Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 9958427, 16 pages
https://doi.org/10.1155/2021/9958427

Research Article

Boosted Fuzzy Granular Regression Trees

Wei Li,[1] **Youmeng Luo**,[1] **Chao Tang**,[2] **Kaiqiang Zhang**,[1] **and Xiaoyu Ma**[1]

[1]*School of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China*
[2]*School of Artificial Intelligence and Big Data, Hefei University, Hefei 230601, China*

Correspondence should be addressed to Wei Li; drweili@hotmail.com

Received 16 March 2021; Revised 7 June 2021; Accepted 9 July 2021; Published 22 July 2021

Academic Editor: Petr Hájek

The regression problem is a valued problem in the domain of machine learning, and it has been widely employed in many fields such as meteorology, transportation, and material. Granular computing (GrC) is a good approach of exploring human intelligent information processing, which has the superiority of knowledge discovery. Ensemble learning is easy to execute parallelly. Based on granular computing and ensemble learning, we convert the regression problem into granular space equivalently to solve and proposed boosted fuzzy granular regression trees (BFGRT) to predict a test instance. The thought of BFGRT is as follows. First, a clustering algorithm with automatic optimization of clustering centers is presented. Next, in terms of the clustering algorithm, we employ MapReduce to parallelly implement fuzzy granulation of the data. Then, we design new operators and metrics of fuzzy granules to build fuzzy granular rule base. Finally, a fuzzy granular regression tree (FGRT) in the fuzzy granular space is presented. In the light of these, BFGRT can be designed by parallelly combing multiple FGRTs via random sampling attributes and MapReduce. Theory and experiments show that BFGRT is accurate, efficient, and robust.

1. Introduction

Learning ability is the basic feature of human intelligence. Prediction is the ability of humans to judge the future based on learning, and it is also a concrete manifestation of human learning ability. Prediction has two concrete forms, regression and classification, which is also one of the substance problems in machine learning, data mining, and statistics. How to train a learner on the basis of existing data is the primary research purpose of the regression problem. It can help people discover the laws of development and change of things from massive historical data, so as to make scientific and quantitative predictions about the future. In the classification problem, the target output is to take values in a finite discrete space, and these values can be either ordered or disordered. In regression problems, the range of output variables is ordered and continuous.

Research on regression problems based on ensemble learning is a hot topic in machine learning research in recent years and has received extensive attention. Nevertheless, the application of ensemble learning to regression problems is still unsatisfactory and needs further research. The regression problem to be solved in reality often comes from a very complex social economic system, and various intricate internal and external factors have linear or nonlinear effects on it. Some are inherent factors, and some are accidental factors. A single learner can only learn for a certain type of data, and it is difficult to get satisfied learning results. Especially in the big data environment, traditional regression learning algorithms are not able to meet the learning requirements of massive complex data in terms of predictive performance and scalability. Although a great progress has been made in the related theoretical research and technology of machine learning, how to continuously improve the generalization ability and learning efficiency of learners is still an important issue and continuous pursuit of machine learning research.

GrC is an effective method for simulating problem-solving thinking of human and processing analysis tasks of big data. It abstracts and divides complex problems into several simpler ones, which helps to better analyze and solve problems. Combining granular computing with ensemble learning to solve regression problems is a good idea.

As the first of the four main research directions in machine learning, ensemble learning learns several different single learners on the training dataset and then combines the respective prediction results as the final output one. Therefore, in most cases, it can perform better than a single learner on generalization and stability [1]. The weak learner can be upgraded to a strong learner is one of the main theoretical foundations of ensemble learning. Kearns and Valiant gave the concepts of weak learning and strong learning from the perspective of classification problems [2]. Avnimelech and Intrator introduced the above concepts into the regression problem and gave a proof of the equivalence of the strong and weak learning of the one [3]. Another major theoretical basis for ensemble learning is the "No free lunch" theory proposed by Wolpert [4]. The implementation method of ensemble learning has received extensive attention from researchers and has achieved some research results. These results can be summarized into two categories: one is the direct strategy and the other is the overproduce and choose approach. Liu and his colleagues proposed an ensemble learning method via negative correlation learning [5]. Fang et al. proposed a selective boosting ensemble learning algorithm [6]. Breiman obtained multiple different training datasets by repeatedly sampling the original sample dataset [7]. Schapire proposed the boosting method, whose main idea is to continuously strengthen the learning of "difficult samples" in the iterative learning process [8]. Ho presented the random subspace method that uses different subsets of the feature space to train and generate multiple learners [9]. This method is different from bootstrap sampling, boosting, and cross-validation approaches and emphasizes the differences between different feature subsets. Breiman designed the output smearing method for regression problems. The primary thought is to inject Gaussian noise into the output variable [10]. The same method is also used to manipulate input variables [11]. Gheyas and Simith presented a dynamic weighted combination method, which is to dynamically adjust the corresponding weights through the predictive performance of the individual learner [12]. The research of ensemble learning on regression problems started late, and there are relatively few research results in applications, such as power load forecasting [13, 14].

Granular computing is a very popular research direction in the field of computational intelligence in the past few decades. The core task of GrC is to construct, represent, and process information granules. Information granule is the foundational element of GrC. It is a set of some elements gathered together according to indistinguishability, similarity, or functionality. As a key component of knowledge representation and processing, information granules always appear with information granulation, and information granulation occurs in the process of abstracting data or inducing knowledge from information. Information (data) forms information granules through information granulation. The representation forms of information granules often include interval [15], fuzzy set [16], and rough set [17]. The purpose of information granulation is to separate complex problems into several simpler problems. This way can make us capture the details of the problem. Information granulation and information granules are almost infiltrated in various human cognition, decision-making, and reasoning processes and are closely related to information granularity. For example, in daily life and work, people usually use different time intervals such as day, month, and year to granulate time to obtain information granules of different sizes, and the size of the formed information granules implies the level of information granularity used in granulation. Through the abstraction of the problem, the "finer" and "more special" information granules are transformed into "more coarse" and "more general" information granules and "more coarse" and "more general" information granules can be refined into "finer" and "more special" information granules. GrC helps people analyze and watch the similar problem from extremely different granularities through the transformation between information granules and finally find the most suitable level of analysis and problem solving. The basic composition of GrC mainly includes three parts: granule, granular layer, and granular structure [18]. Granules are the foundational elements of GrC models [19]. The granular layer is an abstract description of the problem space according to a certain granulation criterion [20]. Granular structure is an abstract description of all granular layers. One granulation criterion corresponds to one granular layer, and different granulation rules correspond to multiple granular layers. It shows that people observe, comprehend, and solve problems from various views. All the interconnections between the granular layers form an interactive structure called the granular structure [21]. There are two basic problems in GrC: granulation and calculation based on granulation [22]. Granulation is the first step of GrC to solve the problem, and it is a process of constructing a problem-solving knowledge space. The human brain's cognitive process of external things is a typical granulation, that is, from the overall and rough cognition, through the continuous processing and refinement of information, it finally forms a partial and detailed analysis and reasoning. The granulation process mainly involves granulation criteria, granulation methods, granule descriptions, and other issues [23]. Granularization-based computing refers to solving the original problem or logical reasoning with granules as the object of operation and the granularity level as the carrier. It is mainly divided into two types: mutual reasoning between granules on the same granular layer and conversion between granules on different granular layers. After years of study and growth, many models of GrC and their extended models have been proposed. The most representative ones are three theoretical models: the fuzzy set model, rough set model, and quotient space model.

1.1. Fuzzy Set Model. Zadeh presented the fuzzy set theory in 1965. He pointed out that a single element always belongs to a set to some extent and may also belong to several sets to a different degree [16]. Under the fuzzy set theory system, he presented a GrC model on the basis of word computing. The core idea of this method is to use words for calculation and reasoning instead of numbers, to achieve fuzzy reasoning,

and control of complex information systems, which is in line with human thinking. In addition, Wang employed natural language knowledge to establish a linguistic dynamics system based on word computing and designed a computational theoretical framework for a linguistic dynamic system based on word computing by fusing concepts and schemes in multiple fields [24].

1.2. Rough Set Model. The degree to an object belonging to a certain set varies with the granularity of the attribute. In order to better characterize the ambiguity of set boundaries, Pawlak proposed the rough set theory in the 1980s. Its essential idea is to adopt indistinguishable relations (equivalence relations) to establish a division of the universe of equivalence classes to establish an approximate space. In the approximate space, upper approximate set and lower approximate set are employed to approximate a set with fuzzy boundaries [17]. The classic rough set theory is mainly aimed at a thorough information system where all feature values of the object processed are understood. In order to make the rough set theory be suitable for handling of uncomplete information systems, there are currently two main ways: one is to fill incomplete data and the other is to expand the classic rough set model. Kryszkiewicz proposed an extended rough set model via tolerance relations [25]. Stefanowski and his teamwork presented an extended rough set model on the basis of asymmetric similarity relations and an extended rough set model by quantitative tolerance relations [26]. Wang analyzed the shortcomings of the previous two expansion models and designed a rough set model based on the restricted tolerance relationship. He found that the tolerance relationship and the asymmetric similarity relationship are the two extremes of the indistinguishable relationship expansion, that is, the condition of the tolerance relationship is too loose, the condition of the asymmetric similar relationship is too tight, and the limit tolerance relationship is between them [27]. Pawlak employed the idea that elements in the equivalent category have the same membership function and explored the structure and granularity of knowledge granules [28]. Polkowski and Skowron adopted the rough mereology method, neural network technology, and the idea of knowledge granulation to design a rough neural computing model, which combines the division block of the rough set and the neural network to form an efficient neural computing method [29]. Peters and his colleagues employed the indistinguishable relationship to divide the real number into multiple subintervals and divided a whole area into several grid units, and each unit was regarded as a granule, and proposed metric is between two information granules on the adjacent relationship and the containment relationship, respectively [30].

1.3. Quotient Space Theory Model. B. Zhang and L. Zhang presented the theory of quotient space when studying problem solving. They said that "the recognized feature of quasi-intelligence is that human can analyze and watch the same problem from different granularity" [31]. The quotient space theory established a formal system of quotient structure and proposed a series of theories and approaches to solve problems in the fields of heuristic search, information fusion, reasoning, and path planning. There were some related research and applications [32, 33].

In addition, many new models have been proposed, such as granular matrices for reduction and evaluation [34], three-way decision model [35–37], and ensemble learning for big data based on MapReduce [38–44]. In this study, we adopt parallel granulation and ensemble learning based on MapReduce to solve the regression problem from granular computing angle and enhance the performance of regression and efficiency of granulation.

2. Contributions

In this study, a regression problem is equivalently transformed into the fuzzy granular space solution, and BFGRT are constructed from angle of GrC and ensemble learning to solve the regression problem. The main contributions are as follows.

(i) First, an adaptive clustering algorithm is proposed, which can adaptively find the optimal cluster centers. It is a global optimization algorithm that solves the problem that classic clustering algorithms rely heavily on the initial cluster centers and are easy to fall into local optimal solution.

(ii) Second, parallel fuzzy granulation of data based on the above clustering algorithm combined with MapReduce is presented, which solves the problem of high complexity of traditional granulation and enhances granulation efficiency

(iii) Third, we define fuzzy granules and related metric operators, design a loss function, construct an individual fuzzy granular regression tree in the granular space by optimizing the loss function, and then parallelly integrate multiple fuzzy granular regression trees built by different attributes into a stronger learner based on MapReduce to accurately solve the regression problem

3. The Regression Problem

The regression problem is divided into two processes: learning and prediction. Suppose that $S = (T, A, Y, h, V)$ be a regression system, where $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is the set of instances $(x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, $A = \{a_1, a_2, \ldots, a_m\}$ is the attribute set, $V = \cup_{h \in H} V_a$ is the set of range, V_a is range of the attribute a, and h: $X \times A \longrightarrow V$ is an information function (it assigns a value to each instance on each attribute, namely, $\forall h \in H$, $x \in X, h(x, a) \in V_a$). Let $X = \{x_1, x_2, \ldots, x_n\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$, where y_i corresponds to the output of x_i $(i = 1, 2, \ldots, n)$. The learning system constructs a model $Y = f(X)$ based on the instance set. For the test instance x_{n+1}, the learning system can predict the corresponding output y_{n+1} by $Y = f(X)$.

4. The Primary Algorithm

In order to find the solution of the problem mentioned above, the algorithm will be presented through three stages. First, cluster the data. The purpose is to prepare for the parallel fuzzy granulation of the data. In this process, we design a novel clustering algorithm that adaptively optimizes the cluster center. According to the algorithm, cluster centers of instances are automatically calculated instead of giving the number of clusters in advance. Next, these cluster centers can be used as reference objects independent of the data to granulate data parallelly. Finally, we transform this instance regression problem into a fuzzy granular regression problem in the granular space. In the fuzzy granular space, we design related operators and loss functions, construct multiple weak fuzzy granular regression trees by optimizing the loss function, and then integrate these weak fuzzy granular regression trees into a strong learner to predict the regression value. The process is shown in Figure 1.

4.1. Clustering Algorithm with Automatic Optimization of Cluster Centers. Classic clustering algorithms need to specify the number of cluster centers in advance to obtain clustering results. The methods depend heavily on the above parameter. If the number of cluster centers is not suitable, it will fall into a local minimum solution. An adaptive clustering algorithm that adaptively selects the number of cluster centers is presented, which is a global optimization algorithm. The principle is as follows. As well-known, if the standard deviation δ of cluster centers is larger and the standard deviation σ_k within clusters is smaller, then the performance is better. Therefore, a loss function $L(\sigma, \delta, K) = \log(\sum_{k=1}^{K} \sigma_k^2) - \log \delta^2$ is designed as the evaluation criterion, where σ_k denotes the standard deviation of the k^{th} cluster and K represents the number of cluster centers. The aim is to decrease the loss function value by adjusting cluster centers until the maximum iteration is achieved or the loss function value hardly changes. In each iteration, a set of cluster centers corresponding to loss function value can be obtained and be integrated as an evaluated set. In each iteration, the ratio of the farthest distance from the remaining instance points to cluster centers and the sum of the farthest distances from all instance points to each cluster center is the probability selected as the next cluster center. When the termination condition is achieved, find a set of cluster centers corresponding to the minimum cost function from this evaluation set, which is what is required.

Step 1: remove the instances missing some attribute values

Step 2: normalize instances

Step 3: initialize parameters, such as maximum iteration I, evaluated set EV = ϕ (contains cluster centers and loss function value), and current iteration $i = 1$

Step 4: initialize current cluster center set $C_i = \phi$ and randomly select one instance point as the cluster center $c_1 = x_i$, $k = 1$

Step 5: calculate the farthest distance between the remaining instance points and all cluster centers and let $d(x_j) = \max_{1 \le k \le i} \{\text{dis}(c_k, x_j)\}$, and the probability of the instance selected as the next cluster center is $p(x_j) = (d(x_j)^2 / \sum_{i=1}^{n} d(x_i)^2)$, $j = 1, 2, \ldots, n$.

Step 6: if x_j is selected as a cluster center, then

$$\begin{aligned} c_k &= x_j, \\ C_i &= C_i \cup \{c_k\}, \\ k &= k + 1. \end{aligned} \tag{1}$$

Step 7: if $k < i + 1$, go Step 5. Otherwise, go Step 8.

Step 8: calculate loss function $L_i(\delta, \sigma, k)$ and update the evaluated set EV = EV $\cup \{(C_i, L(\delta, \sigma, i))\}$

Step 9: update iteration $i = i + 1$

Step 10: if $i > I$ or $\forall N \in (1, i - j), \|L(\delta, \sigma, j) - L((\delta, \sigma, j + N))\|^2 < \epsilon$, go Step 11 (here, $j + N < i$, ϵ is a small positive number). Otherwise, go Step 4.

Step 11: in the evaluated set EV, select the cluster centers according to

$$C^* = \arg\min_{1 \le j \le i} L(\delta, \sigma, j), \tag{2}$$

Namely, $C^* = \{c_1, c_2, \ldots, c_{|C^*|}\}$ and the optimization number of cluster centers $K = |C^*|$ ($|\cdot|$ expresses the number of elements of the set).

Step 12: end.

Algorithm 1 shows the pseudocode of this principle.

4.2. Parallel Fuzzy Granulation. In granulation, serial granulation is adopted in most methods. To enhance efficiency, we propose parallel fuzzy granulation. First, cluster centers can be obtained by the approach mentioned above. Then, instances are divided into a set of instance subsets, which are fuzzy granulated by cluster centers. This process is executed parallelly by MapReduce. According to Algorithm 1, the cluster center set $C = \{c_1, c_2, \ldots, c_K\}$ can be obtained.

For $\forall x_i \in X$, $\forall a_j \in A$, and $\forall c_k \in C$, the distance between x_i and c_k on the attribute a_j can be written as follows:

$$s(x_i, a_j, c_k) = \frac{1}{1 + \exp\left(-\left|h(x_i, a_j) - h(c_k, a_j)\right|\right)}, \tag{3}$$

where $0 \le h(x_i, a_j) \le 1$, $0 \le h(c_k, a_j) \le 1$, and $0 \le s(x_i, a_j, c_k) \le 1$ are established. A fuzzy granule induced by the instance x_i and the cluster center c_j can be written as follows:

$$\overline{q}(x_i, a_j) = \frac{s(x_i, a_j, c_1)}{c_1} + \frac{s(x_i, a_j, c_2)}{c_2} + \cdots + \frac{s(x_i, a_j, c_K)}{c_K}. \tag{4}$$

For simplicity, the fuzzy granule uses also the following equation to denote.

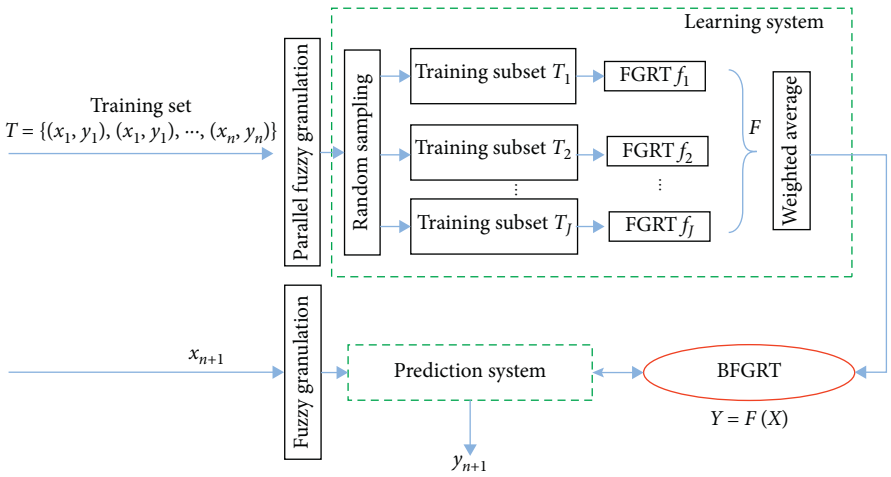

Figure 1: Overview of BFGRT.

Input: instance set X, maximum iteration I, threshold value ϵ, N
Output: optimization cluster center set C^*

(1) Remove instances missing some attribute values.
(2) Normalize each attribute value into $[0, 1]$.
(3) EV = ϕ\\Let evaluated set be an empty set.
(4) $i = 1$\\Initialize current iteration.
(5) WHILE $(i < I + 1)$ OR $(\|L(\delta, \sigma, i) - L(\delta, \sigma, i + N)\|^2 > \epsilon)$
(6) $C_i = \phi$. Initialize the current cluster center set.
(7) $k = 1$. Initialize the number of cluster center.
(8) c_k = RandomSelect$(X, 1)$\\Random choose 1 instance point as cluster center.
(9) $C_i = C_i \cup \{c_k\}$
(10) $k = k + 1$
(11) WHILE $k < i + 1$
(12) $\forall j \in |X|, d(x_j) = \max_{1 \le k \le n}\{\text{dis}(c_k, x_j)\}$
(13) $p(x_j) = (d(x_j)^2 / \sum_{t=1}^{n} d(c_k, x_t)^2)$.\\The probability of x_j that is selected as next cluster center
(14) p = GenProb(); \\Random generate a probability.
(15) IF $p(x_j) > p$ THEN $\{c_k = x_j; C_i = C_i \cup \{x_j\}; k = k + 1;\}$
(16) END WHILE \\$k < i + 1$
(17) EV = EV $\cup \{(C_i, L(\delta, \sigma, i))\}$. Calculate the loss function value and cluster center in this iteration and update the evaluated set.
(18) $i = i + 1$.\\Update the current iteration.
(19) END WHILE \\$i < I + 1$
(20) $C^* = \arg\max_{1 \le j \le i}\{L(\delta, \sigma, j)\}$. In the evaluated set EV, choose the cluster center set with minimum loss function value.
Return the optimization cluster centers and their number.
(21) $C^* = \{c_1, c_2, \ldots, c_{|C|^*}\}$, $k^* = |C^*|$ ($|\cdot|$ represents the number of elements of the set).

Algorithm 1: Clustering algorithm with automatic optimization of cluster centers.

$$\overline{q}(x_i, a_j) = \sum_{t=1}^{K} \frac{s(x_i, a_j, c_t)}{c_t}. \tag{5}$$

Here, symbol "–" is a separator and symbol "+" is an union operation, that is, fuzzy granule $\overline{q}(x_i, a_j)$ denotes the distance set between instance x_i and cluster centers. Its cardinal can be written as

$$\left|\overline{q}(x_i, a_j)\right| = \sum_{t=1}^{K} s(x_i, a_j, c_t). \tag{6}$$

Four operators of fuzzy granule can be designed. For $\forall e, f \in R$, operator $\cup$ and operator $\cap$ can be written as follows:

$$\begin{aligned} e \cup f &= \frac{e + f - ef - (1 - \nu)ef}{\nu + (1 - \nu)(1 - ef)}, \\ e \cap f &= \frac{ef}{\nu + (1 - \nu)(e + f - ef)}, \end{aligned} \tag{7}$$

where $\nu \in [0, \infty]$ is a parameter. For $\forall x, x' \in X$, and $\forall a \in A$, the operators between the fuzzy granule formed by x and the one induced by x' are written as follows:

$$\begin{aligned}
\overline{q}(x,a) \cup \overline{q}(x',a) &= \sum_{t=1}^{K} \frac{s(x,a,c_t) \cup s(x',a,c_t)}{c_t}, \\
\overline{q}(x,a) \cap \overline{q}(x',a) &= \sum_{t=1}^{K} \frac{s(x,a,c_t) \cap s(x',a,c_t)}{c_t}, \\
\overline{q}(x,a) - \overline{q}(x',a) &= \sum_{t=1}^{K} \frac{s(x,a,c_t) - s(x',a,c_t)}{c_t}, \\
\overline{q}(x,a) \oplus \overline{q}(x',a) &= \overline{q}(x,a) \cup \overline{q}(x',a) - \overline{q}(x,a) \cap \overline{q}(x',a).
\end{aligned} \tag{8}$$

For $\forall U \subseteq A$ and $U = \{a_1, a_2, \ldots, a_{|U|}\}$ $(|U| \le |A|)$, fuzzy granular vector induced by x on the attribute set U can be written as

$$\begin{aligned}
&\overline{Q}(x,U) \\
&= \frac{\overline{Q}(x,a_1)}{a_1} + \frac{\overline{Q}(x,a_2)}{a_2} + \cdots + \frac{\overline{Q}(x,a_{|U|})}{a_{|U|}} \\
&= \sum_{t=1}^{|U|} \frac{\overline{q}(x,r_t)}{a_t},
\end{aligned} \tag{9}$$

where symbol "+" denotes an union operation and symbol "−" represents a separator. Its cardinal can be obtained as follows:

$$|\overline{Q}(x,U)| = \sum_{t=1}^{|U|} |\overline{q}(x,a_t)|. \tag{10}$$

Operators of fuzzy granular vector are written as follows:

$$\begin{aligned}
\overline{Q}(x,U) \cup \overline{Q}(x',U) &= \sum_{t=1}^{|U|} \frac{\overline{q}(x,a_t) \cup \overline{q}(x',a_t)}{a_t}, \\
\overline{Q}(x,U) \cap \overline{Q}(x',U) &= \sum_{t=1}^{|U|} \frac{\overline{q}(x,a_t) \cap \overline{q}(x',a_t)}{a_t}, \\
\overline{Q}(x,U) - \overline{Q}(x',U) &= \sum_{t=1}^{|U|} \frac{\overline{q}(x,a_t) - \overline{q}(x',a_t)}{a_t}, \\
\overline{Q}(x,U) \oplus \overline{Q}(x',U) &= \sum_{t=1}^{|U|} \frac{\overline{q}(x,a_t) \oplus \overline{q}(x',a_t)}{a_t}.
\end{aligned} \tag{11}$$

According to the definitions, the distance between the two fuzzy granular vectors is given by

$$\begin{aligned}
&s(\overline{Q}(x,U), \overline{Q}(x',U)) \\
&= \frac{1}{|U|^* |C|} \sum_{a \in U} \frac{|\overline{Q}(x,a) \oplus \overline{Q}(x',a)|}{|\overline{Q}(x,r) \cup \overline{Q}(x',r)|}.
\end{aligned} \tag{12}$$

From the above fuzzy granulation, it can be seen that the fuzzy granules are obtained by calculating instances and cluster centers using the fuzzy operators, and fuzzy granular space consists of these fuzzy granules.

Theorem 1. *For $\forall x, x' \in X$, $\forall U \subseteq A$, and $\forall a \in U$, the distance between fuzzy granular vectors satisfies*

$$0 \le s(\overline{Q}(x,U), \overline{Q}(x',U)) \le 1. \tag{13}$$

Proof. According to the definition of fuzzy granule, we have $\overline{q}(x,a) = \sum_{t=1}^{K} s(x,a,c_t)/c_t$ and $\overline{q}(x',a) = \sum_{t=1}^{K} s(x',a,c_t)/c_t$. Also, from equation (3), we get $0 \le s(x,a,c_t) \le 1$ and $0 \le s(x',a,c_t) \le 1$. Because of $|\overline{q}(x,a)| = \sum_{c_t \in C} s(x,a,c_t)$ and $|\overline{q}(x',a)| = \sum_{c_t \in C} s(x,a',c_t)$, the inequalities $0 \le |\overline{q}(x,a)| \le |C|$ and $0 \le |\overline{q}(x',a)| \le |C|$ can be obtained. Equation (9) shows $Q(x,U) = \sum_{t=1}^{|U|} \overline{q}(x,r_t)/r_t$ and $|Q(x',U)| = |\sum_{t=1}^{|U|} |q(x',r_t)||$. We can further obtain $0 \le |\overline{Q}(x,U)| \le |U|^* |C|$ and $0 \le |\overline{Q}(x',U)| \le |U|^* |C|$. From $\overline{q}(x,a) \oplus \overline{q}(x',a) = q(x_i,r) \cup q(x_j,r) - q(x_i,r) \cap q(x,a)$, we get

$$0 \le \sum_{r \in U} \frac{|q(x_i,r) \oplus q(x_j,r)|}{|q(x,r) \cup q(x',r)|} \le |U|^* |C|. \tag{14}$$

We divide both sides of the formula by $|U|^* |C|$ to achieve $0 \le (1/|U|^* |C|) \sum_{a \in U} |\overline{q}(x,a) \oplus \overline{q}(x\imath,a)| / |\overline{q}(x,a) \cup \overline{q}(x',a)| \le 1$, that is, we prove

$$0 \le s(\overline{Q}(x,U), \overline{Q}(x\imath,U)) \le 1. \tag{15}$$

□

Theorem 2. *For $\forall x \in X$, the attribute subsets U and V satisfy $U \subseteq V \subseteq A$. Let $\overline{Q}(x,V)$ and $\overline{Q}(x,U)$ be fuzzy granular vectors of x on V and U, respectively. Then, $|\overline{Q}(x,U)| \le |\overline{Q}(x,V)|$ has been proven.*

Proof. For $\forall a_t \in U$, equation (9) shows that $\overline{Q}(x,U) = \sum_{t=1}^{|U|} \overline{q}(x,a_t)/a_t$. Thanks to $U \subseteq V$, we have that for $\forall a_t \in V$, $\overline{Q}(x,V) = \sum_{t=1}^{|V|} \overline{q}(x,a_t)/a_t$. Because of $U \subseteq V \subseteq A$, for $a \in U$, we get $a \in V$ and $|U| \le |V|$, that is, if $\overline{q}(x,a) \in \overline{Q}(x,U)$, we can obtain $\overline{q}(x,a) \in Q(x,V)$. In sum, $|\overline{Q}(x,U)| \le |\overline{Q}(x,V)|$ is proved. □

Now, we take a case to describe the fuzzy granulation process.

Example 1. As illustrated in Table 1, given an instance set $X = \{x_1, x_2, x_3, x_4\}$, an attribute set $A = \{a_1, a_2, a_3\}$, regression value set $Y = \{y_1, y_2, y_3, y_4\}$, cluster center set $C = \{c_1, c_2\}$, and parameter $\nu = 0.5$, the fuzzy granulation is as follows.

We take instance x_1 as an example. The distances between x_1 and the cluster centers c_1 and c_2 on the attributes a_1, a_2, and a_3 are given as follows:

$$s(x_1, a_1, c_1) = 1/1 + \exp(-|0.20 - 0.25|) = 0.5125$$
$$s(x_1, a_1, c_2) = 1/1 + \exp(-|0.20 - 0.15|) = 0.5125$$
$$s(x_1, a_2, c_1) = 1/1 + \exp(-|0.30 - 0.35|) = 0.5125$$
$$s(x_1, a_2, c_2) = 1/1 + \exp(-|0.30 - 0.45|) = 0.5374$$
$$s(x_1, a_3, c_1) = 1/1 + \exp(-|0.10 - 0.30|) = 0.5498$$
$$s(x_1, a_3, c_2) = 1/1 + \exp(-|0.10 - 0.60|) = 0.6225$$

Equation (5) shows that fuzzy granules of x_1 on a_1, a_2, and a_3 are calculated as follows:

$$\overline{q}(x_1, a_1) = (0.5125/c_1) + (0.5125/c_2)$$
$$\overline{q}(x_1, a_2) = (0.5125/c_1) + (0.5374/c_2)$$
$$\overline{q}(x_1, a_3) = (0.5498/c_1) + (0.6225/c_2)$$

In the same way, fuzzy granules of x_2 on A are obtained as follows:

$$\overline{q}(x_2, a_1) = (0.5125/c_1) + (0.5374/c_2)$$
$$\overline{q}(x_2, a_2) = (0.5374/c_1) + (0.5622/c_2)$$
$$\overline{q}(x_2, a_3) = (0.5250/c_1) + (0.5987/c_2)$$

According to $\overline{q}(x_1, a_1) \cup \overline{q}(x_2, a_1) = \sum_{j=1}^{K} (e + f - ef - (1-\nu)ef/\nu + (1-\nu)(1-ef))/c_j$, here $\nu = 0.5$. When $j = 1$, suppose that $e = d(x_1, a_1, c_1)$ and $f = d(x_2, a_1, c_1)$. When $j = 2$, suppose that $e = d(x_1, a_1, c_2)$ and $f = d(x_2, a_1, c_2)$), we have

$$\begin{aligned}
&\overline{q}(x_1, a_1) \cup \overline{q}(x_2, a_1) \\
&= \sum_{j=1}^{2} \frac{(e + f - ef - (1-\nu)ef/\nu + (1-\nu)(1-ef))}{c_j} \\
&= \frac{(0.5125 + 0.5125 - 0.5125 * 0.5125 - (1-0.5) * 0.5125 * 0.5125/0.5 + (1-0.5)(1 - 0.5125 * 0.5125))}{c_1} \\
&\quad + \frac{(0.5125 + 0.5374 - 0.5125 * 0.5374 - (1-0.5) * 0.5125 * 0.5374/0.5 + (1-0.5)(1 - 0.5125 * 0.5374))}{c_2} \\
&= \frac{0.7164}{c_1} + \frac{0.7385}{c_2}, \\
&\overline{q}(x_1, a_1) \cap \overline{q}(x_2, a_1) \\
&= \sum_{j=1}^{2} \frac{(ef/\nu + (1-\nu)(e + f - ef))}{c_j} \\
&= \frac{(0.5125 * 0.5125/0.5 + (1-0.5)(0.5125 + 0.5125 - 0.5125 * 0.5125))}{c_1} \\
&\quad + \frac{(0.5125 * 0.5374/0.5 + (1-0.5)(0.5125 + 0.5374 - 0.5125 * 0.5374))}{c_2} \\
&= \frac{0.2981}{c_1} + \frac{0.3024}{c_2}, \\
&\overline{q}(x_1, a_1) \oplus \overline{q}(x_2, a_1) \\
&= \overline{q}(x_1, a_1) \cup \overline{q}(x_2, a_1) - \overline{q}(x_1, a_1) \cap \overline{q}(x_2, a_1) \\
&= \frac{0.7164 - 0.2981}{c_1} + \frac{0.7385 - 0.3024}{c_2} = \frac{0.4183}{c_1} + \frac{0.4361}{c_2}, \\
&\left|\overline{q}(x_1, a_1) \oplus \overline{q}(x_2, a_1)\right| \\
&= \left|\overline{q}(x_1, a_1) \cup \overline{q}(x_2, a_1) - \overline{q}(x_1, a_1) \cap \overline{q}(x_2, a_1)\right| = \left|\frac{04183}{c_1} + \frac{0.4361}{c_2}\right| = 0.8544.
\end{aligned} \tag{16}$$

Similarly, we can get

$$\left|\overline{q}(x_1,a_2)\oplus\overline{q}(x_2,a_2)\right| = \left|\frac{0.4280}{c_1}+\frac{0.4253}{c_2}\right| = 0.8533,$$
$$\left|\overline{q}(x_1,a_3)\oplus\overline{q}(x_2,a_3)\right| = \left|\frac{0.4269}{c_1}+\frac{0.4106}{c_2}\right| = 0.8375. \quad (17)$$

Thus, we can obtain the distance between fuzzy granular vectors of x_1 and x_2 on A with $\nu = 0.5$ by

$$s(\overline{Q}(x_1,A),\overline{Q}(x_2,A)) = \frac{1}{|A|^*|C|}\sum_{a\in A}\frac{\left|\overline{q}(x_1,a)\oplus\overline{q}(x_2,a)\right|}{\left|\overline{q}(x_1,a)\cap\overline{q}(x_2,a)\right|} = \frac{0.8544+0.8533+0.8375}{3*2}\approx 0.42. \quad (18)$$

4.3. Fuzzy Granular Regression Tree. After the above fuzzy granulation, the data can be transformed into fuzzy granules. In fuzzy granular space, we give the following definition.

Definition 1. Suppose that $S = \{T, A, C, Y\}$ is a regression system, where $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is a instance set, x_i is a input variable, and y_i is output variable corresponding to x_i. Let $X = \{x_i | i = 1, 2, \ldots, n\}$, and $Y = \{y_i | i = 1, 2, \ldots, n\}$. $C = \{c_i | i = 1, 2, \ldots, K\}$ is a cluster center set and $A = \{a_i | i = 1, 2, \ldots, m\}$ an attribute set. For $\forall a_j \in A, x_i \in X$, we can obtain fuzzy granule $\overline{q}(x_i, a_j)$ via fuzzy granulation (the following is abbreviated as $\overline{q}$). Fuzzy granules and operators can create new fuzzy granules, such as $\overline{g} = \overline{w}\cap\overline{q}\cup\overline{h}$. Repeating this process can expand a fuzzy granular space $\overline{\mathbb{G}}$ on T. According to training data, a fuzzy granular rule base can be generated as

$$R = \{r_i \mid r_i = <\overline{Q}(x_i, A), \quad y_i>, x_i \in X, y_i \in Y\}. \quad (19)$$

A fuzzy granular regression tree corresponds to a division of the fuzzy granular space and output value on the divided unit.

Suppose that the input space has been splitted into d units $D_1, D_2, \ldots, D_d$, and there is a fixed output value z_i on each unit D_i. Thus, the fuzzy granular regression tree can be expressed as

$$f(\overline{Q}) = \sum_{i=1}^{d} z_i I(\overline{Q}\in D_i). \quad (20)$$

Here, $\overline{Q}$ is a fuzzy granular vector and I is an indicative function, which can be written as

$$I = \begin{cases} 1, & \text{if } \overline{Q}\in D_i, \\ 0, & \text{otherwise}, \end{cases} \qquad z_i = \frac{1}{\|D_i\|}\sum_{\overline{Q}_j\in D_i} y_j, \quad (21)$$

Table 1: Fuzzy granulation and metric.

$X\|C\|A\|Y\|\nu$	a_1	a_2	a_3	y	ν
x_1	0.20	0.30	0.10	0.30	0.50
x_2	0.30	0.20	0.20	0.20	0.50
x_3	0.40	0.20	0.30	0.50	0.50
x_4	0.10	0.10	0.40	0.40	0.50
c_1	0.25	0.35	0.30	—	—
c_2	0.15	0.45	0.60	—	—

where $\|D_i\|$ denotes the number of the fuzzy granular vector. The question becomes how to divide the fuzzy granular space.

Here, we use a heuristic method to select feature a_j as the segmentation variable and the cardinal of fuzzy granule $\overline{q}(x_s, a_j)$ as the optimal segmentation point. Two areas are defined by

$$D_1(j,s) = \left\{\overline{Q}(x,A)\|\overline{q}(x,a_j)| \le \left|\overline{q}(x_s,a_j)\right|\right\},$$
$$D_2(j,s) = \left\{\overline{Q}(x,A)\|\overline{q}(x,a_j)| > \left|\overline{q}(x_s,a_j)\right|\right\}. \quad (22)$$

Then, find the optimal segmentation variable a_j and point $|\overline{q}(x_s, a_j)|$. Specifically, solve the loss function

$$\min_{j,s}\left\{\min_{z_1}\sum_{\overline{Q}_i\in D_1(j,s)}(y_i-z_1)^2 + \min_{z_2}\sum_{\overline{Q}_i\in D_2(j,s)}(y_i-z_2)^2\right\}. \quad (23)$$

For input variable a_j, the optimal segmentation point s can be calculated as

$$\widehat{z}_1 = \frac{1}{\|D_1(j,s)\|}\sum_{\overline{Q}_i\in D_1(j,s)} y_i, \qquad \widehat{z}_2 = \frac{1}{\|D_2(j,s)\|}\sum_{\overline{Q}_i\in D_2(j,s)} y_i, \quad (24)$$

where $\|\cdot\|$ denotes the number of elements.

Traverse all input variables and find the optimal segmentation variable j to form a pair (j, s). Divide the input space into two areas in turn. Then, repeat the above division process for each area until the terminal condition is met. In this way, a fuzzy granule regression tree is generated. The algorithm is described in Algorithm 2.

4.4. Boosted Fuzzy Granular Regression Trees. BFGRT can be an algorithm that integrates multiple fuzzy granular regression trees through the idea of ensemble learning to draw conclusions. It does not rely on only one fuzzy granular regression tree but adopts many fuzzy granular regression trees to solve the task together, using the weighted average of the regression values of multiple fuzzy granular regression trees as the final regression value. Assuming that T is the instance set, fuzzy granular space is $\overline{G}$, fuzzy granular rule base is R, the number of instances is n, and the number of

Input: instance set X, regression value set Y
Output: fuzzy granular tree f
(1) Remove the instances missing some attribute values.
(2) Normalize each attribute value.
(3) Calculate the cluster center set $C^* = \{c_1, c_2, \ldots, c_K\}$ (Algorithm 1).
(4) $X_1 \cup X_2 \cup \cdots \cup X_p = X$ and $\forall X_l, X_t \in X, X_l \cap X_t = \phi$. // Parallel distributed fuzzy granulation.
(5) $\forall X_t \subset X$//This is parallel process. Here, take X_t as example.
FOR $i = 1$ to $|X_t|$
$\exists x_i \in X_t$
FOR $j = 1$ to m
$\exists a \in A$, sample x is fuzzy granulated as $\overline{q}(x_i, a) = \sum_{j=1}^{K} s(x_i, a, c_j)/c_j$
END FOR
Build a fuzzy granular vector $\overline{Q}(x, A) = \sum_{j=1}^{|A|} \overline{q}(x, a_j)/a_j$;
Get label of x_i, y_i;
A fuzzy granular rule can be built. $r_i = <\overline{Q}(x_i, A), y_i>$;
END FOR
(6) Select the optimal segmentation variable j (i.e., the attribute a_j) and segmentation point s (i.e., $|\overline{q}(x_s, a_j)|$) by solving equation
$\min_{j,s}\left\{\min_{z_1}\sum_{\overline{Q}_i \in D_1(j,s)}(y_i - z_1)^2 + \min_{z_2}\sum_{\overline{Q}_i \in D_2(j,s)}(y_i - z_2)^2\right\}$
That is, traverse variable j to find the pair (j, s) that minimizes the loss function by fixing the segmentation variable j and scanning segmentation point s.
(7) Divide the area with the selected pair (j, s) and decide output value as follows:
$\widehat{z}_1 = 1/\|D_1(j,s)\|\sum_{\overline{Q}_i \in D_1(j,s)} y_i$
$\widehat{z}_2 = 1/\|D_2(j,s)\|\sum_{\overline{Q}_i \in D_2(j,s)} y_i$
$D_1(j,s) = \left\{\overline{Q}(x, A) \middle\| \overline{q}(x, a_j)| \le |\overline{q}(x_s, a_j)|\right\}$
$D_2(j,s) = \left\{\overline{Q}(x, A) \middle\| \overline{q}(x, a_j)| > |\overline{q}(x_s, a_j)|\right\}$
(8) Continue to call Step 6 and Step 7 for the two subregions until the number of split nodes is $|A|$.
(9) Divide the input fuzzy granular space into d regions $D_1, D_2, \ldots, D_d$ and generate a fuzzy granular regression tree
$f(\overline{Q}) = \sum_{i=1}^{d} \widehat{z}_i I(\overline{Q} \in D_i)$

Algorithm 2: Fuzzy granular regression tree.

attributes is m, we construct parallelly t fuzzy granular regression trees as follows:

Step 1: create J map tasks

Step 2: instance set extraction: randomly draw n fuzzy granular vectors from R with replacement and repeat them n times. The probability of each fuzzy granular vector being selected is $1/n$. The unselected fuzzy granular vectors form the out of bag data as the test set.

Step 3: attribute extraction: extract t attributes from A to compose attribute subset B $(B \subset A)$

Step 4: attribute selection: calculate the optimal segmentation attribute a_j and the optimal segmentation point $|\overline{q}(x_s, a_j)|$ in the data set of node, divide the node into two child nodes, and allocate the remaining fuzzy granular vectors to the child nodes

Step 5: generate a fuzzy granular tree. Repeat Step 3 in the fuzzy granular vector set of each child node to recursively split the nodes until all leaf nodes are generated.

Step 6: repeat steps 2–5 to get J different fuzzy granular regression trees, which correspond to J Map tasks

Step 7: BFGRT can consist of J fuzzy granular regression trees and the process can be executed by reduce task, that is,

$$F(\overline{Q}) = \sum_{j=1}^{J} \omega_j \cdot f_j(\overline{Q}), \tag{25}$$

Where $\omega_j = 1 - (\exp(\delta_j)/\sum_{i=1}^{J}\exp(\delta_i))$, and δ_j represents the root mean square error (RMSE) of j^{th} fuzzy granular tree. This detail of the algorithm can be illustrated in Algorithm 3.

4.5. Regression. Given a test instance, we fuzzy granulate the test instance to get a fuzzy granular vector. Then, use BFGRT to predict the fuzzy granular vector to achieve the regression value. Algorithm 6 demonstrates the algorithm.

5. Experimental Analysis

In this experiment, the results presented are generated on the server of 2∗Intel Xeon Gold6248R@2.50 GHz with 64 GB memory. Datasets include 3 datasets gathered from the UC Irvine Machine Learning Repository and 3 datasets with 1% noise constructed (Table 2), and 10-fold cross-validation which belongs to sampling without replacement was adopted to test the performance of BFGRT, as illustrated in Algorithm 3. The basic idea of 10-fold cross-validation is to partition the dataset into nonoverlapping L equal parts. In the training process, one part of dataset is selected to verify the generalization ability of the learner, and the remaining

Input: instance set X, regression value set Y, the number of fuzzy granular regression tree J
Output: boosted fuzzy granular regression trees F
(1) Get a fuzzy granular vector rule base R by parallel fuzzy granulation of the dataset (see Algorithm 2, Algorithm 4, and Algorithm 5.)
(2) Create J tasks, namely, $\text{map}_1, \text{map}_2, \ldots, \text{map}_J$
(3) Execute the following operations for each independent task ($j = 1, 2, \ldots, J$):
MapFunction(key, value), where key = offset of instance and value = (n, t) indicates that n fuzzy granular vectors are randomly
//selected from R.
//Randomly select t attributes from the attribute set A (constitutes attribute subset B_{j_t}, that is, $B_{j_t} \subset A$)
//Form a fuzzy granular rule set R_j, build a fuzzy granular regression tree f_j, and get its RMSE
$\delta_j = \sqrt{1/n \sum_{i=1}^{n} (\widehat{z}_i - y_i)^2}$, where $\widehat{z}_i = f_j(\overline{Q}(x_i, B_{j_t}))$, $y_i \in R_j$.
(4) FOR $i = 1$ to instances-total-number
(5) SubsetID = i mod J
(6) context.write(SubsetID, FuzzyGranularVector)
(7) END FOR
END MapFunction
(8) ReduceFunction(key, value)//Here, key = SubsetID, value = FuzzyGranualrVector
(9) Job.addCache(FuzzyGranularVector[SubsetID])
(10) $(f_{\text{SubsetID}}, \delta_{\text{SubsetID}})$ = train(SubsetID, FuzzyGranularVector)//(See Step 6–Step 9 of Algorithm 2.)
(11) context.write(1,$(f_{\text{SubsetID}}, \delta_{\text{SubsetID}})$)
END ReduceFunction
(12) $F(\cdot) = \sum_{j=1}^{J} \omega_j \cdot f_j(\cdot)$//Calculate BFGRT F composed of the linear combination of J fuzzy granular regression trees.
Where $\omega_j = 1 - (\exp(\delta_j) / \sum_{i=1}^{J} \exp(\delta_i))$.

Algorithm 3: Boosted fuzzy granular regression trees.

Input: <offset, instance> and t (the number of divided instance sets)
Output: <SubsetID, instance>
(1) FOR $i = 1$ to instances-total-number
(2) SubsetID = i mod t
(3) context.write(SubsetID, instance)
(4) END FOR

Algorithm 4: Map function of parallel fuzzy granulation.

Input: SubSetID, ClusterSet C, and instance subset X[SubSetID]
Output: Fuzzy Granular Set
(1) Job.addCache(X[SubSetID])
(2) FuzzyGranularSubSet[SubSetID] = Granulation(X[SubSetID],C)//See Step 5 of Algorithm 2.
(3) FuzzyGranularSet = context.write(1FuzzyGranularSubset[SubSetID])

Algorithm 5: Reduce function of parallel fuzzy granulation.

$L - 1$ parts are used as a training learner. After training L times, L learners can be obtained. This method is very similar to the bagging method and also supports parallel learning. Root mean square error (RMSE) and execution time are the metrics of the performance. We compared classic fuzzy granulation with parallel fuzzy granulation proposed in Figure 2 and analyzed the performance of support vector regression (SVR), random forest (RF), long short-term memory (LSTM), and BFGRT proposed in Figures 3–5 . We also gave the relation between the number of cluster centers and RMSE.

Fuzzy granulation of data is an important process of modeling. Some traditional fuzzy granulation methods do not require clustering, of which idea is to construct a matrix using the similarity of each sample to other samples. This kind of thinking does not have the conditions for parallel execution and can only be executed serially, and its time complexity is $O(n^2 * m)$, where n is the number of instances

Input: test instance x, cluster center set C, BFGRT $F = \{(f_1, f_2, \ldots, f_J; \omega_1, \omega_2, \ldots, \omega_J)\}$.
Output: regression value $\hat{y}^*$ of instance x
(1) Granulate x to fuzzy granular vector $\overline{Q}(x, A)$
(2) For $j = 1$ to J
$\hat{y}_j = f_j(\overline{Q}(x, A_j))$, where according to the parameters of f_j, attribute set selected satisfies $A_j \subset A$
$j = j + 1$
End For
(3) $F(Q(x, A)) = \sum_{j=1}^{J} \omega_j \cdot \hat{y}_j$
(4) $\hat{y}^* = F(Q(x, A))$
(5) Return $\hat{y}^*$

Algorithm 6: The algorithm of prediction.

Table 2: Datasets from the UC Irvine Machine Learning Repository.

Dataset	Number of instance	Number of attributes	Regression attribute
Bias correction of numerical prediction model temperature forecast	7750	25	Next_Tmin
Bias correction of numerical prediction model temperature forecast with noise	7750	25	Next_Tmin
Combined cycle power plant	9568	4	EP
Combined cycle power plant with noise	9568	4	EP
Metro interstate traffic volume	48204	9	Traffic volume
Metro interstate traffic volume with noise	48204	9	Traffic volume
Clickstream data for online shopping	165474	14	Price
Clickstream data for online shopping with noise	165474	14	Price

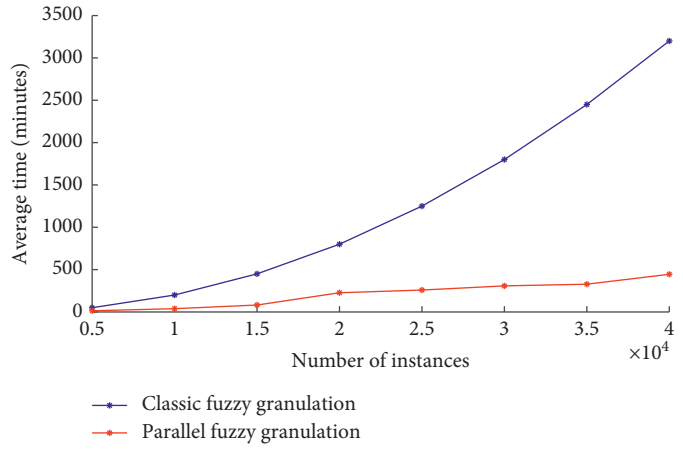

Figure 2: Comparison of executing time.

and m is the number of attributes. Parallel fuzzy granulation proposed is executed as follows. First, we obtain cluster centers by the clustering algorithm designed in the study. Next, we need to construct fuzzy granules through each instance and each cluster center. This process can be executed parallelly by MapReduce. The time complexity is $O(n * k * m/t)$, where k denotes the number of cluster centers $(k < n)$, and t represents the number of parallel tasks. Theoretically, the efficiency of BFGRT is $((nt/k) - 1) * 100\%$ $(k < n)$ higher than that of traditional fuzzy granulation.

MapReduce can be used for parallel fuzzy granulation. The main thought is as follows: partition the sequence file job into multiple independently runnable map tasks, assign them to several processors to execute, produce intermediate results, and then collect the reduce task operations to generate the final output. Map tasks and reduce tasks can be both parallelly executed. The MapReduce process is divided into two parts, i.e., map and reduce. Map function and reduce function of the fuzzy granulation process are shown in Algorithm 4 and Algorithm 5, respectively. The results of parallel granulation are demonstrated in Table 3. As

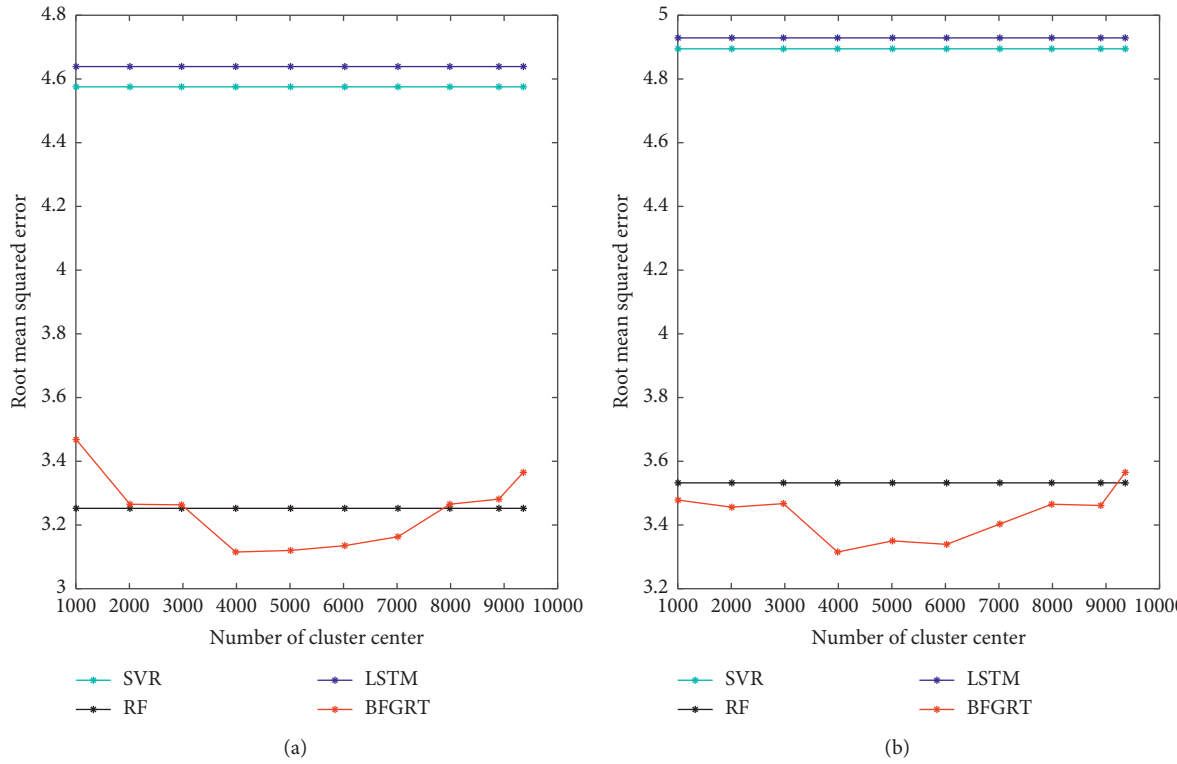

Figure 3: Comparison of dataset (a) combined cycle power plant and (b) combined cycle power plant with noise of 1%.

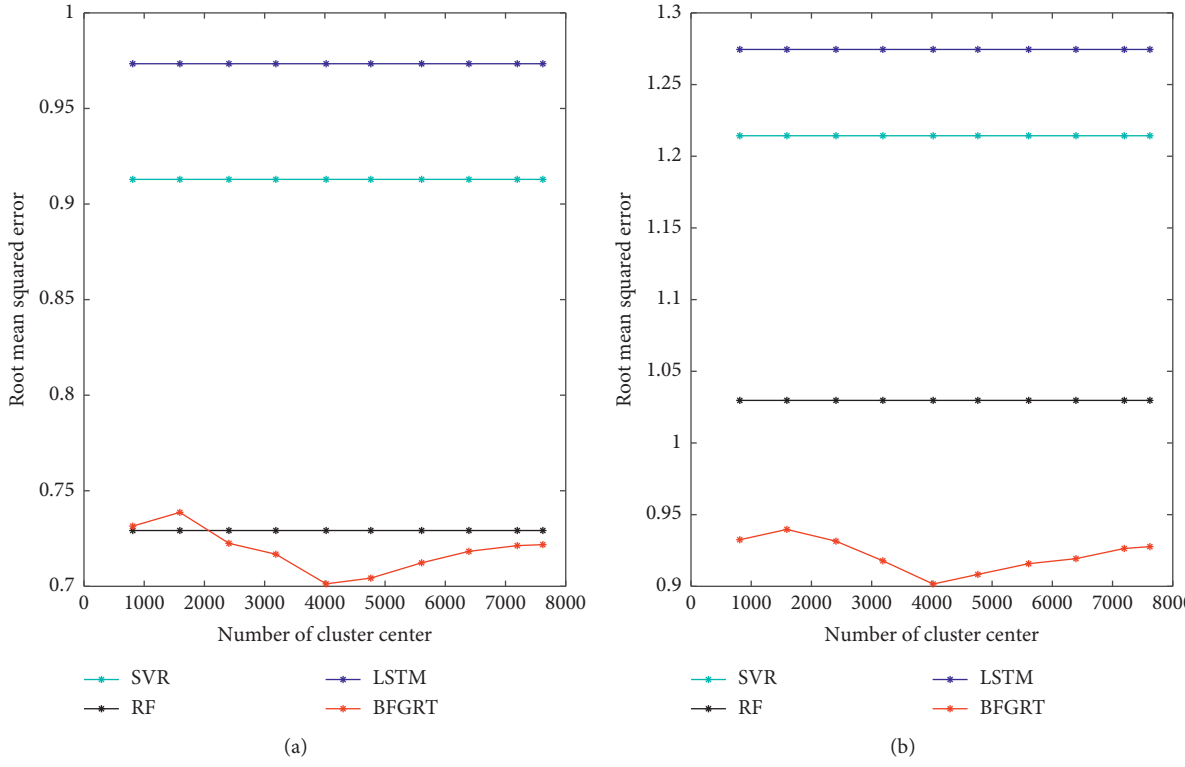

Figure 4: Comparison of dataset (a) bias correction of numerical prediction model temperature forecast and (b) bias correction of numerical prediction model temperature forecast with noise of 1%.

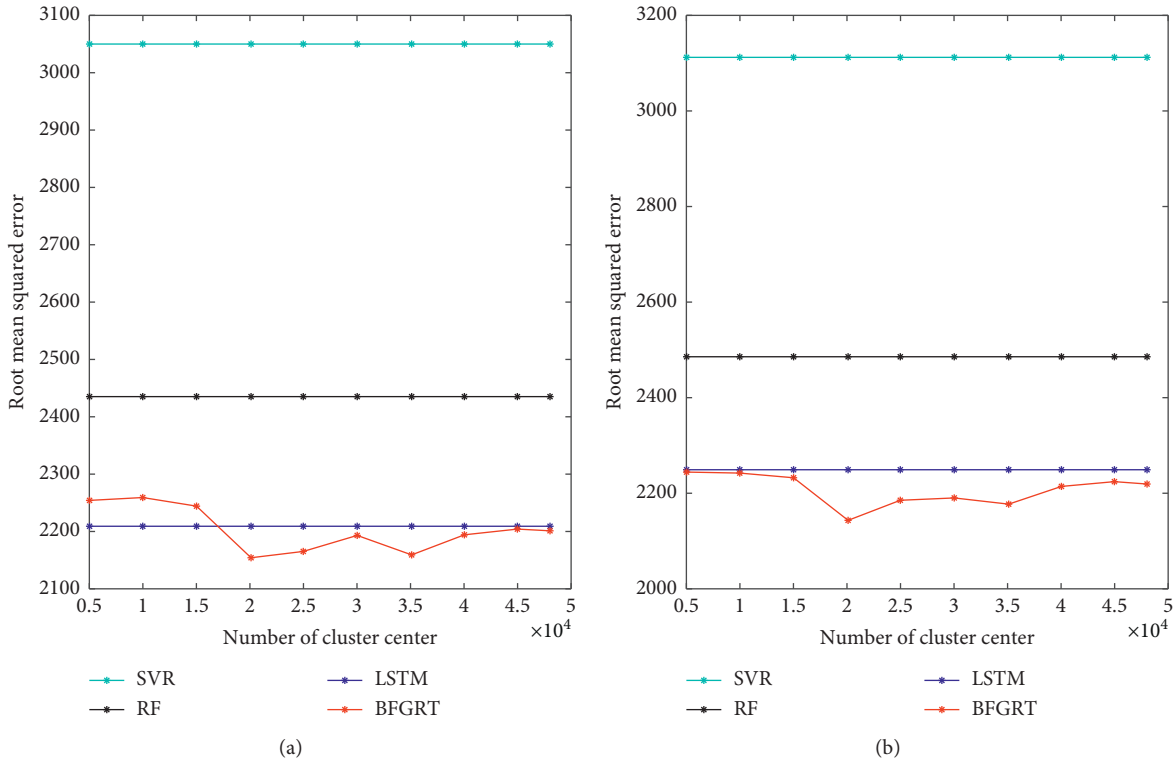

Figure 5: Comparison of dataset (a) metro interstate traffic volume and (b) metro interstate traffic volume with noise of 1%.

demonstrated in Figure 2, when the number of parallel tasks is 3, parallel fuzzy granulation performs better than classic fuzzy granulation by about 252%, 647%, and 438% regarding the metrics, minimum efficiency, maximum efficiency, and average efficiency, respectively.

As shown in Figure 3(a), tested on dataset combined cycle power plant, BFGRT show a shape that is low in the middle and high on both sides. When the number of cluster centers is between 3000 and 6000, the RMSE of BFGRT is lower than the other three methods. When the number of cluster centers is within 3000, the RMSE of BFGRT drops quickly, and the slope of the curve drops significantly. When the number of cluster centers is higher than 5000, the curve of BFGRT shows a small local oscillation, which is lower than the other three methods. In particular, when the number of cluster centers is 3986, the RMSE of BFGRT achieves the minimum value of 3.1151, which is about 46.88%, 4.22%, and 32.85% better than SVR, RF, and LSTM, respectively. BFGRT is slightly better than RF and far better than SVR and LSTM. When the number of cluster centers is 1006, the RMSE of BFGRT reaches the maximum value of 3.4681, which is about 24.20% and 25.24% better than RF and LSTM, respectively, and is about 6.64% worse than RF. On average, the RMSE of BFGRT is 3.2441, which is better than 4.5753 of SVR, 3.2522 of RF, and 4.6389 of LSTM (i.e., 29.10%, 0.25%, and 30.07% improvement, respectively).

After we added 1% noise to the dataset, as illustrated in Figure 3(b), the curve of BFGRT resembles an inverted "Mexican straw hat." When the number of cluster centers is 3986, the RMSE of BFGRT gets the minimum value of 3.3151, while that of SVR, RF, and LSTM are 4.8946, 3.5322, and 4.9289 (i.e., 32.27%, 6.15%, and 32.74% improvement, respectively). When the number of cluster centers is 9360, BFGRT obtain the maximum value of 3.5651, which improves the performance by about 27.16% and 27.67% compared with SVR and LSTM, respectively, and is about 0.93% worse than RF. On average, the RMSE of BFGRT is 3.4300, which performs about 29.92%, 2.89%, and 30.41% better than SVR, RF, and LSTM, respectively. In terms of the degree of noise influence, the RMSE of SVR, RF, LSTM, and BFGRT on the noisy dataset increases by about 6.98%, 8.61%, 6.25%, and 5.73%, respectively. BFGRT is less affected by noise and more robust than the other three algorithms.

As demonstrated in Figure 4(a), in the dataset bias correction of numerical prediction model temperature forecast, there are 7750 instances and 25 features. The shape of RMSE of BFGRT can be roughly divided into two segments with the number of cluster centers of 4000 as the cutting point. The part of less than 4000 is a descending curve and that of greater than 4000 is an ascending one. When the number of cluster centers gets 4021, the minimum value of RMSE of BFGRT is 0.7013, while SVR, RF, and LSTM just get 0.9129, 0.7292, and 0.9734 (that is, 23.18%, 3.83%, and 27.95% improvement, respectively). When the number of cluster centers is 1592, RMSE of BFGRT reach the maximum value of 0.7387, which is about 1.30% more than RF and 19.08% and 24.11% less than SVR and LSTM, respectively. The average RMSE of BFGRT is 0.7189, which is better than 21.25%, 1.42%, and 26.15% than SVR, RF, and LSTM, respectively.

Table 3: Parallel fuzzy granulation output.

Key	Value
$\mathrm{ID}(x_1)$	$\langle G_{a_1}(x_1), G_{a_2}(x_1), \ldots, G_{a_m}(x_1)\rangle$
$\mathrm{ID}(x_2)$	$\langle G_{a_1}(x_2), G_{a_2}(x_2), \ldots, G_{a_m}(x_2)\rangle$
...	...
$\mathrm{ID}(x_n)$	$\langle G_{a_1}(x_n), G_{a_2}(x_n), \ldots, G_{a_m}(x_n)\rangle$

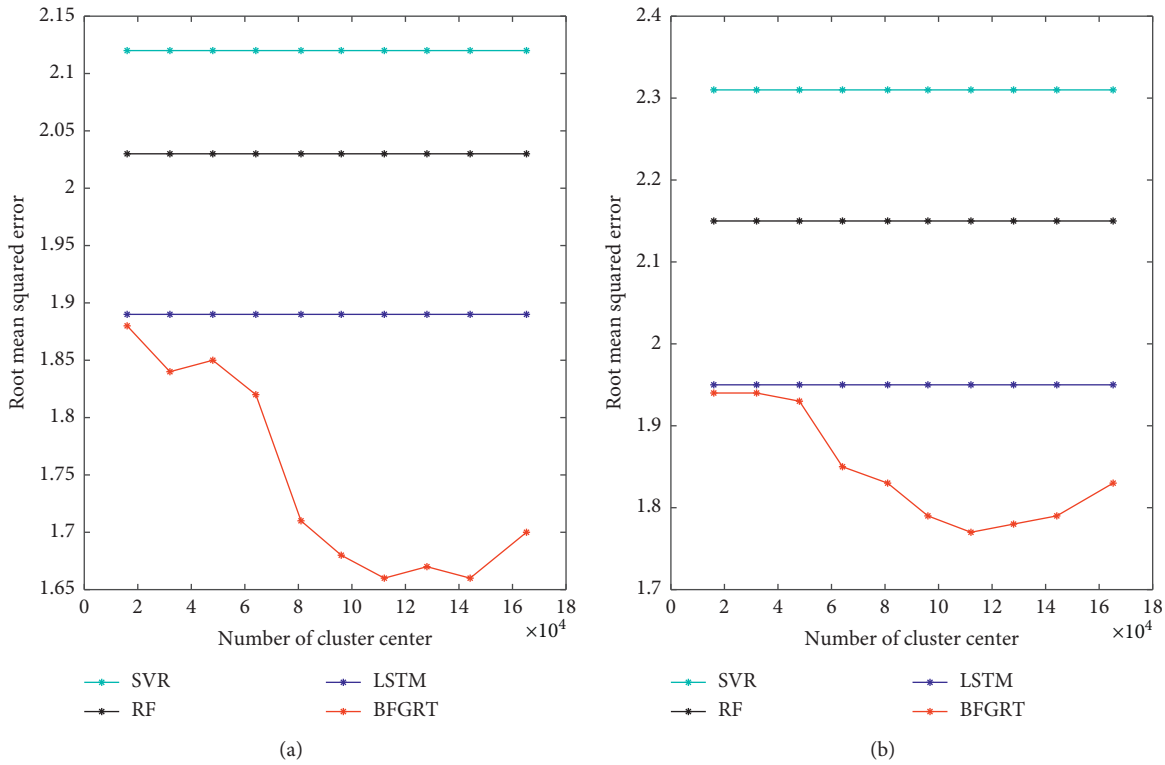

Figure 6: Comparison of (a) clickstream data for online shopping and (b) clickstream data for online shopping with noise of 1%.

The performance on a noisy dataset is shown in Figure 4(b). The RMSE of BFGRT is lower than the other three algorithms. The RMSE of BFGRT has a minimum value of 0.9016, a maximum value of 0.9397, and an average value of 0.9221. By contrast, the RMSE of SVR, RF, and LSTM is 1.2143, 1.0297, and 1.2745, respectively. The mean value of the RMSE of BFGRT is about 24.07%, 10.45%, and 27.65% better than SVR, RF, and LSTM, respectively. The performance of the algorithms analyzed in the dataset containing noise is as follows: SVR, RF, LSTM, and BFGRT have increased by 33.02%, 41.21%, 30.93%, and 27.48%, respectively. It can be seen that BFGRT can be less sensitive to noise than the other three algorithms.

The number of instances in dataset metro interstate traffic volume is more than 4 times that of datasets mentioned above. As illustrated in Figure 5(a), the maximum value of RMSE of BFGRT is 2259.1858, while that of SVR is 3050.0148 and that of RF is 2435.1646 (i.e., 25.93% and 7.23% improvement, respectively). Compared with LSTM, BFGRT decrease by about 2.27%. The minimum value of RMSE of BFGRT is 2154.1888, which performs 29.37%, 11.54%, and 2.49% better than SVR, RF, and LSTM, respectively. The mean value of RMSE of BFGRT is 2202.8738, which is 27.77%, 9.54%, and 0.28% better than SVR, RF, and LSTM, respectively. From the dataset containing 1% noise, as shown in Figure5(b), BFGRT improve the performance by about 29.08%, 11.20%, and 1.86% than SVR, RF, and LSTM, respectively. SVR, RF, LSTM, and BFGRT have increased by 2.04%, 2.07%, 2.10%, and 0.20% (the mean value of RMSE) on the noisy dataset, respectively. Compared with the other three algorithms, BFGRT is the algorithm least affected by noise.

The number of instances in dataset online shopping is more than 160000. As shown in Figure 6(a), the maximum value of RMSE of BFGRT is 1.88, while that of SVR is 2.12, that of RF is 2.03, and that of LSTM is 1.89 (i.e., 11.32%, 7.39%, and 0.53% improvement, respectively). From the dataset containing 1% noise, as shown in Figure 6(b), the maximum value of RMSE of BFGRT is 1.94, which has improved by about 16.01%, 9.77%, and 0.51%, respectively, compared with SVR, RF, and LSTM.

From the above analysis, it can be seen that BFGRT outperforms SVR, RF, and LSTM in the six datasets. In particular, it is stable on the three datasets with noise and is less disturbed by noise. Judging from the shape of the RMSE of BFGRT, it presents a form of low middle and high sides. When the number of cluster centers is close to the median of the number of instances, the performance can be optimal. For datasets that contain noise, we also found that BFGRT has the better robustness. The main reason is that BFGRT contains global comparison ideas in the fuzzy granulation process, which can overcome the noisy interference to some extent. This is also a great advantage of BFGRT designed.

6. Conclusion

In this study, we propose BFGRT suitable for the regression problem. In the algorithm, the idea of parallel fuzzy granulation is introduced to further improve the efficiency of data granulation. In the process of parallel fuzzy granulation, we design a clustering algorithm with automatic optimization of cluster centers. Through parallel fuzzy granulation, a regression problem can be solved in the fuzzy granular space where we present new operators and metrics between fuzzy granules. In the fuzzy granular space, we design a loss function to select optimal attribute as split point and construct recursively a fuzzy granular regression tree. Based on these, we build multiple fuzzy granular regression trees according to different attributes to form BFGRT to predict a test instance. In the future, BFGRT can be combined with cloud computing and thing of Internet to process the big data.

Data Availability

The dataset used to support the findings of this study is available from the UC Irvine Machine Learning Repository.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Scientific Research "Climbing" Program of Xiamen University of Technology, China (XPDKT20027), in part by the University Natural Sciences Research Project of Anhui Province, China (KJ2020A0660), and in part by the Natural Science Foundation of Anhui Province, China (2008085MF202).

References

[1] T. G. Dietterich, "Machine-learning research," *AI Magazine*, vol. 18, no. 4, pp. 97–136, 1997.

[2] M. Kearns and L. Valiant, "Cryptographic limitations on learning boolean formulae and finite automata," *Journal of the ACM*, vol. 41, no. 1, pp. 67–95, 1994.

[3] R. Avnimelech and N. Intrator, "Boosting regression estimators," *Neural Computation*, vol. 11, no. 2, pp. 499–520, 1999.

[4] D. H. Wolpert, "The supervised learning no-free-lunch theorems," *Soft Computing and Industry*, Springer, London, UK, pp. 25–42, 2002.

[5] Y. Liu, X. Yao, and T. Higuchi, "Evolutionary ensembles with negative correlation learning," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 380–387, 2000.

[6] Y.-K. Fang, Y. Fu, J.-L. Zhou, L. She, and C.-J. Sun, "Selective boosting algorithm for maximizing the soft margin," *Journal of Software*, vol. 23, no. 5, pp. 1132–1147, 2012.

[7] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[8] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, pp. 197–227, 1990.

[9] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[10] L. Breiman, "Randomizing outputs to increase prediction accuracy," *Machine Learning*, vol. 40, no. 3, pp. 229–242, 2000.

[11] G. P. Zhang, "A neural network ensemble method with jittered training data for time series forecasting," *Information Sciences*, vol. 177, no. 23, pp. 5329–5346, 2007.

[12] I. A. Gheyas and L. S. Smith, "A novel neural network ensemble architecture for time series forecasting," *Neurocomputing*, vol. 74, no. 18, pp. 3855–3864, 2011.

[13] S. Hassan, A. Khosravi, and J. Jaafar, "Improving load forecasting accuracy through combination of best forecasts," in *Proceedings of the 2012 IEEE International Conference on Power System Technology (POWERCON)*, 2012.

[14] F. Gao, P. Kou, L. Gao, and X. Guan, "Boosting regression methods based on a geometric conversion approach: using SVMS base learners," *Neurocomputing*, vol. 113, pp. 67–87, 2013.

[15] M. J. Cloud, B. C. Drachman, and L. P. Lebedev, "A brief introduction to interval analysis," *Inequalities*, vol. 113, pp. 179–193, 2014.

[16] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[17] Z. a. Pawlak, "Rough sets," *International Journal of Computer & Information Sciences*, vol. 11, no. 5, pp. 341–356, 1982.

[18] D. Miao, *Granular Computing: Past, Present and Prospect*, Science Press, Beijing, China, 2007, in Chinese.

[19] Y. Y. Yao, "Information granulation and rough set approximation," *International Journal of Intelligent Systems*, vol. 16, no. 1, pp. 87–104, 2001.

[20] Y. Y. Yao, "A partition model of granular computing," *Transactions on Rough Sets I*, vol. 3100, pp. 232–253, 2004.

[21] Y. Y. Yao, "Granular computing for data mining," in *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2006*SPIE, Kissimmee, FL, USA, 2006.

[22] T. Y. Lin, "Granular computing: a problem solving paradigm," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 132–137, FUZZ-IEEE, Reno, NV, USA, May 2005.

[23] Z. Zheng, H. Hu, and Z. Shi, "Tolerance granular space and its applications," in *Proceedings of the 2005 IEEE International Conference on Granular Computing*, Beijing, China, 2005.

[24] F. Wang, "Computing with words and a framework for computational linguistic dynamic systems," *Pattern Recognition and Artificial Intelligence*, vol. 14, no. 4, pp. 377–384, 2001, in Chinese.

[25] M. Kryszkiewicz, "Rough set approach to incomplete information systems," *Information Sciences*, vol. 112, no. 1–4, pp. 39–49, 1998.

[26] J. Stefanowski and A. Tsoukias, "On the extension of rough sets under incomplete information," in *Proceedings of the 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing*, pp. 73–81, Berlin, Germany, 1999.

[27] G. Wang, "Extension of rough set under incomplete information systems," in *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems: FUZZ-IEEE'02*, vol. 2, pp. 1098–1103, Honolulu, HI, USA, 2002.

[28] Z. Pawlak, "Granularity of knowledge, indiscernibility and rough sets," in *Proceedings of the IEEE International Conference on Fuzzy Systems Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36228)*, Anchorage, AK, USA, 2002.

[29] L. Polkowski and A. Skowron, "Rough-neuro computing," in *Rough Sets and Current Trends in Computing*, W. Ziarko and Y. Yao, Eds., pp. 57–64, RSCTC, Berlin, Germany, 2001.

[30] J. F. Peters, A. Skowron, Z. Suraj, M. Borkowski, and W. Rzcasa, "Measures of inclusion and closeness of information granules: a rough set approach," in *Rough Sets and Current Trends in Computing*, vol. 2475, pp. 300–307, Springer, Berlin, Germany, 2002.

[31] B. Zhang and L. Zhang, *Problem Solving Theory and Application*, Tsinghua University Press, Beijing, China, 1990, in Chinese.

[32] F. Xu, "The approach of the fuzzy granular computing based on the theory of quotient space (in Chinese)," *Pattern Recognition and Artificial Intelligence*, vol. 17, no. 4, pp. 424–429, 2004, in Chinese.

[33] X. Zhu, W. Pedrycz, and Z. Li, "A development of granular input space in system modeling," *IEEE Transactions on Cybernetics*, vol. 51, no. 3, pp. 1639–1650, 2021.

[34] T. Yang, X. Zhong, G. Lang, Y. Qian, and J. Dai, "Granular matrix: a new approach for granular structure reduction and redundancy evaluation," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 12, pp. 3133–3144, 2020.

[35] Y. Yao, "Three-way decision: an interpretation of rules in rough set theory," in *Proceedings of the International Conference on Rough Sets and Knowledge Technology RSKT 2009: Rough Sets and Knowledge Technology*, pp. 642–649, Gold Coast, Australia, July 2009.

[36] Q. Zhang, Z. Huang, and G. Wang, "A novel sequential three-way decision model with autonomous error correction," *Knowledge-Based Systems*, vol. 212, no. 5, Article ID 1, 2021.

[37] X. Ye and D. Liu, "An interpretable sequential three-way recommendation based on collaborative topic regression," *Expert Systems with Applications*, vol. 168, pp. 1–16, 2021.

[38] J. Chen, H. Chen, X. Wan, and G. Zheng, "MR-ELM: a mapreduce-based framework for large-scale elm training in big data era," *Neural Computing and Applications*, vol. 27, no. 1, pp. 101–110, 2016.

[39] J. Zhai, S. Zhang, and C. Wang, "The classification of imbalanced large data sets based on mapreduce and ensemble of elm classifiers," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 3, pp. 1009–1017, 2017.

[40] S. Ramírez-Gallego, A. Fernández, S. García, M. Chen, and F. Herrera, "Big data: tutorial and guidelines on information and process fusion for analytics algorithms with mapreduce," *Information Fusion*, vol. 42, pp. 51–61, 2018.

[41] J. Zhai, X. Zhou, S. Zhang, and T. Wang, "Ensemble RBM-based classifier using fuzzy integral for big data classification," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 11, pp. 3327–3337, 2019.

[42] Q. Wu, H. Wang, X. Yan, and X. Liu, "Mapreduce-based adaptive random forest algorithm for multi-label classification," *Neural Computing and Applications*, vol. 31, no. 12, pp. 8239–8252, 2019.

[43] J. Zhai, S. Zhang, M. Zhang, and X. Liu, "Fuzzy integral-based elm ensemble for imbalanced big data classification," *Soft Computing*, vol. 22, no. 11, pp. 3519–3531, 2018.

[44] W. Li, X. Ma, Y. Chen et al., "Random fuzzy granular decision tree," *Mathematical Problems in Engineering*, vol. 2021, Article ID 5578682, 17 pages, 2021.