Progressive granular neural networks with Class based granulation

D Arun Kumar*, K Padma Kumari*, Saroj K Meher†
* School of Spatial Information Technology, J N T University, Kakinada, India
dasariak@pes.edu
geologymadam@gmail.com
† Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore, India
saroj.meher@isibang.ac.in

***Abstract*—Granular neural networks (GNNs) process granulated data with neural networks. Class based (CB) granulation of input data considers the belongingness of each feature to the classes present in the data. Advancement in the sensor technology has produced large amount of data which is also called as stream of data. To process the stream of CB granulated data, the article proposes an adaptive neural network model which is named as class based progressive granular neural network (CBPGNN). Learning or updating weights in CBPGNN is based on back propagating the output error. This overcomes the disadvantage of evolving granular neural networks (EGNNs) in which updation of weights is done linearly. CBPGNN posses the advantages of both class based granulation and progressive granular neural network (PGNN). The performance of CBPGNN is tested on remote sensing and waveform datasets. The superiority of proposed model has been justified by comparing its performance with similar types of models. The performance of proposed model is measured using the indices like average accuracy and kappa coefficient.**

I. Introduction

Human beings perceive the environment and takes relevant decisions. Their ability of making decisions is based on granulation. A granule is defined as a group of objects or class of numbers or a set of concepts [1], [2]. The process of grouping the objects based on the characteristics like similarity is called granulation [3]. Human brain process these granules efficiently because of its internal structure. Inspired with this activity of human brain, granular neural networks are used to process the input data and perform the classification of data. This is called as granular computing (GrC). GNNs are used to process granulated data with the help of neural networks (NNs). The architecture and functional behaviour of NNs is similar to human brain [4]. Granular computing is used to build efficient computational models that handle huge amount of data [5], [6]. There are various methods of granulating the feature space. Pal and Mitra [7] have suggested class non contingent (CNC) method of granulating the input data. In CNC method of granulation, the input patterns are granulated in to three linguistic granules called *Low*, *Medium*, and *High*. These granules are mathematically represented using fuzzy set theory [8], [9]. Furthermore, CNC method of granulation was used in several applications, e.g., [10], [11], [12], [13], and [14]. An improved method of granulating input data was suggested by Pal *et al.* [6], which is named as class based (CB) granulation. In CB granulation, each feature of a pattern is represented by its belongingness to the classes present in the data. CB granulation improves the classification accuracy of classifier. This is better than CNC method of granulating input data. [6], [14].

A pattern is associated with features [15]. Pattern classification is defined as labelling the given pattern to a specific class. There are different types of systems that learn from data and perform classification of data. Learning process of a system from the data is termed as the process of defining decision boundary among the classes in the feature space. One of the efficient methods of defining decision boundary among classes is by partitioning the feature space in to granules. The quality of defining the decision boundary depends on the size and the number of granules used for the partition of feature space. Traditional learning systems define the decision boundary with the number of granules and the size of the granules being static. In order to process streams of data, the size and the number of granules should be dynamic such that the system can adapt possible new information contained in the data. The system that is used to process the dynamic granules which are generated from the streams of data are called evolving system. There are various types of evolving systems which are suggested by scientists. Carpenter *et al.* [16] suggested an adaptive network model with incremental learning for real time problems solving. Kasabov [17] has suggested an evolving system to classify unlabelled data streams. The architecture of the system has been defined to adapt the incoming data. Evolving fuzzy neural network [18] with incremental learning has been suggested to accommodate new input data through local element tuning.

Daniel *et al* [19] suggested an advanced evolving system which is called as evolving granular neural network to classify the stream of data. The structure of EGNN is adaptable in dynamic environment. The architecture of EGNN changes according to the incoming data. EGNN acquires the new information within the data by adjusting the size and the number of granules. This property of evolving granular neural networks makes them to learn from the input pattern with a single pass. The learning process in EGNN is based on updating the synaptic weights linearly, which is independent of data. In order to define an evolving system in which updation of weights is based on back propagating the output error and the learning process depends on the data, we were motivated to suggest an adaptive network model which is named as progressive granular neural network to process the stream of CB granulated data. The performance of proposed model is better than the existing evolving systems because of its learning ability from the CB granulated patterns. The performance of CBPGNN is tested on remote sensing and waveform datasets. Its superiority over similar models has been justified with the help of performance metrics like average accuracy and kappa coefficient.

II. Granulation of input data

Granulation provides scope for representing the input data in the form of linguistic variables [20], [21]. Processing linguistic variables using NNs provides transparency in the functional steps [22]. This produces meaningful networks called granular neural networks. There are two important methods of granulating the input data. These are given as class non-contingent granulation and class based granulation.

A. Class non-contingent granulation

In CNC granulation, each feature of input patterns is represented by its belongingness to three fuzzy granules which are termed as *Low*, *Medium*, and *High* [7]. Belongingness of a feature is also called as membership of feature to the granule. A π - type membership function is used to compute the membership of each feature to the three granules. A pattern (P_1) with n number of features is represented as

$$P_1 = \begin{bmatrix} F_1 & F_2 & . & . & . & . & F_n \end{bmatrix}.$$

The membership of a feature F_k to a particular granule is given as:

$$\mu(F_k; c, \alpha) = \begin{cases} 2(1 - \frac{\|F_k - c\|}{\alpha})^2, & \text{for } \frac{\alpha}{2} \leq \|F_k - c\| \leq \alpha \\ 1 - 2(\frac{\|F_k - c\|}{\alpha})^2, & \text{for } 0 \leq \|F_k - c\| \leq \frac{\alpha}{2} \\ 0, & \text{otherwise.} \end{cases}$$

α is the radius of granule with c as the central point. The membership of a feature F_k is a maximum value *1* at the center c and it gradually decreases on either side of membership function with *0.5* at the crossover points. This results in the partition of feature axis in to three granules. The center and the radius of each fuzzy granule is computed by using the maximum ($F_{k_{max}}$) and minimum ($F_{k_{min}}$) values of each feature and is given as:

$$\alpha_{med.}(F_k) = \frac{1}{2}(F_{k_{max}} - F_{k_{min}})$$

$$c_{med.}(F_k) = F_{k_{min}} + \alpha_{med.}(F_k)$$

$$\alpha_{low}(F_k) = \frac{1}{S}(c_{med.}(F_k) - F_{k_{min}})$$

$$c_{low}(F_k) = c_{med.}(F_k) - \frac{1}{2}\alpha_{low}(F_k)$$

$$\alpha_{high}(F_k) = \frac{1}{S}(F_{k_{max}} - c_{med.}(F_k))$$

$$c_{high}(F_k) = c_{med.}(F_k) + \frac{1}{2}\alpha_{high}(F_k).$$

S is a parameter that controls the extent of overlapping among the three fuzzy sets. This results in representing a pattern (P_1) with n number of features in to *3n* granulated features. Each feature of a pattern posses membership to three fuzzy granules along every axis, and this is given as :

$$P_1 = \begin{bmatrix} \mu_{Low}(F_1), & \mu_{Med.}(F_1), & \mu_{High}(F_1), \\ \cdots, & \cdots, & \cdots, \\ \mu_{Low}(F_n), & \mu_{Med.}(F_n), & \mu_{High}(F_n) \end{bmatrix}.$$

The values such as $\mu_{Low}(F_n)$, $\mu_{Med.}(F_n)$, and $\mu_{High}(F_n)$ are the memberships of feature F_n to the three fuzzy granules *Low*, *Medium* and *High*, respectively.

B. Class based granulation

CNC granulation is limited to three fuzzy granules and it does not consider class wise belongingness of each feature during granulation. To overcome this limitation of CNC granulation, Pal *et al.* [6] suggested CB granulation method in which each feature is represented in terms of its membership with respect to the classes present in the data. In class based granulation, the membership of a feature F_k to a specific class is given by:

$$\mu(F_k; x, r, y) = \begin{cases} 0, & \text{for } F_k \leq x \\ 2^{M-1}(\frac{F_k - x}{r - x})^N, & \text{for } x < F_k \leq p \\ 1 - 2^{M-1}(\frac{r - F_k}{r - x})^N, & \text{for } p < F_k \leq r \\ 1 - 2^{M-1}(\frac{F_k - r}{y - r})^N, & \text{for } r < F_k \leq q \\ 2^{M-1}(\frac{y - F_k}{y - r})^N, & \text{for } q < F_{k.} \leq y \\ 0, & \text{for } F_k \geq y. \end{cases}$$

The variables *p* and *q* are the two cross over points at which their membership is equal to 0.5, and *r* is the center of fuzzy granule at which the membership value is *1*. The values of *p*, *q*, and *r* are given as *r=mean(N)* (where *N* is the mean of feature *n* to a particular class), $p = r - (\frac{max(F_n) - min(F_n)}{2})$ and $q = r + (\frac{max(F_n) - min(F_n)}{2})$. The variables $min(F_n)$ and $max(F_n)$ are minimum and maximum values of the dataset for feature *n*. The points x and y are called extreme points, which are given as $x = r - (q - p)$ and $y = r + (q - p)$. Thus a pattern P_1 with *n* number of features and the dataset possesing *c* number of classes can be represented by *n* x *c* number of granulated features as :

$$P_1 = \begin{bmatrix} \mu_1^1(F_1), & \mu_2^1(F_1), & \mu_c^1(F_1), \\ \mu_1^1(F_2), & \mu_2^1(F_2) & \cdots, \\ \mu_1^1(F_n), & \mu_2^1(F_n), & \mu_c^1(F_n) \end{bmatrix}.$$

III. Proposed method for classifying data stream

The number and the size of granules changes with incoming stream of data. This specify the necessity of a system that adapts to the changes in granules. In this process, a progressive granular neural network has been proposed to classify the

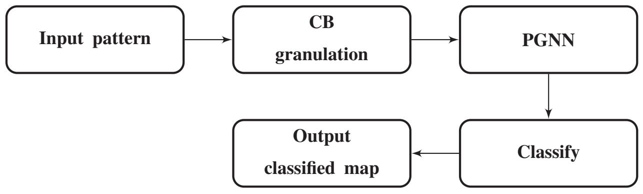

Fig. 1: Schematic flow diagram of proposed methodology

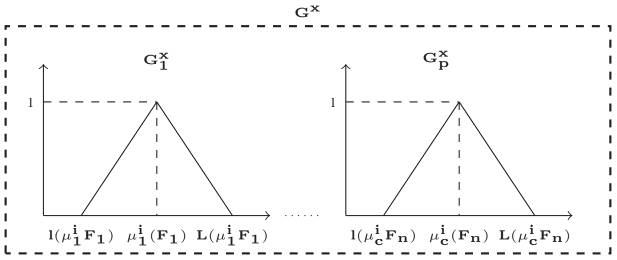
G^x
G^x_1
G^x_p
1
1
$l(\mu^i_1 F_1)$ $\mu^i_1(F_1)$ $L(\mu^i_1 F_1)$
$l(\mu^i_c F_n)$ $\mu^i_c(F_n)$ $L(\mu^i_c F_n)$

Fig. 2: Creation of new granule from CB granulated input

data which is granulated using CB method. The flow diagram of the proposed model is given in the Fig. 1. CBPGNN model consists of two levels of granulation. In the first level of granulation, all the input patterns are granulated using CB method (discussed in the section II B). Furthermore, the feature space that consists of these patterns is granulated using trapezoidal membership function. This is called the second level of granulation, which is used to define the architecture of PGNN. In the second level of granulation, the number of granules and the size of granules increases with the incoming stream of CB granulated data. This is supported with the help of creating and updating the granules by passing CB granulated patterns one by one. The granulation at the second level is divided in to two important parts: 1. Creation of new granules. 2. Updation of existing granules. Creation of a new granule is required when the incoming pattern belongs to a new class, or if the incoming pattern does not fit to existing granules. The process of creating a new granule is explained as: Let,

$$P_i = \begin{bmatrix} \mu^i_1(F_1), & \mu^i_2(F_1), & \mu^i_c(F_1), \\ \mu^i_1(F_2), & \mu^i_2(F_2) & \cdots, \\ \mu^i_1(F_n), & \mu^i_2(F_n), & \mu^i_c(F_n) \end{bmatrix},$$

be a CB granulated pattern. Creation of granule G^x along the axis $\mu^i_1(F_1)$ is done with the help of points l, λ, $\wedge$, and L, as $l(\mu^i_1(F_1)) = \mu^i_1(F_1) - \rho$, $\lambda(\mu^i_1(F_1)) = \wedge(\mu^i_1(F_1)) = \mu^i_1(F_1)$, and $L(\mu^i_1(F_1)) = \mu^i_1(F_1) + \rho$. ρ is the parameter used to control the size of granule. This is implemented along all the axes. Creation of a new granule from the CB granulated pattern, and the membership of pattern to the granule is shown in the Fig. 2 . Each granule which is created using CB granulated pattern is categorized to the class that the granulated pattern belongs to. Similarly, if the pattern P_j belongs to the existing granule G^x then the points along the axis $\mu^i_1(F_1)$ are adjusted to reflect the membership of pattern to the granule. This is given as:

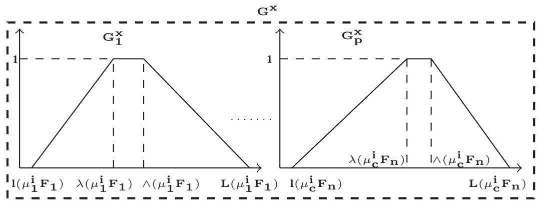
G^x
G^x_1
G^x_p
1
1
$l(\mu^i_1 F_1)$ $\lambda(\mu^i_1 F_1)$ $\wedge(\mu^i_1 F_1)$ $L(\mu^i_1 F_1)$
$l(\mu^i_c F_n)$ $\lambda(\mu^i_c F_n)$ $\wedge(\mu^i_c F_n)$ $L(\mu^i_c F_n)$

Fig. 3: Updating the existing granule to accommodate the current input

```
**if** $(\mu^i_1(F_1) - \rho) < \mu^j_1(F_1) < \mu^i_1(F_1)$ **then**
    $l(\mu^i_1(F_1)) = \mu^i_1(F_1) - \rho$
    $\lambda(\mu^i_1(F_1)) = \mu^j_1(F_1)$
    $\wedge(\mu^i_1 F_1) = \mu^i_1(F_1)$
    $L(\mu^i_1(F_1)) = \mu^i_1(F_1) + \rho$
**else**
    **if** $\mu^i_1(F_1) < \mu^j_1(F_1) < (\mu^i_1(F_1) + \rho)$ **then**
        $l(\mu^i_1(F_1)) = \mu^i_1(F_1) - \rho$
        $\lambda(\mu^i_1(F_1)) = \mu^i_1(F_1)$
        $\wedge(\mu^i_1(F_1)) = \mu^j_1(F_1)$
        $L(\mu^i_1(F_1)) = \mu^i_1(F_1) + \rho$
    **end if.**
```

Pictorial representation of updating an existing granule is shown in the Fig. 3. The structure of PGNN is built according to the number and the size of granules, where as the labelling of granulated pattern is based on its membership to the granules which are generated progressively. The architecture of PGNN which processes CB granulated patterns is shown in the Fig. 4. CBPGNN model consists of four layers which are named as input layer, progressive layer, hidden layer, and output layer. The number of nodes at the input layer is equal to the number of CB granulated features of a pattern (p=n x c, where n = number of features in a pattern and c = number of classes in the data). The number of progressive layer nodes is equal to the product of number of granulated features *(p)* and the number of granules *(m)* which are generated. The number of hidden layer nodes is equal to the total number of granules *(m)* which are created during evolving process. The number of output layer nodes is equal to the number of classes present in the data. Each node of input layer is connected to the granules of progressive layer. The output of a node at progressive layer represents the membership of each feature of a CB granulated pattern to the granules. The membership of a feature $\mu^1_1(F_1)$ to the granule G_x is computed using trapezoidal membership function and it is represented as A_{1x}. Each node of hidden layer aggregates the product of random progressive weights w_{ij} and membership of CB granulated pattern to a granule. The output of a hidden node H_1 is represented as B_{11}, which is given by

$$B_{11} = \frac{1}{1 + e^{-(\sum_{i=1}^{p} A_{i1} w_{i1})}}. \tag{1}$$

This is computed similarly for all the hidden nodes. Each output node of the network aggregates product of hidden

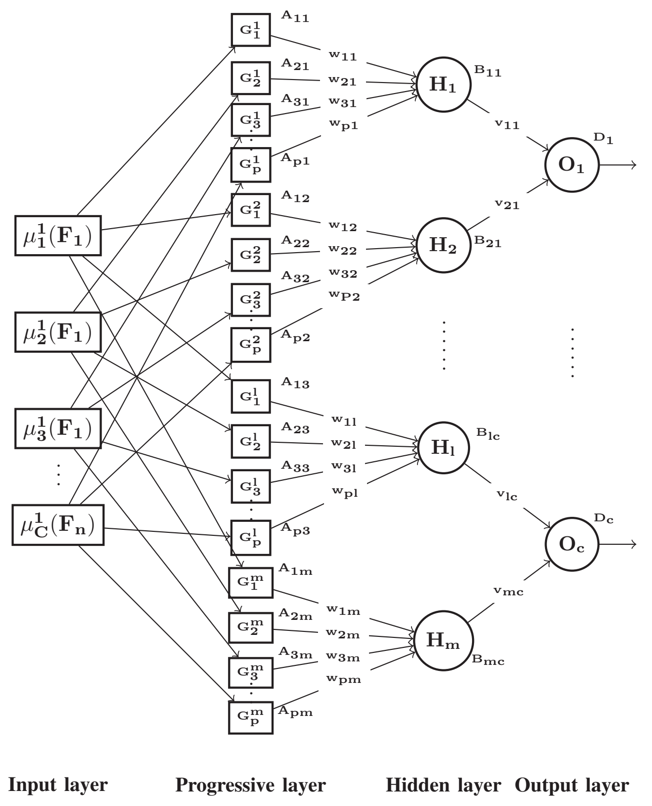
$\mu_1^1(F_1)$
$\mu_2^1(F_1)$
$\mu_3^1(F_1)$
$\mu_C^1(F_n)$
G_1^1 G_2^1 G_3^1 G_p^1
G_1^2 G_2^2 G_3^2 G_p^2
G_1^l G_2^l G_3^l G_p^l
G_1^m G_2^m G_3^m G_p^m
A_{11} A_{21} A_{31} A_{p1}
A_{12} A_{22} A_{32} A_{p2}
A_{13} A_{23} A_{33} A_{p3}
A_{1m} A_{2m} A_{3m} A_{pm}
w_{11} w_{21} w_{31} w_{p1}
w_{12} w_{22} w_{32} w_{P2}
w_{1l} w_{2l} w_{3l} w_{pl}
w_{1m} w_{2m} w_{3m} w_{pm}
H_1 B_{11}
H_2 B_{21}
H_l B_{lc}
H_m B_{mc}
v_{11} v_{21} v_{lc} v_{mc}
O_1 D_1
O_c D_c
Input layer
Progressive layer
Hidden layer
Output layer

Fig. 4: Structure of class based progressive granular neural network

weights v_{jk} and outputs of hidden nodes (each hidden node representing a granule) which belongs to respective class. The output D_1 at the output node (O_1) of network is given as:

$$D_1 = \frac{1}{1 + e^{-(\sum_{i=1}^{t} B_{i1} v_{i1})}}. \quad (2)$$

t is the number of granules that belongs to class *1*. Once the output at each output node is computed, labeling of a pattern is done based on winners take rationale mechanism. If the predicted class of pattern is different from actual class then the weights between the layers of CBPGNN are updated to reduce the overall error. These weights of CBPGNN are updated by back propagating the output error. The error at the output node O_k is given as

$$E_{O_k} = (T_k - D_k)(D_k)(1 - D_k), \quad (3)$$

where as the error at hidden node is given as

$$E_{H_{jk}} = (B_{jk})(1 - B_{jk})(E_{O_k}). \quad (4)$$

T_k is the target value at the output node *k*. These errors at the hidden and output nodes of CBPGNN are used to update the weights (w_{ij}) and (v_{jk}), which makes CBPGNN to learn from CB granulated data. Updation of weight between the *i*th progressive layer node and *j*th hidden node of CBPGNN is given as:

$$\Delta w_{ij} = \eta(A_{ij})(E_{H_{jk}}). \quad (5)$$

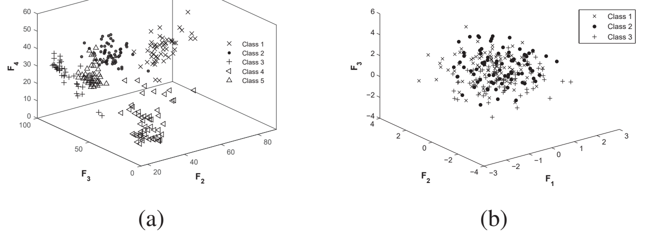

Fig. 5: (a) *IRS LISS III* data in the F_2-F_3-F_4 space, (b) *Waveform* data in the F_1-F_2-F_3 space

Similarly, the weight between the *j*th hidden layer node and the *k*th output node is updated according to the equation given below.

$$\Delta v_{jk} = \eta(B_{jk})(E_{O_k}), \quad (6)$$

η is the learning rate at which the CBPGNN learns from the input data. This process of creating and updating the granules by passing pattern by pattern, defining the structure of CBPGNN, and updating the weights between the layers of CBPGNN is dynamic that makes the system to process streams of CB granulated patterns. The weights of CBPGNN are updated in order to reduce the output error. The learning process of CBPGNN is based on input pattern and output error of the network. This is the advantage of CBPGNN in comparison with EGNN, because the weights in EGNN are updated linearly. In order to test the superiority of CBPGNN, six different types of models were built. These models were built based on the type of granulation method used to granulate the input patterns, and the type of NN used to classify the granulated patterns. Description about the six models are given below:

- *Model : Type of granulation used for input data + Type of neural network*
- *Model 1 : Ungranulated + EGNN*
- *Model 2 : CNC granulation + EGNN*
- *Model 3 : CB granulation + EGNN*
- *Model 4 : Ungranulated + PGNN*
- *Model 5 : CNC granulation + PGNN*
- *Model 6 : CB granulation + PGNN*

IV. Results and Discussion

Performance of all the six models is verified by testing them with two unique datasets. The datasets which are used in the present study is *(a)* taken from Indian remote sensing linear imaging self scanner (IRS LISS III) [23] and *(b)* the waveform dataset [24]. The patterns of these datasets are highly overlapping and their distribution in the feature space is given in the Fig. 5. IRSLISS III image of Mumbai city is given in the Fig. 6. Five land use/land cover classes with each class possessing 200 samples are considered from the remote sensing image. The description of remote sensing data samples is provided in the Table I. Waveform dataset consists of 5000 patterns with 21 number

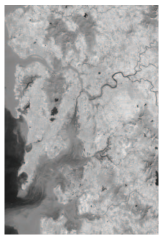

Fig. 6: IRS LISS III satellite image of Mumbai City

TABLE I: IRS LISS III dataset and number of samples

Landuse/Landcover type	Samples
Urban dense	*200*
Urban Sparse	*200*
Forest	*200*
Water	*200*
Agriculture	*200*

of features and three classes. The patterns of each class are generated from a combination of base waves. The features of patterns are corrupted with noise (mean=0,variance=1). The dataset consists of 33% of samples belonging to each class.

In the first level of granulation, all the samples are granulated using CNC/CB method. A strategic procedure has been followed in selecting the training samples from the CB granulated dataset. These training samples are used to define the architecture of model 2, model 3, model 5, and model 6, and also to train them. These models have been trained by using 20%, 40%, 60%, and 80% of CB granulated data samples and rest of the samples have been used for testing the models. Model 1 and model 4 are built and trained with ungranulated data. The performances of all the six models have been tested on IRS LISS III and waveform datasets, and the results are given in the Table II and Table III. Two important performance metrics like average accuracy and kappa coefficient have been used to compare the models. Comparison among the models is carried in order to show the superiority of proposed model (*model 6*) over the remaining models. These comparisons demonstrates the advantage of CB granulation at the first level of granulation, and the better learning ability of CBPGNN over EGNN. This is justified from the results given in the Table II and Table III. Performance of *model 3* is better than *model 2*

TABLE II: Experimental results of all the models for *IRS LISS III* data

		CNC granulation			*Class based granulation*	
Training (%)	*Model 1*	*Model 2*	*Model 3*	*Model 4*	*Model 5*	*Model 6*
20	86.20	87.19	87.92	87.24	87.84	88.86
40	86.44	87.32	87.98	87.48	87.94	89.02
60	86.48	87.54	88.03	87.52	88.12	89.26
80	86.92	87.63	88.24	87.62	88.20	89.34
Average	**86.51**	**87.42**	**88.04**	**87.47**	**88.02**	**89.12**
KC	0.831	0.842	0.849	0.840	0.845	0.853

and *model 1*, which shows the superiority of CB granulation over CNC granulation and ungranulated input data. This is

TABLE III: Experimental results of all the models for *Waveform* data

		CNC granulation			*Class based granulation*	
Training (%)	*Model 1*	*Model 2*	*Model 3*	*Model 4*	*Model 5*	*Model 6*
20	76.48	77.24	78.12	77.36	78.32	79.28
40	76.84	77.48	78.36	77.48	78.56	79.36
60	76.93	77.59	78.58	77.56	78.72	79.46
80	77.08	77.68	78.69	77.64	78.83	79.48
Average	**76.83**	**77.49**	**78.43**	**77.51**	**78.60**	**79.39**
KC	0.657	0.689	0.706	0.691	0.708	0.713

similar in the case of *model 4*, *model 5*, and *model 6*, in which *model 6* is better than *model 4* and *model 5*. The second comparison is between *model 1* and *model 4* which shows the superiority of PGNN over EGNN in classifying ungranulated data. The superiority of PGNN over EGNN is because of its learning process which is non linear and data dependent. Last comparison is between *model 6* (CBPGNN) and *model 3*, which shows the superiority of proposed model over EGNN in processing CB granulated input patterns.

V. Conclusion

A class based progressive granular neural network is suggested in the present study. It considers the advantage of CB granulation, and non linear method of updating weights in PGNN. CBPGNN has been proved 1.08% better than EGNN in classifying CB granulated IRS LISS III patterns, and also 2.61% better than ungranulated EGNN. The performance of proposed model is proved to be similar for waveform dataset.

VI. Acknowledgement

The authors would like to thank the Department of Science and Technology, Government of India, under which a project titled ”Land Use/Cover Classification of Remote Sensing Images Using Granular Computing methodologies” is being carried out at the Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore.

References

[1] W. Pedrycz, A. Skowron, and V. Kreinovich, *Handbook of Granular Computing*. John Wiley and Sons, England, 2008.

[2] W. Pedrycz and G. Vukovich, “Granular neural networks,” *NEUROCOMPUTING*, vol. 36, pp. 205–224, 2001.

[3] W. Pedrycz, *Granular Computing: Analysis and Design of Intelligent Systems*, vol. 1. CRC press, Taylor and Francis group, 2013.

[4] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, New Jersey, USA, second ed., 1998.

[5] S. K. Meher and M. N. Murty, “Efficient method of moving shadow detection and vehicle classification,” *Int. J. Electron. Commun. (AE)*, vol. 67, pp. 665–670, 2013.

[6] S. K. Pal, S. K. Meher, and S. Dutta, “Class-dependent rough-fuzzy granular space, dispersion index and classification,” *PATTERN RECOGN*, vol. 45, pp. 2690–2707, 2012.

[7] S. K. Pal and S. Mitra, “Multilayer perceptron, fuzzy sets, and classification,” *IEEE Trans. Neural Networks*, vol. 3, pp. 683–697, 1992.

[8] L. A. Zadeh, “Fuzzy sets,” *INFORM CONTROL*, vol. 8, pp. 338–353, 1965.

[9] L. A. Zadeh, *Fuzzy sets and information granularity*. in Advances in Fuzzy Set Theory and Applications, M. Gupta, R. Ragade and R. Yager (Eds.), North-Holland Publishing Co., Amsterdam, pp. 3-18, 1979.

[10] M. Banerjee, S. Mitra, and S. Pal, “Rough fuzzy MLP: Knowledge encoding and classification,” *IEEE Trans. Neural Networks*, vol. 9, pp. 1203–1216, 1998.

[11] D. Stathakis and A. Vasilakos, “Satellite image classification using granular neural networks,” *INT J REMOTE SENS*, vol. 27, pp. 3991–4003, 2006.

[12] D. Stathakis and I. Kanellopoulos, “Global elevation ancillary data for land use classification using granular neural networks,” *PHOTOGRAMM ENG REM S*, vol. 74, pp. 55–63, 2008.

[13] A. Ganivada, S. Dutta, and S. K. Pal, “Fuzzy rough granular neural networks, fuzzy granules, and classification,” *THEOR COMPUT SCI*, vol. 412, pp. 5834–5853, 2011.

[14] S. K. Meher, “Explicit rough-fuzzy pattern classification model,” *PATTERN RECOGN LETT*, vol. 36, pp. 54–61, 2014.

[15] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley Interscience Publications, USA, second ed., 2000.

[16] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. Rosen, “Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps,” *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 698–713, 1992.

[17] N. Kasabov, *Evolving Connectionist Systems: The Knowledge Engineering Approach*. Springer, 2007.

[18] N. Kasabov, “Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning,” *IEEE Trans. Syst., Man, Cybern.*, vol. 31-6, pp. 902–918, 2001.

[19] D. F. Leite, P. J. Costa, and F. Gomide, “Evolving granular classification neural networks,” in *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 1736–1743, 2009.

[20] D. A. Kumar and S. K. Meher, “Multiple classifiers systems with granular neural networks,” in *Proc. IEEE Int. Conf. Signal Processing, Computing and Control*, 2013.

[21] S. K. Meher and D. A. Kumar, “Ensemble of adaptive rule-based granular neural network classifiers for multispectral remote sensing images,” *IEEE J. Sel. Topics Appl. Earth Observ.*, vol. 8, no. 5, pp. 2222–2231, 2015.

[22] D. A. Kumar and S. K. Meher, “Granular neural networks models with class-belonging granulation,” in *Proc. IEEE Int. Conf. on Contemporary Computing and Informatics*, 2014.

[23] ISRO, “National remote sensing center open earth observation data archive.” http://bhuvan.nrsc.gov.in, August 2011.

[24] M. learning repository, “Waveform dataset.” https://archive.ics.uci.edu/ml/datasets.