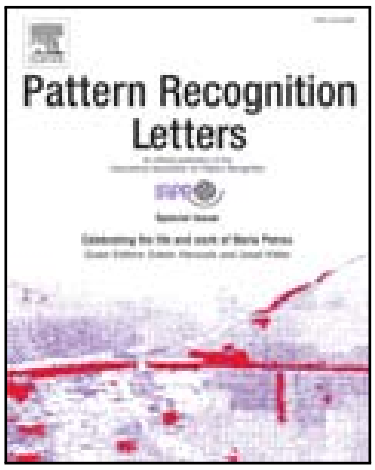

A New Method for Constructing Granular Neural Networks Based on Rule Extraction and Extreme Learning Machine

Xinzheng Xu, Guanying Wang, Shifei Ding, Xiangying Jiang, Zuopeng Zhao

PII: S0167-8655(15)00151-8
DOI: 10.1016/j.patrec.2015.05.006
Reference: PATREC 6225

To appear in: *Pattern Recognition Letters*

Received date: 18 September 2014
Accepted date: 22 May 2015

Please cite this article as: Xinzheng Xu, Guanying Wang, Shifei Ding, Xiangying Jiang, Zuopeng Zhao, A New Method for Constructing Granular Neural Networks Based on Rule Extraction and Extreme Learning Machine, *Pattern Recognition Letters* (2015), doi: 10.1016/j.patrec.2015.05.006

Pattern Recognition Letters
journal homepage: www.elsevier.com

A New Method for Constructing Granular Neural Networks Based on Rule Extraction and Extreme Learning Machine

Xinzheng Xu[a,b,**], Guanying Wang[a], Shifei Ding[a,b], Xiangying Jiang[a], Zuopeng Zhao[a,b]

[a] *School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, 221116, China*
[b] *Jiangsu Key Laboratory of Mine Mechanical and Electrical Equipment, China University of Mining & Technology, Xuzhou, 221116, China*

ABSTRACT

This paper introduces a framework of granular neural networks named rough rule granular extreme learning machine (RRGELM), and develops its comprehensive design process. The proposed granular neural networks are formed on the basis of rough decision rules extracted from training samples through rough set theory. Firstly, Sample data are reduced by the algorithms of attributes reduction and attributes values reduction in rough set theory, and then they are compressed to an irredundant data set. In this data set, each sample can represent a rough rule, and is expressed as an If-Then rule which indicates the relationship between the input and output pattern. Moreover, the confidence level and the coverage level of each rule are calculated. Secondly, granular-neurons can be constructed through the If-Then rules, and all the granular-neurons constitute rule matching layer which is regarded as the hidden layer of the RRGELM. The linked weights between the input neurons and granular-neurons can be determined by the confidences of rough decision rules, while the linked weights between the output neurons and granular-neurons can be initialized as the contributions of the rough rules to the classification. Finally, the extreme learning machine (ELM) algorithm is introduced to improve the learning speed of the RRGELM, rather than the BP algorithm used by other traditional GNN models. Good performance of the proposed RRGELM is demonstrated on several well-known benchmark data sets.

1. Introduction

As an important class of nonlinear and highly adaptable systems, artificial neural networks (ANNs) display many excellent performances throughout the applications of many areas, such as parallel processing, adaptivity, robustness, ruggedness, generalization, and insensitivity to noise. Generally speaking, ANNs are regarded as black boxes and nontransparent models owing to the distributed style of the underlying information processing. As a consequence, it is very difficult to explain their internal learning processes.

Granular computing (GrC) (Zadeh, 1997) is an emerging computation theory for effectively using granules, such as classes, clusters, subsets, groups and intervals, to build an efficient computational model for complex applications with huge amounts of data, information and knowledge. Specifically, GrC attempts to imitate the human ability to reason with abstract quantities and groups rather than numeric precision (Dick et al., 2013). GrC provides a novel framework for information processing, in which computation and operahere are a number of GrC models, such as sets (interval analysis), rough set, fuzzy set, shadowed sets and probability calculus (Ganivada et al., 2011). There are a number of GrC models, such as sets (interval analysis), rough set, fuzzy set, shadowed sets and probability calculus (Pedrycz and Vukovich, 2001).

As the organic integration of above two technologies, Granular neural networks (GNN), introduced by Pedrycz and Vukovich (Pedrycz and Vukovich, 2001), is a framework to process information granules. By considering sets of objects sharing commonalities and imprecise data items instead of precise singular data items, GNN avoids processing detailed and costly data (Leite et al., 2013). GNN does not need to consider all data, which are far more numerous than granules. Furthermore,

**Corresponding author: Tel.: +86-159-5215-1616;
e-mail: xuxinzh@163.com (Xinzheng Xu)

data can be discarded whenever they match an existing granule. For the past few years, several researchers have described different architectures of GNN. Zhang et al. described two different types of GNN, crisp GNN (CGNN) built on CNN and fuzzy GNN (FGNN) built on FNN in (Zhang et al., 2000), to deal with numerical-linguistic data fusion and granular knowledge discovery in numerical-linguistic databases. Pal et al. presented a rough self-organizing map (RSOM) with fuzzy discretization of feature space (Pal et al., 2004). GNN with the high-speed evolutionary interval learning is designed by Zhang et al., to deal with different membership functions of the same linguistic term (Zhang et al., 2008). Ganivada et al. introduced a fuzzy rough GNN (FRGNN) model based on the multilayer perceptron using a back-propagation algorithm for the fuzzy classification of patterns (Ganivada et al., 2011). Leite et al. proposed an evolving GNN (eGNN) model, which can build interpretable multi-sized local models using fuzzy neurons for information fusion using an online incremental learning algorithm (Leite et al., 2013). Song et al. proposed a concept of a granular neural network and develop its comprehensive design process, which is elaborated on with the two main aspects stressed, namely an optimal allocation of information granularity and an objective function guiding the allocation process (Mingli and Pedrycz, 2013). Furthermore, in the latest decade, GNN had been used to many fields successfully, such as approximation and prediction of wages (Marček and Marček, 2008), land use classification (Vasilakos and Stathakis, 2005), modeling of the charging characteristic (Park et al., 2012), stock prediction (Zhang et al., 2002), fast credit card fraud detection (Syeda et al., 2002), satellite image classification (Stathakis and Vasilakos, 2006), imbalanced data classification (Chen et al., 2008), semi-supervised data stream classification (Leite et al., 2010), fuzzy time series forecasting (Leite et al., 2012), early warning of enterprise decline (Cao et al., 2011) and so on.

In the paper, we introduce a method for constructing GNN within the natural computing paradigm, where the structure of granulation is defined on the base of rough set theory. As a typical model of Grc, rough set (RS) theory has proved to be a powerful tool for uncertainty questions and it has been applied to data reduction, rule extraction, data mining and granularity computation (Cao et al., 2011). RS apply the unclear relation and data pattern comparison based on the concept of an information system with indiscernible data, where the data is uncertain or inconsistent (Cheng et al., 2010). Recently, rough set theory has integrated with neural networks, called rough neural networks (RNN) (Cheng et al., 2010; Zhang et al., 2009; Zhao and Zhang, 2011; Hassan, 2011; Chang et al., 2012; Valdés et al., 2012; Xu et al., 2012; Lin et al., 2013).. Generally speaking, there are two integration patterns for RNN, one typical approach is to use RS approach as a pre-processing unit, and another is that RS is used to gain rules from data sets to construct granular-neurons. In the framework of granular-neurons, integration research has been done in the use of rough set for encoding weights of knowledge-based networks. However, the first integration pattern has been considered so far.

This article is an attempt to incorporate rough set methodology in the framework of granular-neurons to construct RNN using rules obtained by RS. In the proposed method, the attributions of data sets are firstly reduced by RS theory to remove redundant attributes. Then, on the base of reduced data sets, the reduction of attribution values is followed to remove redundant samples. After above two processes, each sample represents a rough decision rules, and all rough decision rules are extracted from reduced samples. Furthermore, the confidence level and the coverage level of each decision rule can be calculated. After this, granular-neurons can be constructed by rough decision rules, and the weights between input neurons and granular-neurons can be determined by confidences of decision rules. Here, we name the layer which composed of granular-neurons as the rule matching layer. So, the proposed model contains three layers: the input layer, the rule matching layer, and the output layer. In addition, the extreme learning machine (ELM) algorithm, developed by G.-B. Huang et al (Huang et al., 2006), is introduced to replace the traditional back propagation (BP) algorithm, which is often used by other GNN models.

The remainder of this paper is structured as follows. A brief description of rough set theory and the ELM algorithm are provided in Section 2. In section 3, we introduce extraction processes of decision rules from the decision table. We use Zoo data set from UCI data sets (Blake and Merz, 1998) as a straightforward example to explain how to gain If-Then rules through attribute reduction and value reduction. Structures and learning process of RRGELM are detailed in Section 4. We present our performance analysis experiments in Section 5. The paper is concluded in Section 6.

2. Theoretical Backgrounds

2.1. Rough Set Theory

2.1.1. Basic Conception

The basic concept in rough set theory is an information system which can be expressed by a 4-tuple $S = \langle U, A, U, f\rangle$, where U is a finite set of objects, called the universe; A is a finite set of attributes,$V = \cup_{a\in A} V_a$ is a domain of attribute a, and $f : U \times A \to V$ is an information function such that $f(x, a) \in V_a$, for $\forall a \in A$, $\forall x \in U$. In the classification problems, an information system is also seen as a decision table assuming that $A = C \cup D$ and $C \cup D = \varnothing$, where C is a set of condition attributes and D is a set of decision attributes.

Let $S = \langle U, A, V, f\rangle$ be an information system, every $P \subseteq A$ generates a binary relation on U called an indiscernibility relation, denoted by $IND(P)$ which is defined as follows:

$$IND(P) = \{(x, y) \in U \times U : f(x, a) = f(y, a), \forall a \in P\} \quad (1)$$

The partition of U generated by $IND(P)$ is denoted by $U/IND(P) = C_1, C_2, ..., C_k$, for every C_i is an equivalence class. For $\forall x \in U$ the equivalence class of x in relation $U/IND(P)$ is defined as follows:

$$[x]_{IND(P)} = \{y \in U : f(x, a) = f(y, a), \forall a \in P\} \quad (2)$$

Let $X \subseteq U$ be a target set and $P \subseteq A$ be an attribute subset, The P-lower approximation of X(denoted by P_*X) and P-upper

approximation of X (denoted by P^*X), are defined respectively as follows:

$$P_*X = \{y \in U : [y]_{IND(P)} \subseteq X\} \tag{3}$$

$$P^*X = \{y \in U : [y]_{IND(P)} \cap X \neq \varnothing\} \tag{4}$$

The P-boundary (doubtful region) of set X is defined as follows:

$$Bn_p(X) = P^*X - P_*X \tag{5}$$

P_*X is the set of all objects from U which can be certainly classified as elements of X, employing the set of attributes P. P^*X is the set of objects from U which can be possibly classified as elements of X, using the set of attributes P.$Bn_p(X)$ is the set of objects which cannot be certainly classified to X using the set of attributes Ponly.

2.1.2. Attribute Reduction of Decision Table

Let $P, Q \subseteq A$, the dependency of attribute set Q on attribute set P, $\gamma_P(Q)$, is given by

$$\gamma_P(Q) = \frac{card(\bigcup_{x \in U/IDP(Q)} P_*X)}{card(U)} \tag{6}$$

where $\bigcup_{x \in U/IDP(Q)} P_*X$ can be denoted as $POS_P(Q)$, which means that the objects in it can be classified to one class of the classification$U/IDP(Q)$ by attribute P.

An attribute a is said to be dispensable in $P$with respect to Q, if $\gamma_P(Q) = \gamma_{P-\{a\}}(Q)$; otherwise a is an indispensable attribute in P with respect to Q.

Let $S = \langle U, A, V, f\rangle$ be a decision table, the set of attributes$P(P \subseteq C)$ is a reduction of attribute C, which satisfies the following conditions:

$$\gamma_P(Q) = \gamma_C(Q) \; and \; \gamma_P(\mathcal{D}) \neq \gamma_{P'}(Q), \forall P' \subset P \tag{7}$$

A reduction of condition attributes C is a subset that can discern decision classes with the same accuracy as C, and none of the attributes in the reduction can be eliminated without decreasing its distrainable capability.

2.1.3. Value Reduction of Decision Table

After attribute reduction of the decision table, redundant attributes in decision table are removed and those attributes which can influence the classification ability are preserved. Those attributes and attribute sets which have direct impact on decision classification can be picked up immediately based on the reduced decision table. However, the attribute reduction algorithm can only remove the redundant attributes, so other redundant information cannot be removed. Value reduction furthers the procession of the reduced decision table after attribute reduction. Typical value reduction algorithms include general the value reduction algorithm and heuristic value reduction algorithm. Next, we introduce the latter.

The procedures of the heuristic value reduction algorithm are detailed as follows:

Input: decision table T (This table includes n sample, m-1 conditional attributes and one decision attribute)

Output: T' (The reduced talbe of T)

STEP 1 : Detect the conditional attributes per column in the decision table. For each column, if it brings about conflict records after deleting the column, then keep the attribute value as before. And if there are repeated records, then mark the attribute value with symbol"*". For other samples, mark the attribute value with symbol "?".

STEP 2 : Delete repeated records and check each record which marked with symbol "?". If it can determine the results of decision just based on those attribute values of marked records, then replace symbol "?"with symbol "*". Otherwise, symbol "?" is replaced with former value.

STEP 3 : Delete all of records which conditional attributes marked with symbol "*" and those would be repeated.

STEP 4 : If there exist two records only differ in one conditional attribute value, and one of records is marked with symbol "*", for this record, if its decision result can be detected based on other unmarked conditional attribute values then delete the other record. Otherwise, delete this record.

New decision table is put forward by mentioned value reduction algorithm. All of attribute values in the table are value kernels of the table and each record is a rough decision rule.

2.2. Extreme Learning Machine

ELM, as a novel training algorithm for SLFNs, is very efficient and effective. In this section, we will give a brief review of ELM algorithm (Huang et al., 2012).

Given N distinct training samples $(x_i, t_i) \in R^n \times R^m (i = 1, 2, \ldots, N)$ £the output of a SLFN with $\widetilde{N}$ hidden nodes (additive or RBF nodes) can be represented by

$$O_j = \sum_{i=1}^{\widetilde{N}} \beta_i f_i(x_i) = \sum_{i=1}^{\widetilde{N}} \beta_i f(x_j; a_i; b_i), \quad j = 1, \ldots, N \tag{8}$$

where O_j is the output vector of the SLFN with respect to the input sample x_i .$a_i = [a_{i1}, a_{i2}, ..., a_{in}]^T$ and b_i are learning parameters generated randomly of the jth hidden node, respectively. $\beta_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{in}]^T$ is the link connecting the jth hidden node and the output nodes. $f(x_j; a_i; b_i)$ is the activation function of the original ELM.

Set $a_i \cdot x_i$ be the inner product of a_i and x_i . Eq. (1) can be written compactly as

$$H\beta = O \tag{9}$$

where

$$f(x) = \begin{bmatrix} f(a_1 \cdot x_1 + b_1) \ldots & f(a_{\widetilde{N}} \cdot x_1 + b_N) \\ \vdots & \ldots \quad \vdots \\ f(a_1 \cdot x_N + b_1) \ldots & f(a_{\widetilde{N}} \cdot x_N + b_N) \end{bmatrix}_{N \times \widetilde{N}},$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_N^T \end{bmatrix}_{\widetilde{N} \times m}, \qquad O = \begin{bmatrix} O_1^T \\ \vdots \\ O_N^T \end{bmatrix}_{N \times m}$$

Here, H is called the output matrix of the hidden layer.

To minimize the network cost function$\| O - T \|$, ELM theories claim that the hidden nodes learning parameters $a - i$ and $a - i$ can be assigned randomly without considering the input data. Then, Eq.(9) becomes a linear system and the output

weights β can be analytically determined by finding a least-square solution as follows

$$\widehat{\beta} = H^{+}T \tag{10}$$

where H^{+} is the Moore-Penrose generalized inverse of H. So, calculation of the output weights is done by a mathematical transformation which avoids any lengthy training phrase where the parameters of the network are adjusted iteratively with some appropriate learning parameters (such as learning rate and iterations.).

Thus, the three-step ELM algorithm can be summarized as follows (Ding et al., 2014).

ELM Algorithm:

Input: a training set$(x_i, t_i) \in R^n \times R^m (i = 1, 2, \ldots, N)$, the activation function f, and the hidden node number $\widetilde{N}$.

Output: the output weights β .

Step1. Randomly assign the parameters of hidden nodes$(a_i, b_i), i = 1, \ldots, \widetilde{N}$.

Step2. Calculate the output matrix of the hidden layer H.

Step3. Calculate the output weight $\beta : \beta = H^{+}T$

3. Extraction Processes of Rough Decision Rules

The decision rules are easy to extract after the attribute reduction and value reduction of the samples.

Definition 1 (Decision Rules) Giving a decision table$DT = (U, C \cup D, V, f)$, and then using $X \in U/IND(C)$, $Y \in U/IND(D)$,$\forall x \in X$, $des(X) = \widehat{\forall \alpha \in C(\alpha, \alpha(x))}$ to describe equivalence class X and $des(Y) = \widehat{\forall \beta \in C(\beta, \beta(x))}$ to describe equivalence class Y, then define

$$r : des(X) \rightarrow des(Y), Y \cap X \neq \varnothing \tag{11}$$

as the decision rule from X to Y.

Definition 2 (Confidence and Coverage Level of Decision Rules) Giving a decision table $DT = (U, C \cup D, V, f)$, and then using $X \in U/IND(C)$, $Y \in U/IND(D)$,$\forall x \in X$, $des(X) \rightarrow des(Y)(Y \cap X \neq \varnothing)$ to represent the decision rule r from X to Y, then define

$$\mu(X, Y) = \frac{\mid X \cap Y \mid}{\mid X \mid} \tag{12}$$

as the confidence level of decissec1ion rule r which also called rule strength. And define

$$\psi(X, Y) = \frac{\mid X \cap Y \mid}{\mid Y \mid} \tag{13}$$

as the coverage of decision rule r.

Apparently, $0 < \mu(X, Y) \leq 1$, $0 < \psi(X, Y) \leq 1$. The confidence level of the decision rule shows the possibility of the rule consequent to be valid when the rule antecedent is true. It means the probability estimation of getting correct conclusion based on mentioned rules. When the confidence level of decision rule $\mu(X, Y) = 1$, it means this rule is correct. When $0 < \mu(X, Y) < 1$, it means more than one conclusion can be extract based on the condition of the rule, so this rule is indefinite. However, the confidence level of decision rule cannot reflect the coverage of this rule in decision table, which means percentage of samples base on mentioned rules performed in all samples.

While, the coverage of the decision rule describes the percentage of samples which met the rule in the same class. When the coverage of the decision rule $\psi(X, Y) = 1$, it means this rule cover all samples which meet the decision rule consequent. When $0 < \mu(X, Y) < 1$, it shows that the percentage of samples which meet this rule performed in all samples that meet the rule consequent. If the coverage of the decision rule is tiny, it indicates that only a minority of samples meet the antecedent and consequent of this rule at the same time, so the causal connection between conditions and conclusions of the mentioned rule have not enough samples to support.

To further illustrate the procedure of extracting decision rule, we use Zoo data set from UCI data sets as a straightforward example for following analysis. After the attribute reduction and value reduction, reduced Zoo data set is shown in Table 1.

Fourteen decision rules which are applicable to Zoo data set can be detected based on Table 1 as following:

R1: If $a_4 = 1$ then $d = 1$

R2: If $a_3 = 1$ and $a_{13} = 2$then $d = 2$

R3: If $a_3 = 1$ and $a_6 = 0$ and $a_8 = 1$ then $d = 3$

R4: If $a_3 = 1$ and $a_6 = 0$ and $a_{13} = 4$ then $d = 3$

R5: If $a_3 = 1$ and $a_6 = 1$ and $a_8 = 1$ and $a_{13} = 0$ then $d = 4$

R6: If $_= 1$ and $a_6 = 1$ and $a_8 = 1$ and $a_{13} = 4$ then $d = 5$

R7: If $a_6 = 0$ and $a_{13} = 6$ then $d = 6$

R8: If $a_6 = 1$ and $a_{13} = 6$ then $d = 7$

R9: If $a_{13} = 5$ then $d = 7$

R10: If $a_8 = 0$ and $a_{13} = 0$ then $d = 7$

R11: If $a_3 = 0$ and $a_4 = 0$ and $a_6 = 1$ then $d = 3$

R12: If $a_4 = 0$ and $a_6 = 1$ and $a_8 = 0$ and $a_{13} = 4$ then $d = 7$

R13: If $a_{13} = 8$ then $d = 7$

R14: If $a_3 = 0$ and $a_4 = 0$ and $a_6 = 0$ then $d = 7$

where $R_i (i = 1, 2, , 14)$ represents the code name of the $i - th$ decision rule, ai is the $i - th$ condition attribute of the Zoo data set, and d is the value of the decision attribute.

The confidence and coverage level of decision rules extracted from Table 1 are shown in Table 2. According to Table 2, the confidence level of all rules is 100% which means the rule consequent is valid when rule antecedent is satisfied. However, not all of the coverage of rules meets 100%. For example, the decision rules of class 3 and class 7 whose coverage levels are not meet 100%. The coverage indicates the representative degree of the mentioned rule for decision the class represented by the rule consequent, and it shows the percentage of samples for this rule performed in all samples in this class.

In fact, the confidence level shows the possibility of the rule consequent to be valid when the rule antecedent is true. For example, the confidence level (100%) of the rule R1 in the Table 2, means that the samples whose fourth attribution equals 1 in the Zoo data set belong to the class 1 in the probability of 100%. However, the coverage describes the percentage of those samples which fit the rule in the same class. For instance, sixty percent of the samples belonging to the class 3 supports the R_3, and forty percent supports the R_4.

Table 1. Zoo dataset after attribute reduction and attribute value reduction

U	Condition Attributes					Decision
	a_3	a_4	a_6	a_8	a_{13}	Attributes(d)
1	–	1	–	–	–	1
2	1	–	–	–	2	2
3	1	–	0	1	–	3
4	1	–	0	–	4	3
5	1	–	1	1	0	4
6	1	–	1	1	4	5
7	–	–	0	–	6	6
8	–	–	1	–	6	7
9	–	–	–	–	5	7
10	–	–	–	0	0	7
11	0	0	1	–	–	3
12	–	0	1	0	4	7
13	–	–	–	–	8	7
14	0	0	0	–	–	7

Table 2. Confidence and coverage coefficients of rules extracted from Zoo data set

Decision Rule	Decision Class	Confidence level(%)	Coverage(%)
R_1	1	100	100
R_2	2	100	100
R_3	3	100	60
R_4	3	100	40
R_5	4	100	100
R_6	5	100	100
R_7	6	100	100
R_8	7	100	20
R_9	7	100	10
R_{10}	7	100	40
R_{11}	3	100	20
R_{12}	7	100	10
R_{13}	7	100	20
R_{14}	7	100	10

4. Structures and Learning Process of RRGELM

4.1. Granular-neurons based on Rough Rules

4.1.1. Constructor of Granular-neurons

Each granular neuron indicates a decision rule, and this decision rule corresponds to a granular division in the decision table.

Taking Zoo data as an example, the decision rule, according to Table 1, can construct the corresponding neuron of the rule matching layer. There are 14 rough decision rules totally, so there are 14 neurons in the rule matching layer. Among those neurons, the first neuron represents the decision rule R_1£the second neuron represents the decision rule R_2£on the analogy of this, 14^{th} neuron represents the decision rule $R_1$4£Like this, the constructor of neurons in the rule matching layer is done.

The liner output function is used as the activation function of granular-neuron which is shown as as Eq.(14) shows:

$$f(x) = x \tag{14}$$

4.1.2. Input and Output of Granular-neurons

After the linked weights between the neurons of rule matching layer and that of the input layer are determined, the input of each granular-neuron of rule matching layer can be determined based on input samples. And then, judge whether the input sample meet corresponding rule antecedent or not. If they meet, according to Eq.(7), the output of granular-neuron is the confidence level of this rule(calculate by Eq.(5)). Otherwise, the output of this neuron is 0.

For example, given the input sample $X = \{a_3 = 1, a_4 = 1, a_6 = 0, a_8 = 0, a_{13} = 1,\}$ and linked weight $\omega_1 = \{0, 1, 0, 0, 0\}$, the input of the first neuron of the rule matching layer is $\{0, 1, 0, 0, 0\}$. That is because it meet the corresponding rule R_1 of the granular-neuron ($If\ a_4 = 1\ then\ d = 1$), so the output of the neuron is 1.

4.2. Structures of RRGELM

RRGELM is defined as the three-layer networks, as Fig.1shows:

RRGELM consists of three layers.

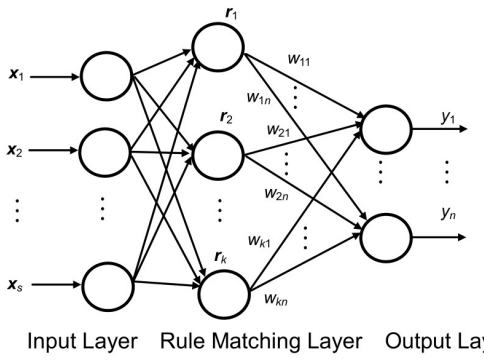

Fig. 1. Structure of RRGELM.

The first layer is the input layer. The input sample $X = (x_1, x_2, \ldots, x_m)$ has been simplified into $X' = (x_1, x_2, \ldots, x_s)(s \leq m)$ by the attribute reduction algorithm, and regarded as the input of RRGELM.

The second layer is the rule matching layer. The number of granular neurons in this layer depends on the amount of decision rules. $r_i(i = 1, 2, \cdots, k)$ indicates the code of neurons in the rule matching layer, and k indicates number of neurons in the rule matching layer (Obviously, k equals to the amount the decision rules).

The third layer is the output layer. $Y(y_1, y_2, \ldots, y_n)$ means the output of RRGELM, and n is the number of neurons (Obviously, n equals to the number of classification).

Different with other traditional ANN models, the hidden layer in the RRGELM model is replaced by the rule matching layer which is constituted by granular-neurons. Further, the linked weights in RRGELM model are not random assignment anymore. The linking weights between neurons of the rule matching layer and that of the input layer are mainly used for the reduction of input samples to meet the antecedent of rules. The linked weights between the rule matching layer and the input layer are defined based on the strength of rules in order to evaluate the contribution of each rule to decisions.

4.3. Learning Process of the RRGELM

The parameters in the model of the RRGELM to be valued during the training process include the number of neurons in each layer, activation functions, thresholds and linked weights among neurons in each layer.

4.3.1. Linked Weights between Granular-neurons and Input Neurons

The linked weights between granular-neurons of rule matching layer and that of input layer are mainly used for reduction of input samples, in order to make it meet the expression form of rule antecedents. For the reduced input samples $X = (x_1, x_2, \ldots, x_s)$ and the rule set $R = (R_1, R_2, \ldots, R_k)$, the number of neurons of rule matching layer is k, and the number of input neurons is s. Let ω_{ij} indicates the linked weights between the j-th granular-neuron and the i-th input neuron. Then,

$$\omega_{ji} = \begin{cases} 1 & if\ x_i \in R_j \\ 0 & if\ x_i \notin R_j \end{cases} \quad i = 1, 2, \cdots, s;\ j = 1, 2, \cdots, k \quad (15)$$

For example, it can be extracted that the linking weight between the first granular-neuron and input neurons is $\omega_1 = \{0, 1, 0, 0, 0\}$ according to the rule R1 of Zoo data set and the input sample $X = \{a_3 = 1, a_4 = 1, a_6 = 0, a_8 = 0, a_{13} = 1\}$. So the linked weights between granular-neurons and input neurons can be deduced by the same way. Then, when the RRGELM is used for Zoo data set, the input linked weights matrix ω is defined as following:

$$\omega = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

4.3.2. Initial Linked Weights between Granular-neurons and Output Neurons

The initial linked weights between granular-neurons and output neurons can be defined based on the contributions of rules to classification decisions. If the consequent of one rule is the same with the corresponding classification of the output neuron, then the initial linked weight between them is 1, otherwise it is 0.

For example, the rule consequent of R_1 ($If\ a_4 = 1\ then\ d = 1$) based on Zoo data set is 1,then the initial linked weight between the corresponding granular-neuron and the output neuron which represents the first class in the output layer is 1, and the initial linking weights between the granular-neuron and other output neurons is 0. That is, the initial linked weight between the first output neuron and granular-neurons is $W_1 = \{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$. So the linked weights between other output neurons and granular-neurons can be deduced by the same way. At this way, when the model of the RRGELM is used for Zoo data set, the output linked weights matrix W is calculated as following:

$$W = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3.3. Structure of Neurons in the Output Layer

The number of output neurons is determined by the number of sample classes, and it generally equals to the number of classes.When input samples are input into the RRGELM, the outputs of the rule matching layer can be calculated through above method. Then, according to the outputs of the rule matching layer and output linked weights, the corresponding neuron of the output layer is fired, and its output is 1 while other outputs equal to 0. In the RRGELM, the sigmoid function is used as the activation function of neurons in output layer as Eq. (16) shows:

$$f(x) = hard\lim s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{16}$$

4.3.4. Learning Algorithm of the RRGELM

After the procedures mentioned above, the RRGELM is constructed and initialized. If the decision rules are abundant, the RRGELM have good classification ability without learning. However, if decision rules are not enough, the RRGELM cannot distinguish those samples which dont meet or fully meet existing decision rules, so the RRGELM need further learning to adjust output linked weights.

Considering that the RRGELM is a typical SLFNs, we introduce ELM algorithm to speed up the learning process of the RRGELM.

The procedures of the algorithm for RRGELM are as following:

STEP1: Pre-treatment. Put the data set into two parts, one is as training samples, another is as testing samples. Decision rules are extracted from the reduced training samples through the attribute reduction algorithm and value reduction algorithm.

STEP2: The number of input neurons, granular-neurons and output neurons are deduced by the number of conditional attributes, decision rules and decision attributes, respectively. Extract input linking weights and output linking weights according to decision rules. Then, calculate the confidence and coverage level of each rule.

STEP3: Input the reduced training samples, and then compute the output matrix H of granular-neurons through input linked weights and the confidence levels of rough decision rules.

STEP4: Calculate the output weights β according to Eq.(10).

STEP5: Check the generalization error and classification precision of the RRGELM according to testing samples.

5. Experiments and Analyses

In the experiments of this section, several well-known benchmark data sets from UCI machine learning data sets Blake and Merz (1998), including Zoo data set, Balance Scale data set, Soybean(small) data set and Car Evaluation data set, are used to verify the correctness, reliability and generalization ability of the RRGELM. The basic features of above mentioned data sets are shown in Table 3.

Table 3. Basic features of four data sets

Data Sets	Number of Samples	Amount of Decision Classes	Number of Attributes
Zoo	101	7	16
Balance Scale	625	3	4
Soybean(small)	47	4	35
Car Evaluation	1728	4	5

Zoo data set includes 101 samples, and each sample consists of 16 conditional attributions and one decision attribution. All samples are classified into 7 classes.

Balance Scale data set includes 625 samples, and each sample consist 4 components: Left-Weight, Left-Distance, Right-Weigh and Right-Distance. The classification of each sample in Balance Scale data set can be extracted based on mentioned 4 components. When $Left\text{-}Weight * Left\text{-}Distance > Right\text{-}Weigh * Right\text{-}Distance$, the classification of this sample is L (It means the left weight is larger). When $Right\text{-}Weigh * Right\text{-}Distance > Left\text{-}Weight * Left\text{-}Distance$, the classification of this sample is R (It means the right weight is larger). When $Left\text{-}Weight * Left\text{-}Distance = Right\text{-}Weigh * Right\text{-}Distance$, the classification of this sample is B (It means two sides have same weight, and the scale is in balance). In Balance Scale data set, there are 288, 462 and 88 samples in three classifications respectively.

Soybean (small) data set is a subset of Soybean data set as well as a typical multi-classification data set which includes 47 samples, and each sample consist 35 attributes. All samples are classified into 4 classes.

Car Evaluation data set includes 1728 samples, and each sample consists of 6 attributions: Buying, Maint, Doors, Persons, Lug_boot and Safety. All samples are classified into 4 classes to indicate the acceptable level of cars: unacc, acc, good, vgood.

Next, the RRGELM is constructed and trained by training samples mainly to test the training precision of the networks. The samples for networks performance test are all testing samples.

Firstly, the data sets are processed by attribute reduction. The features of data set after attribute reduction are as Table 4 shows.

Table 4. Number of attributes after attributes reduction

Data Sets	Number of Original Attributes	Number of Reduced Attributes
Zoo	16	6
Balance Scale	4	3
Soybean(small)	35	4
Car Evaluation	6	6

Then, use the value reduction algorithm to process the data sets which had been reduced by attribute reduction. The result is as Table 5 shows.

Table 5. Number of samples after attributes value reduction

Data Sets	Number of Original Samples	Number of Reduced Samples
Zoo	101	14
Balance Scale	625	303
Soybean(small)	47	17
Car Evaluation	1728	246

According to the result of value reduction, the rules which have the structure as If...Then...can be extracted. The number of rules equals to the number of samples in data set which processed by value reduction. Rule antecedents consist of attributes and their values of reduced samples, and the rule consequents are the classification labels of the reduced samples. Based on Table 5, the number of rules extracted from Zoo data set, Balance Scale data set, Soybean (small) data set and Car Evaluation data set after attribute reduction and value reduction are 14, 303, 17 and 246, respectively.

In our experiments, the generalization ability of the classification model is verified through two methods: 2-fold cross validation and leave one-out cross validation. In the method of 2-fold cross validation, the samples are divided into two parts. One is all the odd-numbered samples, another is all the even-numbered samples. When the odd-numbered samples are regarded as training samples, the even-numbered samples are used as testing samples, and vice versa. In the method of leave one-out cross validation, every sample has a chance as the testing sample while other samples are selected as the training samples.

By means of rough decision rules, the RRGELM is constructed by the method mentioned in Section 3.Then, it is trained by the algorithm mentioned in the section 4.3.4. In order to compare the generalization ability of RRGELM to other similar methods, the tests are compared with the original ELM algorithm Huang et al. (2006). In the ELM algorithm, the number of hidden neurons is defined to 30, and the activation function is selected as the sigmoid function.

To evaluate the performance of RRGELM and ELM, the success rate used in [28] is adopted for classification problems. Experimental results of RRGELM and ELM based on the method of 2-fold cross validation are shown in Table 6. As is shown in Table 6, the generalization ability of RRGELM has better generalization ability than ELM in three data sets. Only in the Soybean (small) data set, the generalization ability of RRGELM is inferior to ELM. This might occur because when the attributions are reduced from 35 to 4, some information may be lost in the reduced process of RRGELM.

Table 6. Success rate of three models based on 2-fold cross validation

Data Sets	RRGELM(%)	ELM(%)
Zoo	96	92
Balance Scale	92.95	86.96
Soybean(small)	86.96	94.87
Car Evaluation	79.75	74.19

The experimental results of RRGELM and ELM based on the method of leave one-out cross validation are shown in Table 7. From Table 7, we can gain the similar phenomenon that the performance of RRGELM is inferior to ELM only in the Soybean (small) data set, while in other three data sets RRGELM is superior to ELM. Furthermore, in two smaller date sets, Zoo and Soybean(small), the generalization ability of RRGELM using leave one-out cross validation is slight lower than that using 2-fold cross validation. Meanwhile, in two larger date sets, Balance Scale and Car Evaluation, the generalization ability of RRGELM using leave one-out cross validation is superior to that using 2-fold cross validation, especially in the Car Evaluation data set. The reasons for the above results ought to be the smaller number of the data set has a greater impact on the method of leave one-out cross validation.

Table 7. Success rate of three models based on leave one-out cross validation

Data Sets	RRGELM(%)	ELM(%)
Zoo	95.05	94.06
Balance Scale	93.75	90.24
Soybean(small)	85.11	95.74
Car Evaluation	91.38	83.28

6. Conclusions

In this paper, we have presented a novel framework for GNN, in which the hidden layer is composed of granular-neurons. Granular-neurons in the proposed model are built by rough decision rules extracted from the reduced the data set through the attribute reduction and value reduction algorithms in rough set theory. All those neurons consist of the rule matching layer of RRGELM. Each neuron of the rule matching layer represents a rough decision rule and the number of granular-neurons equals to the number of decision rules. The input linked weights and output linked weights of rule matching layer can be initialized by the confidence level and the coverage level, respectively. The output linked weights only need to be adjusted once through ELM algorithm during training. The proposed model realizes the seamless integration between rough set theory and neural networks, so it integrates the advantages of those two models. The experiments on several well-known benchmark data sets indicate that our model is suitable for classification problems. Our future works may focus on extending our model to the GNN models with multiple hidden layers, or applying the proposed model to those fields about the big data, as so on.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (No.2013QNA24), the Basic Research Program (Natural Science Foundation) of Jiangsu Province of China (No.BK2013093), and the National Natural Science Foundation (No.61379101).

References

Blake, C., Merz, C.J., 1998. {UCI} repository of machine learning databases .

Cao, Y., Chen, X.H., Wu, D.D., Mo, M., 2011. Early warning of enterprise decline in a life cycle using neural networks and rough set theory. Expert Systems with Applications 38, 6424–6429.

Chang, Y.Y., Tai, S.C., Lin, J.S., 2012. Segmentation of multispectral mr images through an annealed rough neural network. Neural Computing and Applications 21, 911–919.

Chen, M.C., Chen, L.S., Hsu, C.C., Zeng, W.R., 2008. An information granulation based data mining approach for classifying imbalanced data. Information Sciences 178, 3214–3227.

Cheng, J.H., Chen, H.P., Lin, Y.M., 2010. A hybrid forecast marketing timing model based on probabilistic neural network, rough set and c4. 5. Expert systems with Applications 37, 1814–1820.

Dick, S., Tappenden, A., Badke, C., Olarewaju, O., 2013. A granular neural network: Performance analysis and application to re-granulation. International Journal of Approximate Reasoning 54, 1149–1167.

Ding, S., Xu, X., Nie, R., 2014. Extreme learning machine and its applications. Neural Computing and Applications 25, 549–556.

Ganivada, A., Dutta, S., Pal, S.K., 2011. Fuzzy rough granular neural networks, fuzzy granules, and classification. Theoretical Computer Science 412, 5834–5853.

Hassan, Y.F., 2011. Rough sets for adapting wavelet neural networks as a new classifier system. Applied Intelligence 35, 260–268.

Huang, G.B., Zhou, H.M., Ding, X.j., Zhang, R., 2012. Extreme learning machine for regression and multiclass classification. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42, 513–529.

Huang, G.B., Zhu, Q.Y., Siew, C.K., 2006. Extreme learning machine: theory and applications. Neurocomputing 70, 489–501.

Leite, D., Costa, P., Gomide, F., 2010. Evolving granular neural network for semi-supervised data stream classification, in: Neural Networks (IJCNN), The 2010 International Joint Conference on, IEEE. pp. 1–8.

Leite, D., Costa, P., Gomide, F., 2012. Evolving granular neural network for fuzzy time series forecasting, in: Neural Networks (IJCNN), The 2012 International Joint Conference on, IEEE. pp. 1–8.

Leite, D., Costa, P., Gomide, F., 2013. Evolving granular neural networks from fuzzy data streams. Neural Networks 38, 1–16.

Lin, C., Lin, T., Chiu, S., 2013. Corporate performance forecasting using hybrid rough set theory. Neural Networks, and DEA 41, 359–365.

Marček, M., Marček, D., 2008. Approximation and prediction of wages based on granular neural network, in: Rough Sets and Knowledge Technology. Springer, pp. 556–563.

Mingli, S., Pedrycz, W., 2013. Granular neural networks: concepts and development schemes. IEEE transactions on neural networks and learning systems 24, 542–553.

Pal, S.K., Dasgupta, B., Mitra, P., 2004. Rough self organizing map. Applied Intelligence 21, 289–299.

Park, H.S., Pedrycz, W., Chung, Y.D., Oh, S.K., 2012. Modeling of the charging characteristic of linear-type superconducting power supply using granular-based radial basis function neural networks. Expert Systems with Applications 39, 1021–1039.

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

Stathakis, D., Vasilakos, A., 2006. Satellite image classification using granular neural networks. International journal of remote sensing 27, 3991–4003.

Syeda, M., Zhang, Y.Q., Pan, Y., 2002. Parallel granular neural networks for fast credit card fraud detection, in: Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on, IEEE. pp. 572–577.

Valdés, J.J., Romero, E., Barton, A.J., 2012. Data and knowledge visualization with virtual reality spaces, neural networks and rough sets: Application to cancer and geophysical prospecting data. Expert Systems with Applications 39, 13193–13201.

Vasilakos, A., Stathakis, D., 2005. Granular neural networks for land use classification. Soft Computing 9, 332–340.

Xu, X.Z., Ding, S.F., Zhu, H., 2012. Optimizing radial basis function neural network based on rough set and ap clustering algorithm. Zhejiang University (SCIENCE C) 13, 18.

Zadeh, L.A., 1997. Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy sets and systems 90, 111–127.

Zhang, D.B., Wang, Y.N., Huang, H.X., 2009. Rough neural network modeling based on fuzzy rough model and its application to texture classification. Neurocomputing 72, 2433–2443.

Zhang, Y.Q., Akkaladevi, S., Vachtsevanos, G., Lin, T.Y., 2002. Granular neural web agents for stock prediction. Soft Computing 6, 406–413.

Zhang, Y.Q., Fraser, M.D., Gagliano, R.A., Kandel, A., 2000. Granular neural networks for numerical-linguistic data fusion and knowledge discovery. Neural Networks, IEEE Transactions on 11, 658–667.

Zhang, Y.Q., Jin, B., Tang, Y.C., 2008. Granular neural networks with evolutionary interval learning. Fuzzy Systems, IEEE Transactions on 16, 309–319.

Zhao, J.Y., Zhang, Z.L., 2011. Fuzzy rough neural network and its application to feature selection, in: Advanced Computational Intelligence (IWACI), 2011 Fourth International Workshop on, IEEE. pp. 684–687.