. For each input a number of fuzzy sets are defined, the membership function of the fuzzy set is the activation function of the corresponding neuron. (The first layer neurons map each point in the set of inputs to a degree of membership). The second layer is of fuzzy logic neurons. Each neuron performs a weighted aggregation of some of the first layer outputs. Finally, the output layer computes the network output using output layer weights and second layer output.

Fuzzy sets are defined for the inputs, where the first layer consists of neurons whose activation function is the membership function of the fuzzy sets defined for inputs. For each input, a number of fuzzy sets is defined, the membership function of the fuzzy set is the activation function of the corresponding neuron. (The first layer neurons map each point in the set of inputs to a degree of membership). The second layer is of fuzzy logic neurons. Each neuron performs a weighted aggregation of some of the first layer outputs. The output of the hidden layer is activated signals, which are then transferred to the output layer using the same previous procedure.

Finally, the output of the network is generated and the fuzzy output is defuzzied using the following formula [28]

$$O = defuzzification\,(\tilde{O}) = \frac{1}{4}(O_1 + 2O_2 + O_3)$$

And the RMSE is then calculated using the following formula [28]:

$$RMSE = \sqrt{\frac{\sum(o-a)^2}{number\ of\ examples}}$$

In the backward phase, the deviation between the output and the target output is propagated backward. The error is calculated according to the following formula [28]:

$$\delta = O(1-O)(a-O)$$

Where:

δ is the error.

O is the output.

a is the target output.

Based on that, adjustments can be made on the connection weights.

Network learning stops when the RMSE is below a prespecified value, or a large number of epochs have al-ready been run [28].

In this work, Trapezoidal membership function has been used throughout all experiments [4]. However, this framework is not restricted to just Trapezoidal membership functions and any other membership function can be applied. A fuzzy region around threshold i is created by applying trapezoidal membership function f of domain $a \ldots d$ cutting through threshold i

where:

$$f(x,a,b,c,d) = \begin{cases} 0 & for\ x < a \\ \dfrac{x-a}{b-a} & for\ a \le x < b \\ 1 & for\ b \le x < c \\ \dfrac{d-x}{d-c} & for\ c \le x \le d \\ 0 & for\ x > \mathrm{d} \end{cases}$$

$a = \mathrm{i} - n\sigma$, $b = \mathrm{i} - \alpha$, $c = \mathrm{i} + \alpha$, $d = \mathrm{i} + n\sigma$

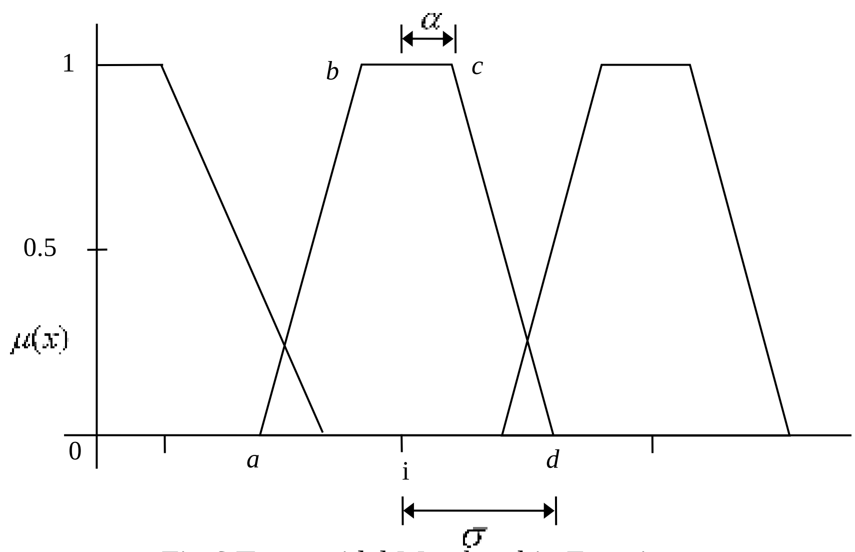

Fig 6 Trapezoidal Membership Function

where σ is the standard deviation, x is the value of attribute A (Figure .6).

$\alpha = m\,\sigma$ where m = 0.0001, 0.001, 0.01 0.1, n is a real number $n \to [0, \infty]$.

The initial membership functions are selected to give FNNs that are equivalent to ANN where $\sigma = 0$

.Experiments:

A. Datasets

This proposal framework is evaluated using AcinoSet dataset of free-running cheetahs in the wild, which were collected from Papers With Code Repository [19]. 900 patches samples of size 15 x 15 were extracted from the images dataset. These patches ware obtained to contain data that belongs to two classes only (1 or 0), and is used to predict the object in the image, which is in this case "cheetah" or "Non-cheetah".

B. Experimental framework

The cross-validation procedure [25] was used throughout all experiments. In *n*-fold cross validation, the complete dataset is randomized and randomly divided into *n* equally sized, disjointed blocks. Each block in turn is used as a test dataset, and the remaining *n-1* blocks are employed as a training dataset. This process is performed *n* times. In other words, the procedure is repeated until each block has been used once as a test dataset and *n-1* times as part of the training dataset. The classification work was done by two sets of experiments. The first phase of the experiments was conducted to obtain the optimal neural network construction, an ANN with standard back propagation algorithm is used. A number of experiments with different structures, weights and epochs are performed to enable the ANN to distinguish between correct and incorrect segmentation points. The experiments' second phase involved applying fuzzification techniques to the optimal neural network, resulting in Fuzzy Min-Max Neural Networks. A series of experiments were undertaken to determine the membership function degrees for each input, using Trapezoidal membership function. The training and testing strategy for the proposal framework followed the standard practice of 10-fold cross validation for all datasets [29].

Results and Discussion:

Tables I shows a comparison of results between Artificial Neural Network (ANN) and Fuzzy Min-Max Neural Networks (FMNN). The tables present the Root Mean Squared Error (RMSE) and the Classification Accuracy (CA). The structure of Artificial Neural Network was 225 input nodes, a first hidden layer with 9 neurons, a second hidden layer with 6 neurons and an output layer with 1 neuron. The results were achieved by applying the proposed framework to the Cheetah images within 10-fold cross-validation.

TABLE I - DATASET RESULTS

Classification Technique	RMSE	CA %
ANN	0.7791	71.16
FMNN	0.7424	75.69

The results obtained from Fuzzy Min-Max Neural Networks, produced by applying Fuzzification technique to Artificial Neural Network, show significant improvement in performance compared with the results that were obtained by Artificial Neural Network. The Fuzzy Min-Max Neural Networks induced from dataset increased the classification accuracy by 4.53 % compared to the Artificial Neural Network. The observation was made that when constructing a suitable fuzzy region around each input, there was a noticeable impact on the accuracy of the classification process. This observation highlights the significance of considering the establishment of an appropriate fuzzy region in order to improve the overall classification accuracy within the given context.

Conclusion:

This paper demonstrates a novel framework for the fuzzification of Backpropagation Neural Networks using Fuzzy Min-Max Neural Networks technique associated with Trapezoidal membership function. Patch-based technique was used to present images to the neural network. The experiments offer very promising results in using this kind of framework technique to increase the classification accuracy of Artificial Neural Network. The methodology behind this framework algorithm can be extended to any other Neural Networks learning algorithm. Further studies involve investigating the effects of the dataset characteristics in determining the most appropriate domain delimiters of membership function and fuzzy membership optimization.

References:

[1] Alexsander, I. and Morton, H., (1995). An Introduction to Neural Computing, Thomson Computer Press.

[2] Suzuki, K., (2013). Artificial Neural Networks: Architectures and Applications, Norderstedt, Germany: Books on Demand.

[3] Kulkarni, A. D. and Cavanaugh, C. D., (2000). Fuzzy Neural Network Models for Classification, Applied Intelligence, pp. 207.

[4] Gasir, F. Bandar, Z. Crockett, K. (2009), Elgasir: An algorithm for creating Fuzzy Regression Trees. FUZZ-IEEE International Conference on Fuzzy Systems, Jeju Island, Korea.

[5] Rafiei, H., & Akbarzadeh-T., M. (2023). Reliable Fuzzy Neural Networks for Systems Identification and Control. IEEE Transactions on Fuzzy Systems, 31, 2251-2263.

[6] Zhang, L., Shi, Y., Chang, Y., & Lin, C. (2023). Robust Fuzzy Neural Network With an Adaptive Inference Engine. IEEE transactions on cybernetics, PP.

[7] Ulusoy, I., & Bishop, C.M. (2005). Generative versus discriminative methods for object recognition. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2*, 258-265 vol. 2.

[8] Fergus, R., Perona, P., & Zisserman, A. (2005). A sparse object category model for efficient learning and exhaustive recognition. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 1, 380-387 vol. 1.

[9] Opelt, A., Pinz, A., Fussenegger, M., & Auer, P. (2006). Generic object recognition with boosting. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28, 416-431.

[10] Ulusoy, I., & Bishop, C.M. (2005). Generative versus discriminative methods for object recognition. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2*, 258-265 vol. 2.

[11] Tajti, T., (2020) Fuzzification of training data class membership binary values for neural network algorithms, Annales Mathematicae et Informaticae 52, 217-228

[12] Wu, H., Chen, T.T., & Chiu, M. (2021). Constructing a Precise Fuzzy Feedforward Neural Network Using an Independent Fuzzification Approach. Axioms, 10, 282.

[13] Ren, Z., Chao, C., Deng, Y., Zhang, W., Jun, W., & Zheng, R. (2020). Short-term load forecasting of multi-layer LSTM neural network considering temperature fuzzification. 2020 IEEE Sustainable Power and Energy Conference (iSPEC), 2398-2404.

[14] Terziyska, M., Todorov, Y., & Olteanu, M. (2016). Input space selective fuzzification in intuitionistic semi fuzzy-neural network. 2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), 1-7.

[15] Reid, D., & Muyeba, M.K. (2008). Fuzzification of Spiked Neural Networks. 2008 Second UKSIM European Symposium on Computer Modeling and Simulation, 135-140.

[16] Simpson, P.K. (1992). Fuzzy min-max neural networks. I. Classification. IEEE transactions on neural networks, 3 5, 776-86 .

[17] Zadeh, L.A., Klir, G.J., & Yuan, B. (1996). Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems - Selected Papers by Lotfi A Zadeh. Advances in Fuzzy Systems - Applications and Theory.

[18] Souza, P.V. (2018). Regularized Fuzzy Neural Networks for Pattern Classification Problems.

[19] Papers With Code Repository; available at https://paperswithcode.com/dataset/acinoset.

[20] Patel, C., Pandey, A., Wadhvani, R., & Patil, D. (2022). Forecasting Nonstationary Wind Data Using Adaptive Min-Max Normalization. *2022 1st International Conference on Sustainable Technology for Power and Energy Systems (STPES)*, 1-6.

[21] Low, A. (1991) *INTRODUCTORY COMPUTER VISION AND IMAGE PROCESSING*. McGraw-Hill Introduction (UK) Limited.
[22] Giri, S., & Joshi, B. (2021). Multilayer Backpropagation Neural Networks for Implementation of Logic Gates. International Journal of Computer Science & Engineering Survey.
[23] Mclean, D. (2000). Kevin Gurney, An Introduction to Neural Networks, University College London (UCL) Press, 1997. ISBN 1-85728-673-1 HB. xi+234 pages. Natural Language Engineering, 6, 203 - 204.
[24] Sammouda, R. (2008). How Magnification of the Root-Mean-Square Deviation (RMSD) Value Affects the Convergence Speed of Hopfield Neural Network Classifier.
[25] Hassoun, M. H. (1995) FUNDAMENTALS OF ARTIFICAIL NEURAL NETWORKS. The MIT Press.
[26] Smagt, P., & Krose, B. (2009). Introduction to Neural Networks.
[27] Fausett, L. (1994) FANDAMENTALS OF NEURAL NETWORKS. Prentice-Hall
[28] Chen, T. (2003). 'A fuzzy back propagation network for output time prediction in a wafer fab. Applied Soft Computing'. pp. 216 – 218.
[29] Williams, G., (2005). Data Mining Desktop Survival Guide. Togaware.com.