Soft Computing (2019) 23:4051–4064
https://doi.org/10.1007/s00500-018-3052-x

METHODOLOGIES AND APPLICATION

Fusion of progressive granular neural networks for pattern classification

D. Arun Kumar[1] · Saroj K. Meher[2] · K. Padma Kumari[1]

Published online: 1 February 2018

Abstract
Evolving granular neural network (eGNN) is a system that classifies the data by adapting its architecture according to the input information. eGNN posses the ability to learn from the data with a single pass. However, eGNN has the disadvantage of linear updation of weights during training stage that may not handle the real-time uncertain and imprecise data appropriately. To address this issue, the present study aims to use multiple numbers of progressive granular neural networks (PGNNs) in the framework of fusion of classification systems. PGNN is a modified version of eGNN, where the learning process is nonlinear and takes the class-sensitive (CS) granulated data unlike eGNN. Collective opinion from the group of PGNNs increases the classifier performance in comparison with individual PGNNs. The proposed model thus takes the advantages of CS granulation, structural adaptability, nonlinear updation of weights and collective opinions from the group of PGNNs. Performance of model is tested on various datasets, and its superiority to similar other methods has been justified. Various performance indices, such as overall accuracy, dispersion score, kappa coefficient, producer's accuracy, and user's accuracy, have been used for performance analysis.

Keywords Pattern classification · Class-sensitive granulation · Fuzzy sets · Progressive granular · neural networks

Abbreviations

eGNN	Evolving granular neural network
PGNN	Progressive granular neural network
FPGNNs	Fusion of progressive granular neural networks
NN	Neural network
CIS	Class-insensitive
CS	Class-sensitive
IRS LISS III	Indian Remote Sensing Linear Imaging Self-Scanning Sensor
OA	Overall accuracy
UA	User's accuracy
PA	Producer's accuracy
DS	Dispersion score
KC	Kappa coefficient
FMLP	Fuzzy multilayer perceptron
RFMLP	Rough fuzzy multilayer perceptron
FRGNN	Fuzzy rough granular neural networks
EGNNs	Ensemble granular neural networks
UGNNs	Unified granular neural networks

Communicated by V. Loia.

✉ D. Arun Kumar
arunkumar.mtech09@gmail.com

Saroj K. Meher
saroj.meher@isibang.ac.in

K. Padma Kumari
geologymadam@gmail.com

1 School of Spatial information Technology, JNT University, Kakinada, India

2 Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore, India

1 Introduction

Inspired with the granular processing of information by human brain, granulation operation is normally used to handle uncertainty present in the datasets. Grouping objects based on the characteristics like similarity is called granulation (Pedrycz 2013). A granule is defined as a set of concepts, group of objects or class of numbers (Pedrycz and Vukovich 2001). When these granules are processed through neural networks, the resultant network is called granular neural network (GNN).

In the recent past, various studies have been carried out to suggest methodologies for granulating input data. Pal and Mitra (1992) suggested class-insensitive (CIS) method of granulation, where each feature of input patterns is granulated to three fuzzy linguistic granules, such as *low*, *medium* and *high* (Zadeh 1965). CIS method of granulation is used in several applications(e.g. Banerjee et al. 1998; Stathakis and Vasilakos 2006; Stathakis and Kanellopoulos 2008; Meher 2014). An advanced method of granulating input pattern was suggested by Pal et al. (2012), which is named in the present study as class-sensitive (CS) granulation. Unlike CIS granulation, CS method explores the belonging of each feature to all classes present in the dataset and is more efficient in handling the uncertainty factors that help to enhance the classification accuracy of a model.

Advancement of technology to acquire information in various fields has produced large number of datasets. Classification of these datasets provides improved interpretation of information that can be useful for various applications. Evolving classification models are more suitable than the generic/static one for their ability to classify these time-varying or dynamic datasets. Evolving NNs that belong to this category have the ability to learn quickly from a large size of dataset through the process of creating and updating new modules. There exist various types of learning methodologies, which were used to train the evolving NNs.

An evolving NN called min–max NN was described by Simpson (1992) for the classification of data. In min–max model, a neural network uses fuzzy sets as pattern classes, where each fuzzy set is an union of fuzzy hyperboxes. Each fuzzy hyperbox is an *n*-dimensional box, which is defined by a minimum and maximum point with a respective membership function. This methodology of fuzzy min–max hyperboxes was extended for clustering task (Simpson 1993; Quteishat and Lim 2008). Gabrys and Bargiela (2000) suggested a general fuzzy min–max (GFMM) NN, which is a generalization and extension of the fuzzy min–max clustering and classification algorithms developed by Simpson (1992). Nandedkar and Biswas (2009) suggested an improved evolving NN model, which is called granular reflex fuzzy min–max NN (GrRFMN). It can learn and classify the data by using hyperbox fuzzy set that represents granular information. Zhou et al. (2017) proposed a global context verification scheme to filter false matches for image copy detection. The model is based on obtaining initial scale invariant feature transform (SIFT) matches between images based on the bag-of-visual-words (BOW) quantization. Also, the model is based on region-based global context descriptor (OR-GCD), which is used for the verification of matches to filter false matches. The advantage of OR-GCD is its ability to encode relatively rich global context information of SIFT features and also good robustness and efficiency, which results in effective and efficient verification.

Gu and Sheng (2017) presented a new equivalent dual formulation for υ-support vector classification (υ-SVC) and, then, proposed a robust υ-SvcPath, based on lower upper decomposition with partial pivoting. The extension of υ-SVC algorithm is given in the form of incremental support vector ordinal regression (SVOR) algorithm (Gu et al. 2015). SVOR algorithm can handle a quadratic formulation with multiple constraints, where each constraint is constituted of an equality and an inequality. Gu et al. (2017) proposed structural minimax probability machine (SMPM) for pattern classification. Furthermore, a linear model of SMPM is extended to a nonlinear model by exploiting kernelization techniques. Deng et al. (2017b) proposed a model with multi-population collaborative strategy and adaptive control parameters introduced into the genetic algorithm (GA) and ant colony optimization (ACO) algorithm to produce a genetic and ant colony adaptive collaborative optimization (MGACACO) algorithm for solving complex optimization problems. Wang et al. (2017) designed a back propagation (BP) neural network model using solar radiation (SR) as the input parameter to establish the relationship between solar radiation and air temperature error (ATE) with all the data in the month of May. Later, a trained back propagation (BP) model was used to correct the errors in other months. From the experimental results, it is proved that BP neural network is very suitable for solving this problem due to its powerful functions of nonlinear fitting. Zhang et al. (2017) proposed an optimal cluster-based mechanisms by a modified multi-hop layered model for load balancing with multiple mobile sinks for energy conservation or energy balance in the process of data gathering. Furthermore, to reduce the consumption of network energy, quantization technologies were exploited. Deng et al. (2017a) proposed an efficient multi-objective optimization model for gate assignment in hub airport called as adaptive particle swarm optimization (DOADAPO) algorithm. DOADAPO algorithm is based on making full use of the advantages of alpha-stable distribution and dynamic fractional calculus. Tian and Chen (2017) developed correlation component manifold space learning (CCMSL) to estimate the age from heterogeneous datasets. The superiority of CCMSL model over similar models was experimentally demonstrated with the various type of datasets.

Xue et al. (2017) proposed a self-adaptive ant bee colony (ABC) algorithm based on the global best candidate (SABC-GB) for global optimization. Experimentally, it was demonstrated that SABC-GB is superior to the other algorithms for solving complex optimization problems. It is proved that performance of ABC can be improved by introducing self-adaptive mechanism. Ma et al. (2016) proposed community detection algorithm called as loop edges delete (LED). LED algorithm is based on structural clustering, which converts structural similarity between vertices to weights of network. The results showed that LED is superior to other methods in accuracy and efficiency. Liu et al. (2016) developed an adaptive method aiming at spatial–temporal efficiency in a heterogeneous cloud environment. The model is a optimized kernel-based extreme learning machine, which is proposed for faster forecast of job execution duration and space occupation, which consequently facilitates the process of task scheduling through a multi-objective algorithm called time and space optimized. Kong et al. (2017) proposed a decentralized belief propagation-based method (PD-LBP), for multi-agent task allocation in open and dynamic grid and cloud environments where both the sets of agents and tasks constantly change. Ma et al. (2015) proposed a kk-degree anonymity with vertex and edge modification algorithm, which includes two phase: first, finding the optimal target degree of each vertex; second, deciding the candidates to increase the vertex degree and adding the edges between vertices to satisfy the requirement. The proposed approach proved best when tested on real-world datasets.

Leite et al. (2009) suggested an advanced system called as evolving granular neural network (eGNN) to classify data. eGNNs were constructed from the data using fast incremental learning algorithm. It found information through the development of granules and considered triangular-norm (T) triangular-conorm (S) neurons as basic processing elements. The structure of eGNN evolves according to the size and the number of granules (Leite et al. 2009). However, the disadvantage of eGNN is in its linear updation of weight parameters that may not capture the nonlinearity present in the decision-making process. To mitigate this, a nonlinear learning algorithm was used for the eGNN in Kumar et al. (2016a), which was named as progressive GNN (PGNN). However, the granules that are created or updated during the evolving process play a key role in the network performance and many such granules are obtained for complex datasets. As a result, it becomes a difficult task for the developer to prioritize them and use further. This factor motivated us to suggest a fusion of PGNNs for the classification of complex datasets. The fusion network is a collection of PGNNs, where each PGNN is built using one granule from each class. The number of PGNNs used in the fusion process increases with the number of granules those are created in the evolving process. The final decision of the fusion of PGNN (FPGNN) depends on the strategic combination of individual PGNNs. The fusion of the proposed model is similar to the process of taking decision by considering individual opinion of a group of experts (Kuncheva 2004). We have used different types of decision combination strategies, such as majority voting, aggregation operators like mean, median, product, sum, minimum and maximum to get a combined decision. With this approach, the model can handle uncertainty in the complex dataset more effectively. The nonlinear learning approach of the PGNN further improves the overall performance.

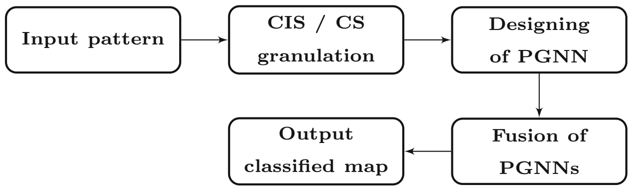

Fig. 1 Schematic flow diagram of the fusion of progressive granular neural networks (FPGNNs) with class-insensitive (CIS) and class-sensitive (CS) granulated input pattern

1.1 Motivation for the fusion of classification models using PGNNs

Three major factors motivated us to propose the fusion of PGNNs-based classification model; (1) eGNNs used a granulation method that was not considering the class information of features, and thus we suggested to use the class-sensitive (CS) granulation, which explores class-belonging information of each feature to different classes present in the dataset. (2) eGNNs learned from the data with a linear algorithm that adapted its structure according to the granules those were created in the evolving process. With linear learning process, eGNNs may not capture nonlinear behaviour of the dataset, which leads to the deterioration of overall classification performance. For this matter, we suggest PGNN with a nonlinear algorithm. (3) With the increase in number of patterns of a dataset, more number of granules get generated. These granules are then used to build the architecture of PGNNs. Each of the granules created during the evolving process possesses information at different levels, and hence, it is difficult to prioritize them. To overcome this, we considered all the granules to build PGNNs. In order to improve the performance of PGNNs while giving importance to each of the granules, we suggested a model with the fusion of PGNNs.

2 Proposed classification model

The proposed model works in the framework of fusion of several PGNNs for the classification task. Functional block diagram of the proposed model is given in Fig. 1. In the first stage of the proposed model, the input patterns of the dataset are granulated using CS method. A set of PGNNs are then built by the granules which are generated from CS granulated data. The collective opinion from the set of PGNNs is used to take the final decision in classifying the given input patterns. The steps involved in the functional block diagram (Fig. 1) are discussed in the following sections.

2.1 Information granulation

Information granulation is one of the solutions to deal with uncertainty in the data. Broadly, one can consider two important methods of data granulation, such as CIS and CS. Brief description of these two methods is provided in the following.

2.1.1 Class-insensitive (CIS) granulation

In CIS granulation, each feature of input patterns is represented by its membership to three fuzzy granules. These granules are termed as *low*, *medium*, and *high* (Pal and Mitra 1992; Banerjee et al. 1998; Kumar and Meher 2013). A π-membership function is used to compute the belongingness of each feature to these three granules (Kumar and Meher 2014). Let a pattern P_1 with n number of features be represented as $P_1 = [f_1, f_2, \ldots, f_n]$. The membership of feature f_i to a granule is given as:

$$\mu(f_i; c, \gamma) = \begin{cases} 2\left(1 - \frac{\|f_i - c\|}{\gamma}\right)^2, & \text{for } \frac{\gamma}{2} \le \|f_i - c\| \le \gamma \\ 1 - 2\left(\frac{\|f_i - c\|}{\gamma}\right)^2, & \text{for } 0 \le \|f_i - c\| \le \frac{\gamma}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

γ and c are the radius and the central point of granule, respectively. The membership of a feature f_i is maximum at the centre of granule with a value 1. This membership gradually decreases on either side of granule and becomes *0.5* at the crossover points. The centre and the radius of three granules along the feature axis f_i are computed by using maximum $(f_{i_{\max}})$ and minimum $(f_{i_{\min}})$ values, which are computed as:

$$\gamma_{\text{medium}}(f_i) = \frac{1}{2}\left(f_{i_{\max}} - f_{i_{\min}}\right), \tag{2}$$

$$c_{\text{medium}}(f_i) = f_{i_{\min}} + \gamma_{\text{medium}}(f_i), \tag{3}$$

$$\gamma_{\text{low}}(f_i) = \frac{1}{\zeta}\left(c_{\text{medium}}(f_i) - f_{i_{\min}}\right), \tag{4}$$

$$c_{\text{low}}(f_i) = c_{\text{medium}}(f_i) - \frac{1}{2}\gamma_{\text{low}}(f_i), \tag{5}$$

$$\gamma_{\text{high}}(f_i) = \frac{1}{\zeta}\left(f_{i_{\max}} - c_{\text{medium}}(f_i)\right), \tag{6}$$

$$c_{\text{high}}(f_i) = c_{\text{Medium}}(f_i) + \frac{1}{2}\gamma_{\text{high}}(f_i). \tag{7}$$

ζ is a parameter that controls the extent of overlapping among these granules. Each feature of a pattern expresses its membership to three fuzzy granules, for example, a pattern P_1 with CIS granulation is represented in the form of a multidimensional vector as

$$P_1 = \begin{bmatrix} \mu_{\text{low}}(f_1), & \mu_{\text{medium}}(f_1), & \mu_{\text{high}}(f_1), \\ \cdots, & \cdots, & \cdots, \\ \mu_{low}(f_n), & \mu_{\text{medium}}(f_n), & \mu_{\text{high}}(f_n) \end{bmatrix}. \tag{8}$$

The values such as $\mu_{\text{low}}(f_n)$, $\mu_{\text{medium}}(f_n)$ and $\mu_{\text{high}}(f_n)$ are the memberships of feature f_n to the three fuzzy granules *low*, *medium* and *high*, respectively.

2.1.2 Class-sensitive granulation

CIS granulation works with three fuzzy granules that do not exploit class belongingness of each feature. To overcome this limitation, Pal et al. (2012) suggested CS method of granulating input feature space. In CS granulation, each feature of a pattern is represented in terms of its membership with respect to the classes present in the dataset and the membership of a feature f_i to a specific class is given by:

$$\mu(f_i; a, r, b) = \begin{cases} 0, & \text{for } f_i \le a \\ 2^{N-1}\left(\frac{f_i - a}{r - a}\right)^N, & \text{for } a < F_i \le p \\ 1 - 2^{N-1}\left(\frac{r - f_i}{r - a}\right)^N, & \text{for } p < F_i \le r \\ 1 - 2^{N-1}\left(\frac{f_i - r}{b - r}\right)^N, & \text{for } r < F_i \le q \\ 2^{N-1}\left(\frac{b - f_i}{b - r}\right)^N, & \text{for } q < F_i \le b \\ 0, & \text{for } f_i \ge b. \end{cases} \tag{9}$$

The variables p and q are the two crossover points at which the membership is equal to 0.5, and r is the centre of granule with membership value *1*. The values of p, q and r are computed as $r =$ *mean of feature values that belong to a particular class*, $p = r - (\frac{\max(F_n) - \min(F_n)}{2})$ and $q = r + (\frac{\max(F_n) - \min(F_n)}{2})$. N is the parameter whose value is between 0 and 1. The variables $\min(F_n)$ and $\max(F_n)$ are minimum and maximum values of the class for a feature F_n. The points a and b are called extreme points, which are given as $a = r - (q - p)$ and $b = r + (q - p)$. Thus, a pattern P_1 with n number of features and c number of classes can be represented by $n \times c$ number of granulated features as:

$$P_1 = \begin{bmatrix} \mu_1(f_1), & \mu_2(f_1), & \mu_c(f_1), \\ \mu_1(f_2), & \mu_2(f_2) & \ldots, \\ \mu_1(f_n), & \mu_2(f_n), & \mu_c(f_n) \end{bmatrix}, \tag{10}$$

where $\mu_c(f_n)$ is the membership of feature f_n to the class c.

2.2 Development of PGNN

PGNN processes the linguistic granules and makes the network more interpretable. PGNNs are known for their ability to process granules for performance improvement. The architecture of PGNN adapts itself based on the behaviour of input granules. The structural adaptability of PGNNs is based on the number and the size of granules. The structure of PGNNs

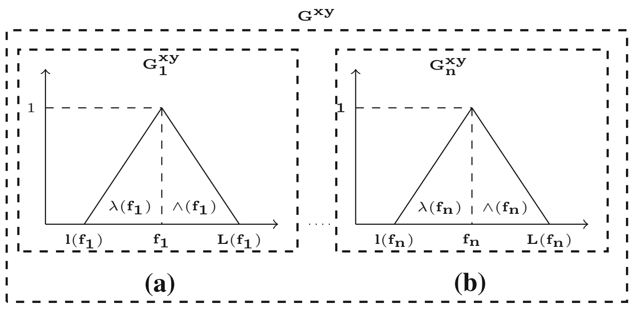

Fig. 2 **a** Shape of the granule created for the features f_1 and f_n

adapts to the input data by (1) creating the new granules and (2) updating the existing granules.

2.2.1 Creating granules

Architecture of PGNNs are built using the granules created from dataset. Creation of new granule is implemented in three different cases: those are (1) if the incoming data sample is the first one, (2) when the input sample belongs to a new class and (3) when the input pattern cannot be fit in the existing granules. A new granule is created for an input sample/pattern $P_i = \left[f_1\ f_2\ \ldots\ .\ f_n\right]$, along each feature axis. Boundary of new granule for the feature value f_1 is defined by the parameters $l(f_1)$ and $L(f_1)$, with two crossover points $\lambda(f_1)$ and $\wedge(f_1)$. These parameters are given as: $l(f_1) = f_1 - \rho$, $\lambda(f_1) = \wedge(f_1) = f_1$ and $L(f_1) = f_1 + \rho$. ρ is the parameter which is used to control the extent of granularity. Similarly, the boundary of granule for the rest of features is computed along the respective feature axis. The resultant granule is labelled with the corresponding class of input pattern. The process of creating and labelling the granule is shown in Fig. 2. Once a granule is created, it is labelled with the corresponding class and also with its number in that class and is represented as G^{xy} (where G = granule, y = class of granule and x = number of granule that belongs to class y). If an incoming pattern belongs to a granule, its membership to the granule is computed by using trapezoidal membership function, and this membership value is used to train the PGNN.

2.2.2 Updating granules

The size of a granule is updated if the incoming pattern does not fit within the granule. This mechanism of updating a granule is performed to accommodate the incoming pattern. If the pattern fits itself in the granule, the boundaries of that granule along each feature axis are not updated. This is implemented by updating the parameters l, λ, $\wedge$ and L. Let G^{xy} be an existing granule that belongs to class y, and a new pattern P_j is to be fit within G^{xy}. The size of granule G^{xy} is updated along

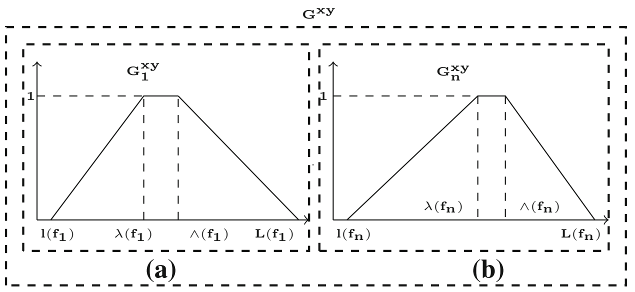

Fig. 3 Possible shapes of the granule with updated parameters **a** Example 1 obtained by feature 1, and **b** Example 2 by feature n

each feature axis to accommodate P_j, which is explained as:

```
**if** $f_i - \rho < f_j < f_i$ **then**
    $l(f_i) = f_i - \rho, \lambda(f_i) = f_j, \wedge(f_i) = f_i, L(f_i) = f_i + \rho$
**else if** $f_i < f_j < f_i + \rho$ **then**
    $l(f_i) = f_i - \rho, \lambda(f_i) = f_i, \wedge(f_i) = f_j, L(f_i) = f_i + \rho$
**end if**.
```

f_i and f_j are the present and future feature values, respectively. Updating the parameters of granule G^{xy} is implemented by considering all features of the incoming pattern (starting from f_1 to f_n). Pictorial representation of updating process of a granule is shown in Fig. 3. The process of creating and updating the granules is implemented by passing each pattern one by one from the dataset. As a result, the feature space with all the patterns is divided into multiple granules of different classes.

2.2.3 Architecture of PGNN

With the created/updated granules, the architecture of PGNN is designed. The number and size of granules increase with the number of input patterns. Basic architecture of PGNN consists of four layers, such as input layer, progressive layer, aggregation layer and output layer. Each layer possesses the processing nodes, which are key elements in PGNN. The number of input layer nodes is determined by the number of features. Product of the number of granules and number of features in the pattern is equal to the number of progressive layer nodes. Each node of input layer is connected to the granules at the progressive layer through which the membership of feature is computed. The output at each node of progressive layer is the membership of respective feature to the granule. The number of aggregation layer nodes is equal to the number of granules. Membership of all the features of a pattern to a specific granule is considered as an input to a node of aggregation layer. Number of classes present in the data determines the size of output layer. The output at aggregation layer nodes is given as an input to the nodes of output layer. As an example, in Fig. 4, outputs of aggregation layer nodes, which represent the granules (belonging

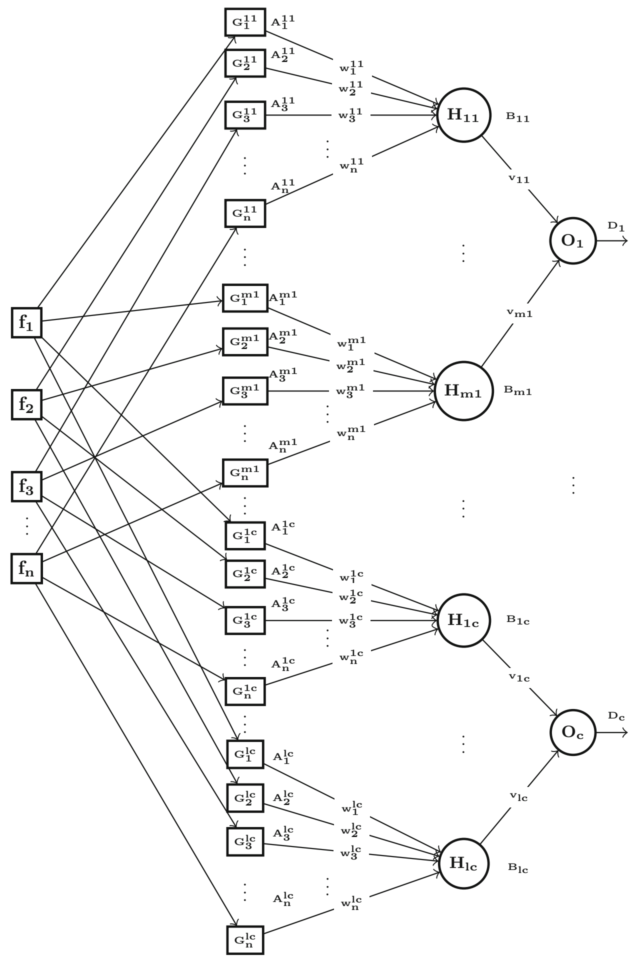

Fig. 4 Architecture of progressive granular neural network consisting of four layers, namely input, progressive, aggregation, and output layers

to class 1) are connected to the first node of output layer. Similarly, those nodes that belong to a specific class are connected to the corresponding nodes of output layer. With the designed architecture of PGNN, the error at the nodes of output layer is calculated. If the output error for an input pattern is zero, then the weights between the layers of PGNN are not updated. Otherwise, the weights are updated with the errors that are computed at each layer. This process is implemented by passing each pattern one by one during the training stage.

2.2.4 Learning process of PGNN

PGNN learns from the labelled patterns by experiencing each one of them only once during the training stage. For each input sample, creation or updation of granule is implemented after which the membership of each feature of a sample to the granules is computed. This membership is the output at corresponding node of progressive layer. The memberships of features to a granule and the initial weights are integrated at the aggregation layer node, and the output at each aggregation node is computed. The outputs of aggregation layer nodes which belong to a specific class are connected to the output layer node of PGNN, and the output at output layer node is computed. If the output error is zero, then the weights between the layers of PGNN are not updated. Otherwise, weights between the layers of PGNN are updated. This process of learning is implemented to decrease the cost function of PGNN. The cost function of PGNN is given as:

$$Er = \frac{1}{2}\sum_{k=1}^{c}(T_k - D_k)^2, \tag{11}$$

where c is the number of classes and T_k is the target value at the output node O_k. Initially, the weights between the layers are assigned with the random values, and these weights are updated according to the output error which is computed for every pattern.

2.3 Fusion of progressive granular neural networks

In the design of a classification model, the basic limitation of a PGNN is clearly seen, i.e. to decide optimum number of granules for each class. The task becomes more cumbersome for the datasets with highly overlapping class boundaries, as normally seen in remote sensing imageries. With the increasing complexity of a dataset, number of granules per class increases. Under such circumstances, there is every possibility that some of the granules may act as redundant that leads to more computation with deteriorated performance. These factors motivated us to modify the architecture of a PGNN to design a fusion structure that can reduce the computational burden and at the same time improve the performance of model. With this assumption, we design number of PGNNs, where each PGNN is having one granule per class. Thus, number of PGNNs is equal to the minimum number of granules that can be created among the classes of a dataset. Once the PGNNs are designed, the outputs (decisions) of these PGNNs are combined using different decision combination methods. The framework is similar to a committee of experts, where the collective opinion is taken as the final decision (Jatindra et al. 2016; El-Melegy and Ahmed 2007; Chee and Robert 2003; Zhi et al. 2002; Fong and Wei 2008; Bae and Jin 1995; Zhang 2000). Conceptually, the proposed fusion of PGNN (FPGNN) is similar to the model [ensemble of GNN (EGNN)] described in Meher and Kumar (2015). In EGNN (Fig. 5), the author has designed each GNN with one rule per class, as compared to one granule per class in FPGNN. The number of rules per class in EGNN determines the number of GNNs, which is predetermined based on some fuzzy rules. However, number of PGNNs in FPGNN is adaptive and hence the method is more flexible to learn the input infor-

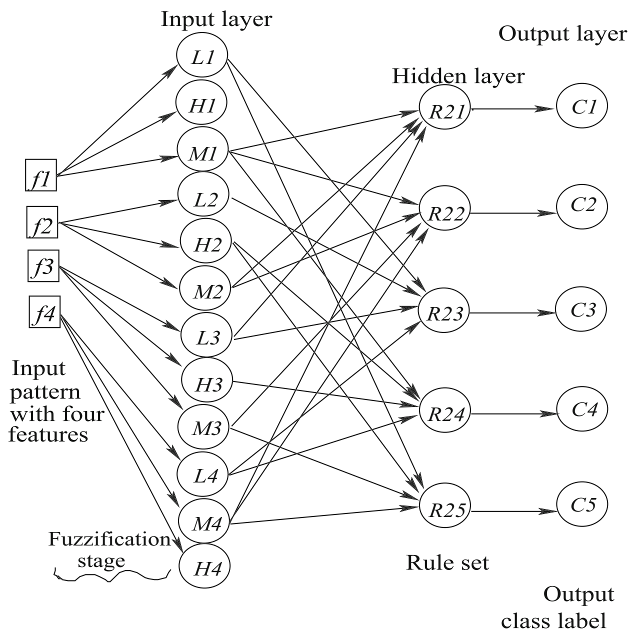

Fig. 5 The architecture of a GNN (used in EGNNs) with 12 inputs, five rules and five classes for the rules extracted with IRS LISS III dataset

mation compared to EGNN. In addition, EGNN requires an iterative learning process, which is time-consuming unlike FPGNN, which learns the information with a single pass of input. In a comparison of FPGNN and PGNN (Fig. 4) with respect to network architecture, PGNN keeps generating the granules and increasing the computation because of its evolving characteristic and ends up with large number of granules, out of which many of them may act as redundant for taking the final decision. This procedure thus leads to the generation of uneven number of granules for the classes, and thus, the number of granules depends on the types of information available for the classes of a dataset. On the other hand, FPGNN generates required number of granules per class for each PGNN and at the end combines the decisions of individual PGNNs.

We have illustrated the block diagram of FPGNN model in Fig. 6. PGNN_1 is built using first granule of each class. The architecture of PGNN_1 which is used to build fusion model is given in Fig. 7. Similarly, PGNN_2 is built using second granule of each class. Accordingly, fusion of PGNNs is built using n number of PGNNs (PGNN_1, $\text{PGNN}_2, \ldots,$ PGNN_n). The outputs of all the PGNNs are combined using the combining criteria, and final class label is assigned to the pattern.

3 Datasets used

Six different datasets have been considered in the present study to evaluate the performance of the proposed model.

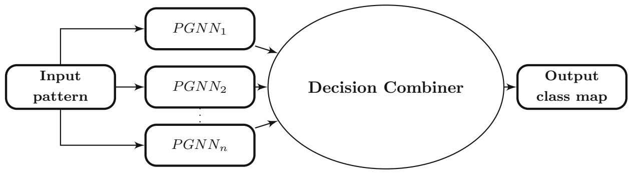

Fig. 6 Schematic flow diagram of the proposed fusion framework using several PGNNs

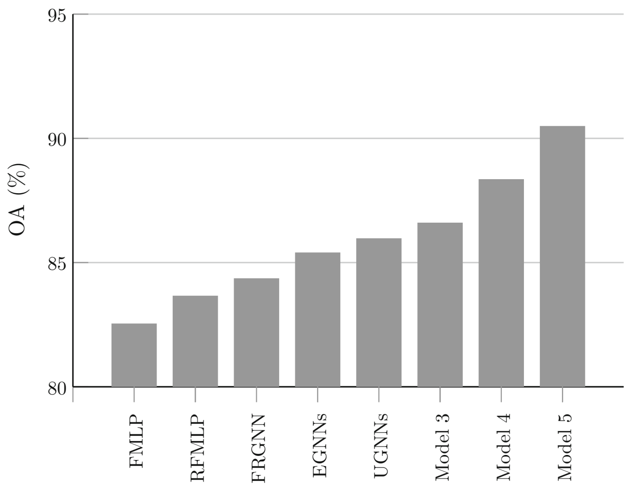

Fig. 7 Architecture of a single PGNN that is used in the fusion framework (Fig. 6) of the proposed model

These datasets are highly overlapping with multiple number of features. Indian Remote Sensing Linear Imaging Self-Scanning Sensor (IRS LISS III) ISRO (2011) image of Mumbai region, India, is considered to prepare the dataset with five classes, namely Urban Dense, Urban Sparse, Water, Forest and Agriculture. A total of 1000 samples have been considered from the image, where 200 samples from each of the five land use/cover classes are selected randomly. Vowel dataset consists of 871 samples with three features and six classes (Lichman 2013). These samples are highly overlapping and possess ill-defined boundaries. Pima Indian dataset consists of 768 samples with eight attributes and two classes. These classes specify whether the patient is tested positive or negative for diabetes (Lichman 2013). Phoneme dataset consists of 5404 patterns with two classes that distinguish between nasal and oral vowels. Phoneme dataset contains vowels coming from 1809 isolated syllables, and the amplitudes of first five harmonics are chosen as features to characterize each vowel (Lichman 2013). Satimage dataset is the collection of samples from Landsat Multi-Spectral Scanner (LMSS) image with 82 × 100 pixels (Lichman 2013). Each pixel value has information from four spectral bands, and the dataset consists of six different land cover classes with 6435 pixels. In the present experiment, four features out of existing 36 features were considered as per the recommendations of dataset designer. Waveform data consist of

5000 patterns with three classes of waves and 21 number of features. Each class of the dataset is generated by the combination of base waves, and the features are corrupted with noise (mean $= 0$ and variance $= 1$) (Lichman 2013).

4 Performance measurement indexes

We have used various measurement indexes to assess the classification performance of the proposed model. All these indices are obtained through the error matrix also known as confusion matrix (CM). CM is a cross-tabulation of the actual classified output classes assigned by a model. Overall accuracy (OA) is the sum of all diagonal elements of CM. OA only takes care of overall agreement between actual and predicted class labels. This misleads the decision-making capability of a model. To overcome this drawback, several other measures, such as producer's accuracy and user's accuracy, are used to get better analysis. Errors of exclusion also known as omission errors are evaluated in the case of user's accuracy (Foody 2002). Omission error is defined as the total number of correct samples in a category divided by the total number of samples from the actual/reference data (Stehman 1996). User's accuracy is the probability of a reference sample being correctly classified. Producer's accuracy is also known as commission error, defined as the ratio of total number of correctly classified samples in a class to the total number of samples that are classified to that class.PA is the most preferred than UA (Richards and Jia 2006). Dispersion score (DS) is the measure (Pal et al. 2012) that quantifies the nature of distribution of classified samples among the classes present in dataset. DS helps in understanding the overlapping characteristics of various class boundaries. Kappa coefficient (KC) is the measure of overall performance of the model based on the individual class-wise agreement (Cohen et al. 1991).

5 Experimental results and analysis

5.1 Model description

Five different models have been considered for the comparative analysis in order to show the advantages of the proposed model. These models are characterized by different granulation methods and neural networks (NNs). All models belong to the group of evolving systems, where the network architecture of these models adapts its structure according to the incoming samples. The models considered for analysis learn from the incoming data with a single pass. This overcomes the problems of iterative process of the conventional NN. Model 1 is an eGNN (Leite et al. 2009), which classifies ungranulated datasets. Models 2 and 3 are eGNNs with CIS and CS granulated input, respectively. Models 4 and 5 are PGNN and fusion of PGNNs with CS granulated input data, respectively. The description of five models is given as:

- *Model 1: Ungranulated input + eGNN, Model 2: Class-insensitive granulation of input + eGNN, Model 3: Class-sensitive granulation of input + eGNN, Model 4: Class-sensitive granulation of input + PGNN and Model 5: Class-sensitive granulation of input + Fusion of PGNNs*

5.2 Strategy of selecting the training data

The models considered in the present study are trained and tested with the patterns that are randomly collected from the datasets. For selection, each dataset is divided into two parts. Part I is used for training and Part II for testing. Parameters of the classification model (like the number of granules, ρ, etc.) are computed using the training data, and performance of model is evaluated with the test data. Four sets of samples have been considered, such that 20, 40, 60 and 80% as training and the rest 80, 60, 40 and 20% as testing. Equal per cent of samples are collected from each of the classes for training. We have performed tenfold cross-validation for each dataset, and average of all the folds (overall accuracy) is provided in the result sections.

5.3 Classification of datasets

5.3.1 IRS LISS III dataset

The training samples from IRS LISS III dataset are selected according to the selection criteria as mentioned in Sect. 5.2. Once the training samples are collected, creation of new granules or updation of existing granules is performed. The granules for different models are generated by considering the parameter ρ as constant along each feature axis. This is computed as: $\rho = \text{mean}(D)$, where D is the set of training samples that belong to a specific class.

During the testing phase, the unlabelled samples are processed one by one and the memberships of each sample to the granules are computed. These membership values are integrated according to the architecture of the model. The output at each node of output layer is computed. Similarly, the outputs at the output layers of all the PGNNs are combined using the combining criteria, and then the class label is assigned to the pattern. For the IRS LISS III dataset, 11 number of PGNNs are obtained for the design of fusion structure. Performance of all the models with various combinations of training samples is tested on IRS LISS III dataset, and results are depicted in Table 1. Performance of models is analysed by three facets of comparison of results obtained from the above-mentioned five models. In the first facet, comparison among Models 1, 2 and 3 has been carried out. The OA obtained with Models 1, 2 and 3 are 86.52, 86.90 and 87.56%, respec-

Table 1 Experimental results of models for *IRS LISS III* dataset

Model	Training						
	20%	40%	60%	80%	OA%	KC	T_c
Model 1	83.14	85.83	87.98	88.53	**86.52**	0.826	0.32
Model 2	83.88	86.13	88.18	89.41	86.90	0.830	0.48
Model 3	84.04	86.98	88.26	90.96	87.56	0.838	0.57
Model 4	84.74	87.58	88.62	92.14	88.27	0.846	0.46
Model 5							
Mean	85.08	87.68	92.48	93.36	89.65	0.860	0.18
Median	85.86	88.08	92.08	93.38	89.85	0.866	
Product	86.98	88.68	92.48	93.50	**90.41**	**0.874**	
Sum	87.04	88.78	89.65	92.48	89.65	0.860	
Maximum	87.46	88.98	89.52	91.88	89.46	0.854	
Minimum	87.88	89.04	90.88	92.00	89.95	0.870	

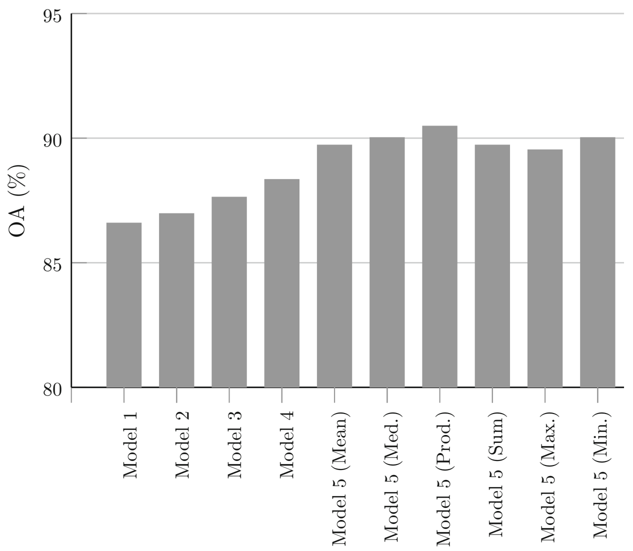

Fig. 8 Overall accuracy of the models for IRS LISS III dataset

tively. This shows the superiority of Model 3 over Models 2 and 1. The OA of Model 3 is 1.04% greater than Model 1 and 0.66% greater than Model 3. Similarly, superiority of Model 3 over Models 2 and 1 is justified with KC. Model 3 has KC value of 0.838, which is better than the KC values of Models 1 and 2.

Superiority of Model 3 over Models 2 and 1 is because of CS granulation of input samples which was reflected in the performance of model. CS granulation of input data handles uncertainty present in the data better than CIS granulation and ungranulated input data (Pal et al. 2012; Kumar et al. 2016b). This is justified with the first facet of comparison. Second facet of comparison is between Models 3 and 4. This comparison is carried out to know the superiority of PGNN over eGNN for CS granulated data. Model 4 performed better than Model 3 with an improved OA of 0.71%. Superiority of Model 4 over Model 3 is supported with KC. Model 4 has a KC of 0.846, which is 1.9% greater than Model 3. Superiority of Model 4 over Model 3 was because of learning process in PGNN, which is data dependent and nonlinear that improved the overall performance of a classifier. The third facet of comparison is between Models 4 and 5. Model 4 is a PGNN, which is built by considering 11 granules from each class that are generated from the training data. Model 5 is a fusion of 11 PGNNs such that each PGNN is built by considering one granule from each class. The collective opinion for Model 5 is obtained by considering six combining criteria, and the corresponding results are given in Table 1.

Model 5 yielded the maximum OA of 90.41% for the combining criteria *product* and minimum OA of 89.46% for the combining criteria *maximum*. The bold texts in all the tables of the result section represent the best result. This shows Model 5 is superior to Model 4 with minimum OA of 89.46%, which is 1.19% better than 88.27%. Also, Model 5 is superior to Model 4 with maximum OA of 90.41%, which is 2.14% better than 88.27%. Superiority of Model 5 over Model 4 is justified with KC values. The third facet of comparison demonstrates the superiority of FPGNNs over individual PGNN for CS granulation of input data. Pictorial representation of the results obtained for the models is provided in Fig. 8.

Performance comparison of the proposed FPGNNs model with other classifiers The performance of the proposed classification model (Model 5) has also been compared with other similar types of methods. The reason in comparing Model 5 with others is to show how evolving systems perform better in classifying the datasets. One of the advantages of evolving systems is their ability to classify the samples with single pass. Traditional classifiers need the data to be passed multiple times during learning. Number of times the training samples passed through a classifier is called here as number of iterations (NOI). We have considered five existing models to compare the performance of the proposed model. These models are (1) fuzzy multilayer perceptron (FMLP): it is a multilayer perceptron NN (Pal and Mitra 1992), which is trained and used to classify CIS granulated data. (2) Rough fuzzy multilayer perceptron (RFMLP): RFMLP is a multilayer perceptron NN (Banerjee et al. 1998), which is encoded with a knowledge extracted using rough set-based theoretic concepts. Input to RFMLP is granulated using CIS granulation method. (3) Fuzzy rough granular neural network (FRGNN): FRGNN is a GNN (Ganivada et al. 2011) with CIS granulated input data. The network is equipped with the knowledge extracted using fuzzy rough set-based concepts. (4) Ensemble of GNNs (EGNNs): EGNNs (Meher and Kumar 2015) is a collection of GNNs in which each GNN is built according to the rules extracted using adap-

Table 2 Comparison with other models using IRS LISS III dataset

Model	OA (%)	NOI	T_c (s)
FMLP	82.46	800	13.48
RFMLP	83.58	800	12.46
FRGNN	84.28	800	10.89
EGNNs	85.32	800	6.24
UGNNs	85.89	800	5.68
Model 3	86.52	Single pass	0.26
Model 4	88.27	Single pass	0.22
Model 5 (Product)	90.41	Single pass	0.18

tive rule extraction method (ARE). (5) Unification of GNNs (UGNNs): UGNNs model (Kumar et al. 2016b) is an association of GNNs which are built with the set of rules extracted using Kasabov rule extraction method (KRE). The architecture of all the models which are considered for the comparison is different. We have considered 800 iterations to train all the five models used for comparison.

Models 3, 4 and 5 are the evolving systems that are used to classify the data using CS granulated input, and the performance results are provided in Table 2. It is seen from the Table 2 that the proposed Model 5 is superior to remaining models with the highest OA of 90.41%, which justifies the importance of considering the collective opinion from the group of PGNNs. The major advantage of FPGNNs is the ability to perform with the single pass of data points. The second highest performance has been obtained for Model 4 with an OA of 88.27%, followed by Model 3 with an OA of 86.52%. This shows the advantage of evolving systems over existing conventional models. EGNNs (OA 85.32%) and UGNNs (OA 85.89%) are similar to Model 5 but with the difference in the architecture of GNNs used to build them. The architecture of GNN used in the EGNNs model is given in Fig. 5. The GNN shown in Fig. 5 is built using single rule per class. The GNN model consists of three layers, namely input layer, hidden layer, and output layer. The architecture of GNN is built based on the *if then* rules that are extracted using adaptive rule extraction (ADE) method. The architectural difference between GNN used in EGNNs and the PGNN in FPGNNs produces differences in the performance. Due to this, FGNNs have produced OA of 90.41% and EGNNs have produced (OA 85.32%). Also, EGNNs are trained for 800 iterations which took T_c equal to 6.24 s. FPGNNs with single pass learning have taken 0.18 s during the training. The first three models (1) FMLP has OA of 82.46%, (2) RFMLP has OA 83.58% and (3) FRGNN has OA 84.28%, out of which FRGNN has achieved better performance because of the initial weights that are computed using fuzzy rough set theory. This results in less training time T_c of 10.89 (s) for FRGNN compared with FMLP and RFMLP. The mean

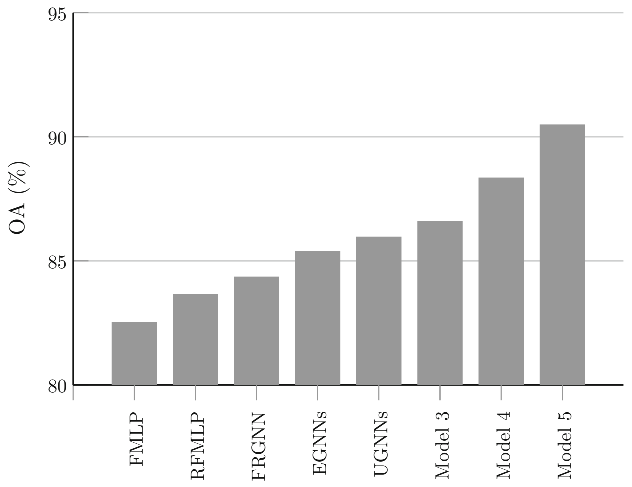

Fig. 9 Comparison of the proposed model with similar other types of method for IRS LISS III dataset

square error of these three models decreases over the number of iterations, indicating the learning ability of these models. The major advantage of Model 5 is its learning ability from dataset with a single pass and less training time. In the present study, Model 5 has taken 0.18 s to learn from the IRS LISS III dataset, which is superior than conventional models. The OA of comparative models is pictorially represented in Fig. 9.

Performances of all the models for IRS LISS III dataset were compared by the three metrics, namely user's accuracy, producer's accuracy and dispersion score. We have considered Models 3, 4 and 5 for comparison. The reason behind this comparison is to show how Model 5 is superior to other models, i.e. eGNN and PGNN by considering CS granulated input data. The performance of Models 3, 4 and 5 with UA, PA, DS and KC is provided in Table 3. For the IRS LISS III dataset, the improvement in UA from Models 1 to 5 has been observed for each class. For example, UA of class 1 in the case of Model 3 is 76.00%, which has increased to 87.53% for Model 5. This is similar for all the classes in each model. A similar trend has been observed in the case of PA, where Model 3 has PA of 56.54% for class 1 and it has been improved to 62.72% in the case of Model 5. Improvement in the performance trend of UA, PA and DS from Models 3 to 5 is provided in Fig. 10. DS gives the impression of overlapping characteristics of various class boundaries. Lesser the DS of a model, better the performance of a classifier. A DS of 0.452 has been obtained by Model 1 in case of class 1 that has been reduced to 0.190 with Model 5. This shows the better classifying ability of Model 5 over Models 4 and 3.

5.3.2 Vowel dataset

Results of five models for various combinations of training samples of vowel dataset are provided in Table 4. It is seen

Table 3 Five classes of IRS LISS III dataset with its corresponding user's accuracy, producer's accuracy, dispersion score and kappa coefficient for Models 3, 4 and 5

Model	Class	UA	PA	DS
Model 3	1	**76.00**	**56.54**	**0.452**
	2	87.11	100	0.315
	3	91.44	92.95	0.196
	4	91.27	87.14	0.240
	5	86.16	86.42	0.339
Model 4	1	**84.43**	**62.52**	**0.293**
	2	91.78	100	0.263
	3	91.64	94.26	0.193
	4	92.88	89.67	0.173
	5	87.83	87.82	0.308
Model 5	1	**87.53**	**62.72**	**0.253**
	2	95.58	100	0.218
	3	92.74	94.30	0.190
	4	92.88	85.36	0.177
	5	87.58	91.20	0.342

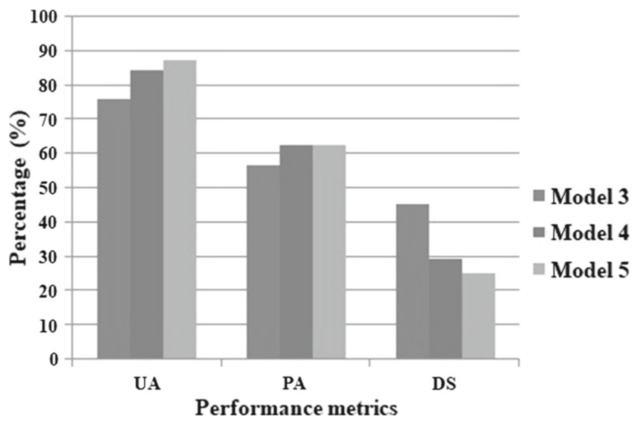

Fig. 10 UA, PA and DS for class 1 of IRS LISS III dataset using Models 3, 4 and 5

from Table 4 that Model 5 (FPGNNs) provided a maximum OA of 88.89% for the combining criteria *product* and Model 1 yielded a minimum OA of 86.44%. Superiority of FPGNNs is also supported by the KC value of 0.877, whereas Model 1 has a minimum KC value of 0.810. Pictorially, variations in the OA values with different models is provided in Fig. 11.

5.3.3 Waveform dataset

The trend of performance of all the models for waveform dataset is similar to the result obtained with IRS LISS III and vowel datasets. Models considered in the present study have also been tested with waveform dataset, and the results are given in Table 5. The results are pictorially represented in Fig. 12. Fusion of PGNNs produced the highest OA (88.68%)

Table 4 Experimental results of all the models for *Vowel* data

Model	Training					
	20%	40%	60%	80%	OA%	KC
Model 1	84.78	85.98	86.84	88.16	**86.44**	0.810
Model 2	85.78	86.58	87.74	88.14	**87.06**	0.828
Model 3	85.88	86.93	87.64	89.71	**87.54**	0.842
Model 4	86.98	87.86	88.64	89.16	**88.16**	0.864
Model 5						
Product	86.99	87.67	89.04	91.86	**88.89**	0.877
Maximum	87.09	88.18	89.24	89.04	**88.48**	0.874

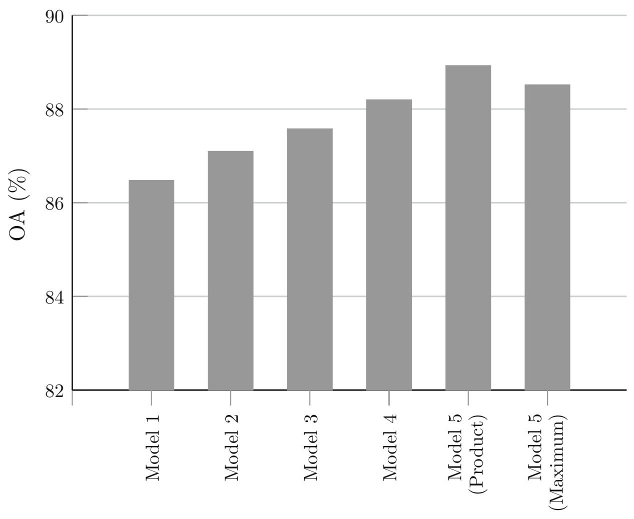

Fig. 11 Overall accuracy of the models for vowel dataset

Table 5 Experimental results of all the models for *Waveform* dataset

Model	OA%	KC
Model 1	**85.48**	**0.761**
Model 2	86.02	0.772
Model 3	86.78	0.794
Model 4	87.56	0.814
Model 5 (mean)	**88.68**	**0.862**

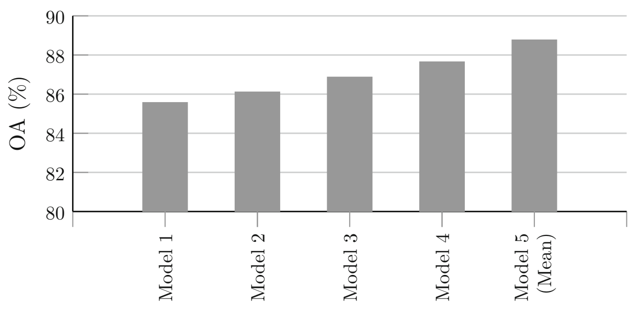

Fig. 12 Overall accuracy of the models for Waveform dataset

Table 6 Experimental results of all the models for *Satimage* dataset

Model	OA%	KC
Model 1	**88.02**	**0.846**
Model 2	88.78	0.852
Model 3	89.23	0.858
Model 4	90.02	0.871
Model 5 (sum)	**91.12**	**0.880**

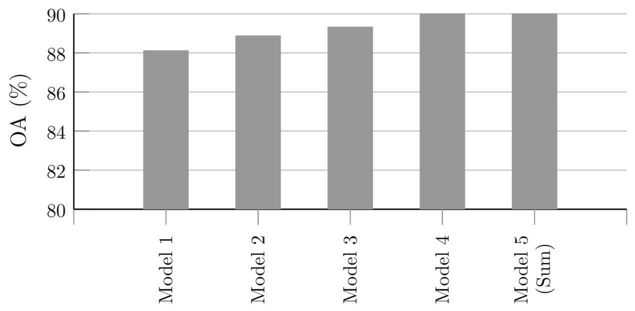

Fig. 13 Overall accuracy of the models for Satimage dataset

Table 7 Experimental results of models for *Pima Indian* dataset

Model	20%	40%	60%	80%	OA	KC
Model 1	79.42	80.45	81.22	81.79	80.72	0.72
Model 5 (Mean)	82.89	83.88	84.86	87.21	84.71	0.84

for the combining criteria *mean* with KC value 0.862. Model 1 yielded the lowest OA of 85.48% with the KC value 0.761.

5.3.4 Satimage dataset

All five models are tested with Satimage dataset, and the results are provided in Table 6 and in Fig. 13. A similar trend in the performance of models has been observed with Satimage dataset also as seen in case of IRS LISS III dataset. Model 5 has an highest OA of 91.12% with KC value 0.880 and Model 1 with lowest OA of 88.02% and KC 0.846.

5.3.5 Pima Indian dataset

Pima Indian dataset possesses 768 samples that belong to two classes. All the five models have been tested on the Pima Indian dataset, and the results are given in Table 7. As the trend of performance for this dataset is similar to LISS III dataset, We have given only the results of Models 1 and 5 in Table 7. FPGNNs (i.e. Model 5) has an highest OA (with combining criteria *mean*) of 84.71%, which is 4.01% greater than eGNN which has the least OA of 80.72%. This is also supported with KC as Model 5 has KC of 0.84 with 16.6% gain over Model 1 which is having KC of 0.72. The OA accuracy of each model is pictorially represented in Fig. 14.

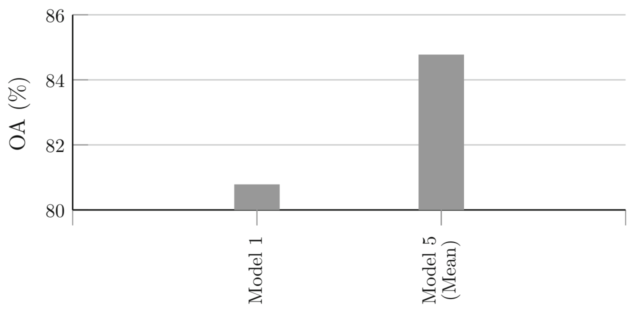

Fig. 14 Overall accuracy of the models for Pima Indian dataset

Table 8 Experimental results of models for *Phoneme* dataset

Model	10%	30%	50%	70%	OA	KC
Model 1	82.66	83.48	84.02	84.84	83.75	0.624
Model 5 (Product)	85.98	86.88	87.46	88.78	87.41	0.716

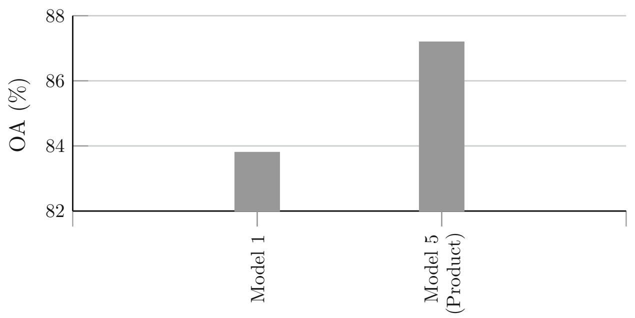

Fig. 15 Overall accuracy of models for Phoneme dataset

5.3.6 Phoneme dataset

Phoneme dataset consists of 5404 patterns with two classes that distinguish between nasal and oral vowels. Seven models have been tested on the phoneme dataset, and the results were obtained similar to other datasets. The performances of Models 1 and 5 for different combinations of training data are provided in Table 8. We have considered the results of these models for 10, 30, 50 and 70% of training samples. Model 5 has an highest OA (with combining criteria *Product*) of 87.41% and Model 1 with lowest OA of 83.75%. The superiority of Model 5 over other models is also justified with highest KC of 0.716. Pictorial representation of the OA of the models is given in Fig. 15.

6 Conclusions

An efficient classification model with the fusion of progressive granular neural networks was proposed in the present article. The model explored class-sensitive granulation of patterns which could handle uncertainty in the data efficiently. In the comparative analysis, eGNN with CS granulation has produced 1.02% better OA compared with eGNN without granulation as tested with IRS LISS III dataset. The nonlinear and data-dependent learning ability of PGNN has produced 1.75% better OA compared with eGNN without granulation. In fusion of PGNNs, the collective opinion from PGNNs produced better classification results. These characteristics of FPGNNs make it to be the best among similar models. These advantages of FPGNNs overcome the limitations of eGNN and resulted in better classification performance. Performance of FPGNNs model was proved to be better than the conventional models with an advantage of single-pass learning. In the present article, FPGNNs produced 3.89% better OA and 0.10 s better T_c compared with eGNN. The proposed model has fulfilled the three motivated factors in the present study, namely CS granulation of data, nonlinear and data-dependent learning of PGNN and collective opinion from the group of PGNNs. Superiority of FPGNNs model over similar types of methods has been observed with various datasets. FPGNNs have produced 7.95% better OA compared with fuzzy multilayer perceptron. In comparison with rough fuzzy multilayer perceptron, the proposed model has produced 6.83% better OA. FPGNN has produced 6.13% OA compared with FRGNN. The superiority of FPGNN model over ensemble of GNNs and unified GNN is justified with an improved OA of 5.09 and 4.52%, respectively. FPGNN model has an highest UA of 87.53% and highest PA of 62.72% for class 1, and the performance is similar in the case of other classes. FPGNN possessed the best DS of 0.253 for class 1 as compared with other models. The superiority of the proposed model is justified with the performance indices, such as overall accuracy, dispersion score, kappa coefficient, producer's accuracy and user's accuracy. Superiority of the proposed model over similar models is proved with remaining five datasets. Furthermore, this work can be extended with knowledge-based PGNNs, where PGNN is encoded with the knowledge obtained from the data by using fuzzy rough set theoretic concepts.

Acknowledgements The authors would like to thank the Department of Science and Technology, Government of India, under which a project titled Land Use/Land 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.

Compliance with ethical standards

Conflict of interest Authors declare that they do not have potential conflict of interest.

Humans and animals rights This article does not contain any studies with humans and animals performed by any of the authors.

References

Bae CS, Jin HK (1995) Combining multiple neural networks by fuzzy integral for robust classification. IEEE Trans Syst Man Cybern 25(2):380–384

Banerjee M, Mitra S, Pal S (1998) Rough fuzzy MLP: knowledge encoding and classification. IEEE Trans Neural Netw 9:1203–1216

Chee P, Robert F (2003) Online pattern classification with multiple neural network systems: an experimental study. IEEE Trans Syst Man Cybern Part C (Appl Rev) 33(2):235–247

Cohen FS, Fan Z, Patel MA (1991) Classification of rotation and scaled textured images using gaussian markov random field models. IEEE Trans Pattern Anal Mach Intell 13:192–202

Deng W, Zhao H, Yang X, Xiong J, Sun M, Li B (2017a) Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment. Appl Soft Comput 59:288–302

Deng W, Zhao H, Zou L, Li G, Yang X, Wu D (2017b) A novel collaborative optimization algorithm in solving complex optimization problems. Soft Comput 21(15):4387–4398

El-Melegy MT, Ahmed SM (2007) Neural networks in multiple classifier systems for remote-sensing image classification. In: Nachtegael M, Van der Weken D, Kerre EE, Philips W (eds) Soft computing in image processing. Springer, Berlin, pp 65–94

Fong TC, Wei WJ (2008) Using neural network ensembles for bankruptcy prediction and credit scoring. Expert Syst Appl 34(4):2639–2649

Foody G (2002) Status of land cover classification accuracy assessment. Remote Sens Environ 80(1):185–201

Gabrys B, Bargiela A (2000) General fuzzy min-max neural network for clustering and classification. IEEE Trans Neural Netw 11(3):769–783

Ganivada A, Dutta S, Pal SK (2011) Fuzzy rough granular neural networks, fuzzy granules, and classification. Theor Comput Sci 412:5834–5853

Gu B, Sheng VS (2017) A robust regularization path algorithm for v-support vector classification. IEEE Trans Neural Netw Learn Syst 28(5):1241–1248

Gu B, Sheng VS, Tay KY, Romano W, Li S (2015) Incremental support vector learning for ordinal regression. IEEE Trans Neural Netw Learn Syst 26(7):1403–1416

Gu B, Sun X, Sheng VS (2017) Structural minimax probability machine. IEEE Trans Neural Netw Learn Syst 28:1646–1656

ISRO (2011) National remote sensing center open earth observation data archive. http://bhuvan.nrsc.gov.in

Jatindra KD, Sudipta M, Rahul D (2016) Multiple classifier system using classification confidence for texture classification. Multimedia Tools Appl 76(2):2535–2556

Kong Y, Zhang M, Ye D (2017) A belief propagation-based method for task allocation in open and dynamic cloud environments. Knowl Based Syst 115:123–132

Kumar DA, Meher SK (2013) Multiple classifiers systems with granular neural networks. In: 2013 IEEE international conference on signal processing, computing and control (ISPCC). IEEE, pp 1–5

Kumar DA, Meher SK (2014) Granular neural networks models with class-belonging granulation. In: 2014 international conference on contemporary computing and informatics (IC3I). IEEE, pp 1198–1202

Kumar DA, Kumari KP, Meher SK (2016a) Progressive granular neural networks with class based granulation. In: 2016 IEEE annual on India conference (INDICON). IEEE, pp 1–6

Kumar DA, Meher SK, Kanhar D, Kumari KP (2016b) Unified granular neural networks for pattern classification. Neurocomputing 216:109–125

Kuncheva L (2004) Combining pattern classifiers: methods and algorithms, 1st edn. Wiley, Hoboken

Leite D, Costa P, Gomide F (2009) Evolving granular classification neural networks. In: 2009 international joint conference on neural networks. IEEE, pp 1736–1743

Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml

Liu Q, Cai W, Shen J, Fu Z, Liu X, Linge N (2016) A speculative approach to spatial–temporal efficiency with multi-objective optimization in a heterogeneous cloud environment. Secur Commun Netw 9(17):4002–4012

Ma T, Zhang Y, Cao J, Shen J, Tang M, Tian Y, Al-Dhelaan A, Al-Rodhaan M (2015) \varvec{\textit{KDVEM}}: a k-degree anonymity with vertex and edge modification algorithm. Computing 97(12):1165–1184

Ma T, Wang Y, Tang M, Cao J, Tian Y, Al-Dhelaan A, Al-Rodhaan M (2016) Led: a fast overlapping communities detection algorithm based on structural clustering. Neurocomputing 207:488–500

Meher SK (2014) Explicit rough-fuzzy pattern classification model. Pattern Recognit Lett 36:54–61

Meher SK, Kumar DA (2015) Ensemble of adaptive rule-based granular neural network classifiers for multispectral remote sensing images. IEEE J Sel Top Appl Earth Observ 8(5):2222–2231

Nandedkar AV, Biswas PK (2009) A granular reflex fuzzy min–max neural network for classification. IEEE Trans Neural Netw 20(7):1117–1134

Pal SK, Mitra S (1992) Multilayer perceptron, fuzzy sets, and classification. IEEE Trans Neural Netw 3:683–697

Pal SK, Meher SK, Dutta S (2012) Class-dependent rough-fuzzy granular space, dispersion index and classification. Pattern Recognit 45:2690–2707

Pedrycz W (2013) Granular computing: analysis and design of intelligent systems, vol 1. CRC Press, Boca Raton

Pedrycz W, Vukovich G (2001) Granular neural networks. Neurocomputing 36:205–224

Quteishat A, Lim C (2008) A modified fuzzy min–max neural network with rule extraction and its application to fault detection and classification. Appl Soft Comput 8:985–995

Richards JA, Jia X (2006) Remote sensing digital image analysis: an introduction, 4th edn. Springer, Berlin

Simpson P (1992) Fuzzy min–max neural networks. Part I: classification. IEEE Trans Neural Netw 3(5):776–786

Simpson P (1993) Fuzzy min–max neural networks. Part II: clustering. IEEE Trans Fuzzy Syst 1(1):32–45

Stathakis D, Kanellopoulos I (2008) Global elevation ancillary data for land use classification using granular neural networks. Photogramm Eng Remote Sens 74:55–63

Stathakis D, Vasilakos A (2006) Satellite image classification using granular neural networks. Int J Remote Sens 27:3991–4003

Stehman S (1996) Estimating the kappa coefficient and its variance under stratified random sampling. Photogramm Eng Remote Sens 62(4):401–407

Tian Q, Chen S (2017) Cross-heterogeneous-database age estimation through correlation representation learning. Neurocomputing 238:286–295

Wang B, Gu X, Ma L, Yan S (2017) Temperature error correction based on bp neural network in meteorological wireless sensor network. Int J Sens Netw 23(4):265–278

Xue Y, Jiang J, Zhao B, Ma T (2017) A self-adaptive artificial bee colony algorithm based on global best for global optimization. Soft Comput. https://doi.org/10.1007/s00500-017-2547-1

Zadeh LA (1965) Fuzzy sets. Inf Control 8:338–353

Zhang G (2000) Neural networks for classification: a survey. IEEE Trans Syst Man Cybern Part C (Appl Rev) 4(30):451–462

Zhang J, Tang J, Wang T, Chen F (2017) Energy-efficient data-gathering rendezvous algorithms with mobile sinks for wireless sensor networks. Int J Sens Netw 23(4):248–257

Zhi H, Jianxin W, Wei T (2002) Ensembling neural networks: many could be better than all. Artif Intell 137(1):239–263

Zhou Z, Wang Y, Wu QJ, Yang CN, Sun X (2017) Effective and efficient global context verification for image copy detection. IEEE Trans Inf Forensics Secur 12(1):48–63