Hybridization of Fuzzy Min-Max Neural Networks with kNN for Enhanced Pattern Classification.

Anil Kumar and P.S.V.S Sai Prasad

March 1, 2021

Hybridization of Fuzzy Min-Max Neural Networks with kNN for Enhanced Pattern Classification

Anil Kumar[1] and P.S.V.S Sai Prasad[2]

[1] University of Hyderabad, Telangana, India
anilhcu@uohyd.ac.in
[2] University of Hyderabad, Telangana, India
saics@uohyd.ernet.in

Abstract. Fuzzy Min-Max Neural Networks (FMNN) is a single epoch learning Pattern Classification algorithm with several advantages for online learning. The information loss due to Contraction step of FMNN leads to several improvements in literature such as MLF, FMCN etc. These approaches do not use Contraction step and provide additional structures in FMNN for decision making in overlapped regions overcoming the problem of Contraction with the cost of an increase in training complexity of FMNN.This work proposes a hybridization of FMNN with kNN algorithm for achieving the ability to handle decision making in overlapped regions without altering the structure of FMNN. Comparative studies with existing approaches over benchmark decision systems have proved the utility of the proposed kNN-FMNN approach.

Keywords: Fuzzy Min-Max Neural Network · FMNN · Fuzzy Sets · Neural Networks · Classification · kNN · Hybrid System · MLF.

1 Introduction

In 1965, Zadeh [16] introduced the new concept called Fuzzy sets, to manipulate the imprecise data into the fuzzy pattern. The Fuzzy logic aims at creating approximate human reasoning that is helpful on cognitive decision making. Several Hybrid systems were developed with Fuzzy sets combining other soft computing models such as artificial neural networks, expert systems and genetic algorithm etc [6, 12, 14, 18, 19].

A hybrid system like the combination of the artificial neural network with fuzzy logic has proved their effectiveness in being helpful for real-world problems [6]. In 1992, Simpson [15] proposed Fuzzy Min-Max Neural Network (FMNN) classifier based on fuzzy hyperboxes. The union of fuzzy hyperboxes represents individual decision classes. A hyperbox is defined as a region in n-dimensional pattern space characterized by minimum points, maximum points and fuzzy membership function. FMNN learning algorithm computes the min-max points of hyperboxes to acquire knowledge. These placing and adjustment of hyperboxes create a granular structure of pattern in pattern space which is useful for pattern classification. This method also constitutes with several salient learning features like online learning, non-linear separability and non-parametric classification, thus, making FMNN more flexible.

FMNN has been applied successfully in different applications such as fault detection, lung cancer, medical data analysis, classification of music and text classification etc. [1, 4, 9–13, 20].

However, FMNN is facing problems due to contraint of the size of hyperboxes and contraction process which may lead to gradation error in classification [8, 17]. Several developments have been proposed for FMNN in order to overcome its limitation and for enhanced classification.

In 2000, Gabrys et al. [7] proposed a generalization and extension based on FMNN known as General Fuzzy Min-Max Neural Network (GFMNN) that incorporates a significant modification on conventional FMNN with the new fuzzy membership function and hyperbox expansion criteria. But they used the same contraction process as FMNN that tempered the acquired knowledge in the boundary region causing gradation error in classification.

Many researchers have achieved an innovative way to exclude the contraction process to retains overlapping information for better pattern classification. In 2004, Bargiela et al. [2] proposed a new classifier known as Inclusion/Exclusion Fuzzy Hyperbox classifier (EFC) using an inclusion hyperboxes that includes input patterns belonging to the same class, and exclusion hyperboxes (erroneous hyperboxes) that includes input patterns in a confusion region of a different class. However, this method resulted in the reduction in classification accuracy owing to the removal of exclusion hyperboxes.

In 2007, Nandedkar et al. [8] introduced a novel concept called Fuzzy Min-Max Neural Network classifier with Compensatory Neuron Architecture (FMCN). This method can protect the min-max points of confusion overlap region to enhance the learning algorithm as this information is highly significant for pattern classification. Although this method did not allow the overlapped hyperboxes to be expanded for next time which tends to increase in cardinality of hyperboxes, thus increasing the time and space complexity.

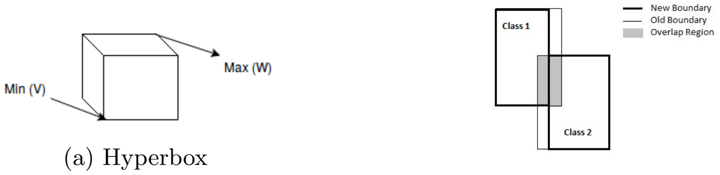
In 2007, Zhang et al. [17] proposed a new approach called as Data Core Based Fuzzy Neural Network (DCFMN) to overcome the limitation in FMCN with the help of geometrical centre and data core of hyperbox which can additionally benefit to handle noisy data. Hence, this method results in high classification accuracy than other prominent approaches like GFMNN, FMCN, EFC and also classical FMNN. In 2014, Devtalab et al. [3] proposed a new method called Multi-Level Fuzzy Min-Max Neural Network (MLF) classifier employing a multi-level tree structure to classify the pattern. Each level of model operates the smaller hyperboxes to handle the confusion region problem. It resulted in the enhancement of classification accuracy in the boundary region compared with existing approach GFMNN, EFC, FMCN, DCFMN and FMNN.

The above-mentioned improvement to FMNN has been obtained at an increased cost of training as additional is added to the simple three-layer architecture of FMNN. This motivated us to explore for the methodology for achieving the better classification accuracies without resorting to modification of the structure of FMNN. The proposed work introduces hybridization of k Nearest Neighbour algorithm in FMNN as kNN-FMNN classifier for dealing with overlapping regions without change of Neural Network structure of FMNN. We perform the experiments on benchmark dataset mentioned in [3] to establish the importance of kNN-FMNN. Comparative experiments are conducted against the existing approaches GFMNN, EFC, FMCN, DCFMN and MLF respectively for establishing the relevance of the proposed approach.

The remaining part in this paper is organized as follows: Section 2 briefly introduces the basics of FMNN for Classification. Section 3 gives the proposed kNN-FMNN algorithm. Section 4 provides the experiments and analysis of results. Paper ends with the conclusion.

2 Fuzzy Min-Max Neural Network

In 1992, Simpson [15] proposed the single-pass dynamic network structure with salient learning features as online learning, non-linear separability and non-parametric classification, to deal with pattern classification using fuzzy systems known as the fuzzy min-max neural network (FMNN). It is a supervised learning neural network that uses n-dimensional hyperbox fuzzy sets to represent pattern spaces [15]. FMNN learning process creates and adjusts hyperboxes in n-dimensions space for all decision classes in the pattern space.



(a) Hyperbox

(b) Overlapped region by two hyperboxes

Fig. 1

Each hyperbox is determined by min points, max points with corresponding fuzzy membership function, defined as:

$$B_j = \{X, V_j, W_j, f(X, V_j, W_j)\} \quad \forall X \in I^n \tag{1}$$

where X is the input pattern, V_j and W_j are the minimum and maximum points of B_j hyperbox. I^n is the n-dimensional unit pattern space.

The fuzzy membership function (b_j) defined in Eq. [2]:

$$b_j(X_h) = \frac{1}{2n}\sum_{i=1}^{n}[max(0, 1 - max\,(0, \gamma.min\,(1, x_{hi} - w_{ji}))) \\ + max(0, 1 - max\,(0, \gamma.min\,(1, v_{ji} - x_{hi})))] \tag{2}$$

where $X_h = (x_{h1}, x_{h2}, ..., x_{hn})$ is input pattern in n dimensional space and, $V_j = (v_{j1}, v_{j2}, ..., v_{jn})$ and $W_j = (w_{j1}, w_{j2}, ..., w_{jn})$ are the corresponding min points and max points for hyperbox B_j. γ is the sensitive parameter that regulates how fast the membership decreases as the distance between A_h and B_j increases.

FMNN training is a single epoch algorithm. For each training pattern, the learning involves three stages: 1) Expansion 2) Overlap Test 3) Contraction Process of Hyperboxes. During the training phase, when an input pattern enters into the network, the network tries to accommodate into one of existing same class hyperbox that gives full membership value. Otherwise, the network attempts to find the closest same label hyperbox which have the highest membership degree. The input pattern attempts to expand the particular hyperbox, bounded by expansion criteria given in Eq. [3]. The range of user-defined parameter theta in Eq. [3] is ($0 < \theta < 1$) and controls the volume of hyperbox.

$$\sum_{i=1}^{n}(max(w_{ji}, x_{hi}) - min(v_{ji}, x_{hi})) \leq n\theta \tag{3}$$

When the condition in Eq. [3] is satisfied, the Hyperbox expands to incorporate the input pattern by adjusting the min and max points by using the equations [4] and [5].

$$v_{ji}^{new} = min(v_{ji}^{old}, x_{hi}) \quad \forall i = 1, 2, 3, \ldots, n. \tag{4}$$

$$w_{ji}^{new} = min(w_{ji}^{old}, x_{hi}) \quad \forall i = 1, 2, 3, \ldots, n. \tag{5}$$

If the condition Eq.[3] is not satisfied, a point hyperbox is created with minimum and maximum value same as the input pattern.

After the expansion process, the overlap test [15] examines the overlap for the expanded hyperbox with all hyperboxes of other decision classes. Two hyperboxes don't overlap as long as there is at least one dimension at which they are not overlapping. If there is an overlap in all dimensions, then the test determine the dimension at which the smallest overlap occurs. If overlap test results in identifying the dimension having the smallest overlap, the contraction steps [15] adjust the hyperboxes along that dimension resulting in non-overlapping hyperboxes.

For example in Fig. [1b], both hyperboxes have overlapped in all dimensions. Overlap test determines that the least overlap exist horizontal dimension. The contraction step adjusts the hyperboxes along this dimension and resulting adjusting hyperboxes are given with a bold outline.

In testing an FMNN for a given test pattern x, the fuzzy membership of x into all the hyperboxes is computed. The test pattern x is classified to the decision class corresponding to the hyperbox achieving highest fuzzy membership.

3 Proposed kNN-FMNN algorithm

Classical FMNN algorithm [15], described in section 2, results in non-overlapping among the hyperbox of different classes. This results in information loss existing in the overlapping (boundary) region and results in the possibility of objects of one class being absolute members of hyperboxes of other class. The defuzzification of the overlapping region affects the generalizability of the FMNN. The existing approaches [2, 3, 7, 8, 17] dealing with the representation of overlapping region by avoiding overlapping and contraction process, are resulting in increasing the complexity of FMNN structure. The proposed kNN-FMNN approach aims at retaining the simple structure of FMNN while having the ability to deal with decision making in overlapping regions.

In kNN-FMNN approach, the kNN classification algorithm is used for decision making when a testing pattern falls into an overlapping region.

kNN classification algorithm doesn't have any training phase. For every input test pattern, the distance is evaluated between the test pattern with all the training patterns. The nearest k training patterns are selected as the nearest neighbours. Based on the classes of those k nearest neighbours, voting is conducted, and the test pattern is characterized to majority class of nearest neighbours. But in the presence of large training data, kNN requires significant testing time.

FMNN gives a natural way to group the nearest objects into the granular structure of hyperbox. So, using this we can restrict the space in which k nearest neighbour computation needs to be performed. This aspect we are employing in dealing with respect to overlapping region of FMNN testing algorithm.

The rest of section described the training and testing phases of kNN-FMNN algorithms given in algorithm [1] and [2] respectively.

3.1 Training of kNN-FMNN algorithm

Let DT represents the set of the training pattern, and FM represents the FMNN model be constructed. Initially, FM is empty, and as training proceeds, hyperboxes are added to the FM model extending the representation of hyperbox H in FMNN, given in section 2.1. The index list of objects belonging to H is maintained in our approach. For each input pattern x belonging to DT, only the expansion step is performed to preserve the overlapping region. In traditional FMNN based on expansion criteria, given in equation [3], hyperbox can expand non-uniformly in a different dimension as cumulative widths of all dimensions needs to be less than $n\theta$. This can result in a narrow strip of hyperboxes along few dimension and found to be unsuitable for decision making with respect to kNN. To overcome this, we have adopted the modified expansion criteria given by Gabriel et al. [7] in their work on General Fuzzy Min-Max Neural Network for Clustering and Classification (GFMNN). For a hyperbox H with V and W as min and max points, and x as the input pattern, the modified expansion criteria is given in Eqn. [6].

$$\forall_{i=1\ldots n}(\max(w_{ji}, x_{hi}) - \min(v_{ji}, x_{hi}) \leq \theta \tag{6}$$

This modified expansion criteria bounds every width of hyperbox on each dimension by θ and helps in generation of more uniform hyperboxes found suitable for kNN based decision making.

Algorithm 1: Training of kNN-FMNN

```
Input  : DT:Training Samples, $\gamma$, $\theta$
Output: Learning Model FM
Let FM: FMNN model (Initial empty);
for every $x$ in $DT$ do
    if FM.Belong(x) == True then
        H = FM.HMemb(x);
        FM.Save(H,x);
    else
        H = FM.HMemb(x);
        if $H$ exist then
            if $Exp_H(x)$ == True then
                FM.Expand(H,x);
                FM.Save(H,x);
            else
                FM.Create(x);
                FM.Save(H,x);
            end
        else
            FM.Create(x);
            FM.Save(H,x);
        end
    end
end
Return FM
```

For every training pattern x, the method $Belongs(x)$ finds fuzzy membership value of x with all hyperboxes pertaining to class of x using equation [2] and determine whether there exists a hyperbox giving full membership of one to x.

$$Belong(x) = \{\exists h \in HBS \mid Memb_h(x) == 1 \ \& \ class(x) == class(h)\} \tag{7}$$

where HBS represents a set of hyperboxes.

If $Belongs(x)$ is true, x is added to the hyperbox giving the full membership without resulting in any modificaiton of hyperbox. Otherwise using $HMemb(x)$ the hyperbox H giving the highest membership is obtained. In case of the existence of such H, if expansion criteria are satisfied the hyperbox H is expanded using equation [4] and [5] and object x stored as a member of resulting hyperbox. In case of expansion criteria not being met, or no hyperbox of a corresponding class existing, a point hyperbox is created using $Create(x)$, and x is added to the point hyperbox created.

3.2 Testing of kNN-FMNN Algorithm

Let DS be a set of testing sample. For every testing pattern x in DS, we compute the fuzzy membership value with all hyperboxes in FM. Because the overlapping among hyperboxes is allowed in the training phase, it is possible to obtain absolute membership of 1 to multiple hyperboxes. The $absMemb(x)$ returns all the hyperboxes giving full membership. If this set is empty, then the testing pattern is not belonging to any of hyperboxes and decision is taken like traditional FMNN testing by assigning the decision class corresponding to nearest hyperbox. In case $absMemb(x)$ return a non-empty collection of hyperboxes then the purity of collection is examined. The resulting collection is pure if only if all hyperboxes correspond to a single decision class and in which case, without ambiguity that class is assigned to the testing pattern. In case of impurity objects belonging to all this hyperboxes collected in $LocalSet$ function and kNN is performed locally for determining the decision class of x. The descriptions of functions used is given below:

$absMemb(x) = \{h \in HBS \mid Memb_h(x) == 1\}$: Collection of hyperboxes which have full membership for the object x.

$pure(absMemb(x)) = \{\forall h_1, h_2 \in absMemb(x) \mid class(h_1) == class(h_2)\}$: Collection of all hyperboxes that containing x correspond to same decision class.

$Members(h)$: Collecting the objects belonging to hyperbox h.

$LocalSet(absMemb(x)) = \bigcup_{h \in absMemb(x)}(Members(h))$.

$knnlocal(LocalSet(absMemb(x)), x)$: applying kNN methods on selected objects.

Algorithm 2: Testing of kNN-FMNN Algorithm

```
Input : DS: Testing Samples, Learning Model FM, k
for every $x$ in DS do
    Compute fuzzy membership of $x$ with all hyperboxes of FM model;
    if $|absMemb(x)| > 1$ then
        if pure(absMemb(x)) == True then
            Classify $x$ as $absMemb(x).class$ ;
        else
            HO = $LocalSet(absMemb(x))$ ;
            Classify $x$ using $knnlocal(HO, x).class$
        end
    else
        Classify $x$ to highest membership hyperbox class
    end
end
```

4 Experiments

4.1 Performance comparison with MLF [3] approach.

For evaluating the performance of kNN-FMNN, we have adopted the experimentaion model given in [3] for MLF algorithm. In [3], MLF algorithm's performance was compared with popular variants of FMNN such as FMNN [15], GFMNN [7], EFC [2], FMCN [8] and DCFMN [17] in the aspects of average mis-classification, average number of hyperboxes produced and computational time given in milliseconds. The experiments were conducted on a synthesized and standard datasets. Furthermore, The stratified 3-fold cross-validation technique was performed on the original dataset to comprehend the model's ability. The original dataset was partitioned into three subsets. In each iteration, one group was retained for the testing part and, the remaining two groups were used for training the model. This validation continues for three times (in each fold).

Table 1: Benchmark Datasets

Dataset	**Attributes**	**Objects**	**Decision Classes**
Iris	3	150	3
Breast Cancer	30	569	2
Glass	9	214	6
Ionosphere	32	351	2
Thyroid	21	7200	3
Wine	13	178	3
Parkinson	22	195	2
Ozone Layer	72	1848	2
Spambase	57	4597	2

In this paper, we have conducted experiments on kNN-FMNN following the same procedure given in [3]. The system configuration used for our experiments is CPU: Intel i5 7500, Clock Speed: 3.40GHz × 4, RAM: 8 GB DDR4, OS: Ubuntu 16.04 LTS 64 bit and Software: Rstudio Version 1.1.456. The configuration for experimentation in [3] for MLF and associated algorithms is CPU:core 2 dual, Clock Speed: 1.3 MHz and Ram: 4GB.

The experiments are conducted on nine benchmarks numeric dataset, collected from the UCI machine learning repository [5] which were used in MLF [3] experimentation. The description of numeric datasets used is shown in Table [1]. The synthesized datasets used in [3] were not experimented because of unavailability of datasets. Here, we employed different expansion criterion (theta = 0.2 and 0.3) on kNN-FMNN with γ as 0.4 and 'k' in kNN is set to 3.

All experiment results of the kNN-FMNN with other FMNN methods are listed in Table [2]. The results given for MLF and associated algorithms are reproduced from [3] for comparison. The best result under each category for each dataset is shown in boldface.

Table 2: Comparative Experiment Results

Dataset	Method	Classification (%)			No. of	Time
		Max	Min	Average	Hyperbox	(ms)
Iris	GFMNN	81.33	96.67	86.33	37.5	175
	EFC	88.67	96.67	91.31	42.0	54
	FMCN	90.00	97.33	92.76	63.6	44
	DCFMN	92.11	97.33	94.66	37.7	50
	MLF	**95.33**	97.33	**96.33**	56.3	60
	kNN-FMNN($\theta = 0.2$)	94.0	96.0	94.66	22.3	20
	kNN-FMNN($\theta = 0.3$)	94.0	**98.0**	96.0	**13**	15
Breast Cancer	GFMNN	55.51	95.74	78.35	154	1586
	EFC	58.44	94.27	79.34	183	629
	FMCN	68.58	95.30	83.16	244	472
	DCFMN	84.15	95.74	90.54	190	416
	MLF	93.69	96.48	95.59	227	1451
	kNN-FMNN($\theta = 0.2$)	**94.66**	**96.82**	**96.13**	164	700
	kNN-FMNN($\theta = 0.3$)	94.70	96.85	95.6	**79.6**	347
Glass	GFMNN	43.48	66.67	49.79	56	583
	EFC	35.27	66.63	42.53	89	115
	FMCN	29.47	66.63	41.26	394	96
	DCFMN	29.47	66.67	39.97	77	114
	MLF	56.04	68.12	60.32	227	225
	kNN-FMNN($\theta = 0.2$)	**60.56**	**76.05**	**65.91**	49	41
	kNN-FMNN($\theta = 0.3$)	56.33	69.01	63.54	**35.6**	33
Ionosphere	GFMNN	42.74	89.46	68.93	**120**	2163
	EFC	33.33	87.46	62.12	150	424
	FMCN	46.15	89.46	78.43	271	329
	DCFMN	79.54	**93.45**	87.77	132	448
	MLF	83.76	**93.45**	89.30	184	909
	kNN-FMNN($\theta = 0.2$)	**88.03**	91.45	**90.02**	146.6	399
	kNN-FMNN($\theta = 0.3$)	87.17	90.59	88.88	121.3	308
Thyroid	GFMNN	30.59	85.39	61.37	57	345
	EFC	74.89	90.87	85.30	**66**	116
	FMCN	74.89	90.87	85.78	116	94
	DCFMN	74.89	93.17	89.76	**66**	103
	MLF	75.34	94.52	90.88	72	152
	kNN-FMNN($\theta = 0.2$)	**94.72**	**95.29**	**94.86**	327	16542
	kNN-FMNN($\theta = 0.3$)	94.12	94.37	94.26	255	13811
Wine	GFMNN	85.00	96.67	92.91	129	817
	EFC	92.22	96.67	94.47	133	234
	FMCN	92.22	97.33	94.47	183	160
	DCFMN	92.22	97.33	94.53	124	127
	MLF	92.22	97.33	94.61	133	274
	kNN-FMNN($\theta = 0.2$)	**93.33**	**100**	96.08	91	95
	kNN-FMNN($\theta = 0.3$)	**93.33**	**100**	**96.64**	**55.6**	63
Parkinson	GFMNN	76.41	82.97	80.83	98	786
	EFC	40.77	78.97	74.66	111	256
	FMCN	40.77	82.56	75.54	207	216
	DCFMN	67.70	83.08	78.97	99	265
	MLF	77.95	84.10	83.49	111	302
	kNN-FMNN($\theta = 0.2$)	**87.69**	**96.92**	**92.30**	81.6	163
	kNN-FMNN($\theta = 0.3$)	86.15	**96.92**	91.79	**53.3**	109
Ozone Layer	GFMNN	41.66	92.16	79.84	929	61481
	EFC	77.13	92.95	87.38	1031	27221
	FMCN	90.05	92.87	91.89	2759	30850
	DCFMN	73.79	92.60	87.51	988	24694
	MLF	89.92	92.78	91.12	1314	49699
	kNN-FMNN($\theta = 0.2$)	96.80	96.42	**97.41**	1034.6	28436
	kNN-FMNN($\theta = 0.3$)	**96.94**	**96.75**	96.32	**531**	14208
SpamBase	GFMNN	39.90	83.39	54.15	293	86696
	EFC	43.21	81.04	56.78	700	12988
	FMCN	67.67	86.91	75.42	4982	6310
	DCFMN	71.95	86.91	80.70	693	98644
	MLF	82.89	89.39	88.41	7614	76929
	kNN-FMNN($\theta = 0.2$)	**89.29**	88.31	88.68	550	12686
	kNN-FMNN($\theta = 0.3$)	87.40	**90.14**	**89.21**	**254.3**	4974

4.2 Analysis of results

The computational complexity of FMNN training algorithm is proportional to the cardinality of hyperboxes created. In addition to the cardinality of hyperboxes, the cost of complex structures such as a compensatory neuron, exclusion hyperboxes and hierarchical layers in algorithms like FMCN, DCFMN and MLF increases the complexity. The computational time reported in table [2] validates the same as kNN-FMNN achieved training much lesser time compared to MLF and other approaches. The computational efficiently of kNN-FMNN is due to adapting FMNN with only the expansion step and also achieving much lesser cardinality of hyperboxes compared to other approaches.

From Fig. [3], it is observed that the average number of hyperbox creation is less than other FMNN methods except for thyroid dataset because of sparsity in the dataset. As the number of hyperboxes created thyroid dataset is more, the computation training time is higher only for this dataset.

Fig. [2] depicts the experimental result of classification accuracy. It is observed that in all the datasets, kNN-FMNN with theta (0.2 or 0.3) achieved similar or better classification accuracy compared to MLF and other approaches.

kNN-FMNN has achieved significantly better classification accuracies in Parkinson, Ozone layer, Glass and Thyroid datasets. For example for Parkinson dataset, by using the kNN-FMNN algorithm with user-defined parameter ($\theta = 0.2$), obtained 92.3% of average classification accuracy with 81 average number of hyperboxes creation respectively, whereas MLF gave 83.49% accuracy along with 111 hyperboxes.

We have experimented kNN-FMNN with several theta (θ) values, and all the results were not reported due to space contraint. It is observed that for less theta values such as 0.02, the cardinality of hyperboxes is huge and for high theta values such as 0.9, the cardinality of hyperboxes is less in the cost of misclassification. The best result (minimization of the cardinality of hyperboxes, maximation of classification accuracy) are obtained for theta values between 0.2 to 0.3, and the same is recommended.

5 Conclusion

Several improvements were proposed for the Fuzzy Min-Max Neural Network to overcome limitations arise due to contraction step. These extensions have resulted in adding additional complexity to FMNN thus increasing the training time. This work proposed kNN-FMNN as hybridization of FMNN with kNN for overcoming the contraction step in FMNN. The proposed approach resulted in building the classification model with a fewer number of hyperboxes and achieving good classification accuracy by utilizing kNN locally for disambiguating classification decision in the overlapping region. The experimental results have established that kNN-FMNN achieved better classification accuracy than existing approaches such as MLF, FMCN, DCFMN, EFC, GFMNN in less computation

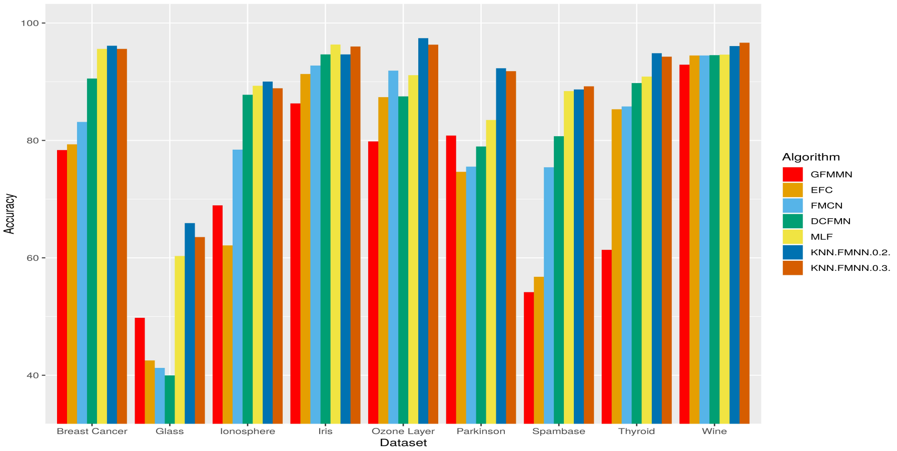


Fig. 2: Comparison of Average Classification Accuracy

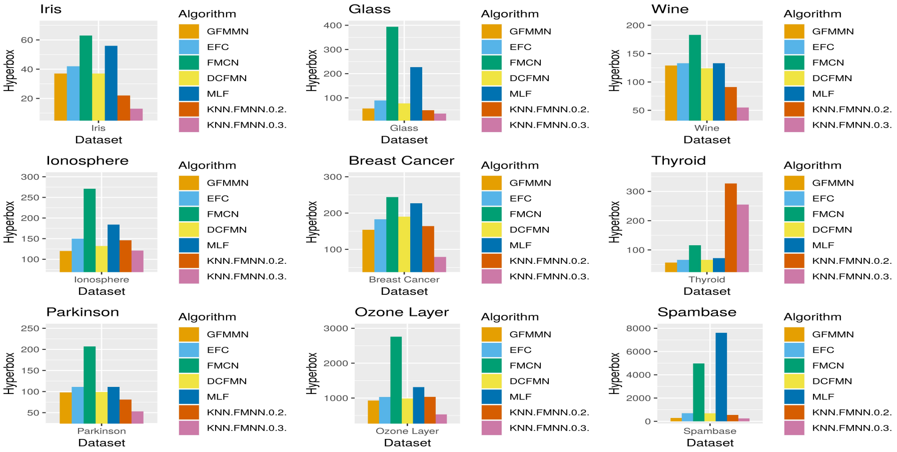


Fig. 3: Comparison of Average Hyperbox Size

time. In future, attempts will be done for proposing parallel and distributing kNN-FMNN for achieving scalability in large-scale decision systems.

References

1. Ahmed, A.A., Mohammed, M.F.: Sairf: A similarity approach for attack intention recognition using fuzzy min-max neural network. Journal of Computational Science **25**, 467 – 473 (2018)
2. Bargiela, A., Pedrycz, W., Tanaka, M.: An inclusion/exclusion fuzzy hyperbox classifier. KES Journal **8**, 91–98 (08 2004)

3. Davtalab, R., Dezfoulian, M.H., Mansoorizadeh, M.: Multi-level fuzzy min-max neural network classifier. IEEE Transactions on Neural Networks and Learning Systems **25**, 470–482 (2014)
4. Deshmukh, S., Shinde, S.: Diagnosis of lung cancer using pruned fuzzy min-max neural network. In: 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT). pp. 398–402 (Sept 2016)
5. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
6. Fullr, R.: Introduction to neuro-fuzzy systems. Advances in soft computing, Physica-Verlag (2000)
7. Gabrys, B., Bargiela, A.: General fuzzy min-max neural network for clustering and classification. IEEE Transactions on Neural Networks **11**(3), 769–783 (May 2000)
8. Nandedkar, A.V., Biswas, P.K.: A fuzzy min-max neural network classifier with compensatory neuron architecture. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. vol. 4, pp. 553–556 (Aug 2004)
9. Quteishat, A., Lim, C.P.: Application of the fuzzy min-max neural networks to medical diagnosis. In: Lovrek, I., Howlett, R.J., Jain, L.C. (eds.) Knowledge-Based Intelligent Information and Engineering Systems. pp. 548–555. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
10. Quteishat, A., Lim, C.P.: A modified fuzzy minmax neural network with rule extraction and its application to fault detection and classification. Applied Soft Computing **8**(2), 985 – 995 (2008)
11. Sadeghian, P., Wilson, C., Goeddel, S., Olmsted, A.: Classification of music by composer using fuzzy min-max neural networks. In: 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST). pp. 189–192 (Dec 2017)
12. Sayaydeh, O.N., Mohammed, M.F., Lim, C.P.: A survey of fuzzy min max neural networks for pattern classification: Variants and applications. IEEE Transactions on Fuzzy Systems pp. 1–1 (2018)
13. Seera, M., Wong, M.D., Nandi, A.K.: Classification of ball bearing faults using a hybrid intelligent model. Applied Soft Computing **57**, 427 – 435 (2017)
14. Shuang, F., Chen, C.L.P.: Fuzzy restricted boltzmann machine and deep belief network: A comparison on image reconstruction. In: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 1828–1833 (Oct 2017)
15. Simpson, P.K.: Fuzzy min-max neural networks. i. classification. IEEE Transactions on Neural Networks **3**(5), 776–786 (1992)
16. Zadeh, L.: Fuzzy sets. Information and Control **8**(3), 338 – 353 (1965)
17. Zhang, H., Liu, J., Ma, D., Wang, Z.: Data-core-based fuzzy minmax neural network for pattern classification. IEEE Transactions on Neural Networks **22**(12), 2339–2352 (Dec 2011)
18. Zhou, S., Chen, Q., Wang, X.: Fuzzy deep belief networks for semi-supervised sentiment classification. Neurocomputing **131**, 312 – 322 (2014)
19. Zimmermann, H.: Fuzzy Sets and Expert Systems, pp. 185–222. Springer Netherlands, Dordrecht (2001)
20. Zobeidi, S., Naderan, M., Alavi, S.E.: Effective text classification using multi-level fuzzy neural network. In: 2017 5th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS). pp. 91–96 (March 2017)