Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 2977126, 15 pages
https://doi.org/10.1155/2023/2977126

Research Article

Decision Key-Value Feature Construction for Multihoming Big Data Network

Musaddak Maher Abdul Zahra,[1] **M. Nagabushanam**,[2] **S. Kumaraswamy**,[3] **Devangkumar Umakant Shah**,[4] **Charanjeet Singh**,[5] **and Mohammad Najmus Saquib Hasan**[6]

[1]*Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah 51001, Iraq*
[2]*Department of Electronics & Communication Engineering, M.S. Ramaiah Institute of Technology, Bangalore, Karnataka, India*
[3]*Department of Computer Science and Engineering, Global Academy of Technology, Bengaluru, India*
[4]*Department of Electrical Engineering, K. J. Institute of Engineering & Technology, Savli, Vadodara, India*
[5]*Electronics and Communication Department, Deenbandhu Chhotu Ram University of Science and Technology, Murthal, India*
[6]*Wollega University, Ethiopia*

Correspondence should be addressed to Mohammad Najmus Saquib Hasan; mohammadk@wollegauniversity.edu.et

Received 26 May 2022; Accepted 15 September 2022; Published 30 April 2023

Academic Editor: Isma F Siddiqui

The random forest algorithm under the MapReduce framework has too many redundant and irrelevant features, low training feature information, and low parallelization efficiency when dealing with multihoming big data network problems, so parallelism is based on information theory, and norms is proposed for random forest algorithm (PRFITN). In this paper, the technique used first builds a hybrid dimensional reduction approach (DRIGFN) focused on information gain and the Frobenius norm, successfully reducing the number of redundant and irrelevant features; then, an information theory feature is offered. This results in the dimensionality-reduced dataset. Finally, a technique is suggested in the Reduce stage. The features are grouped in the FGSIT strategy, and the stratified sampling approach is employed to assure the information quantity of the training features in the building of the decision tree in the random forest. When datasets are provided as key/value pairs, it is common to want to aggregate statistics across all objects with the same key. To acquire global classification results and achieve a rapid and equal distribution of key-value pairs, a key-value pair redistribution method (RSKP) is used, which improves the cluster's parallel efficiency. The approach provides a superior classification impact in multihoming large data networks, particularly for datasets with numerous characteristics, according to the experimental findings. We can utilize feature selection and feature extraction together. In addition to minimizing overfitting and redundancy, lowering dimensionality contributes to improved human interpretation and cheaper computing costs through model simplicity.

1. Introduction

A classification algorithm is a supervised learning algorithm, which can discover classification rules and construct classification models based on labeled information, to predict the attributes of unlabeled data [1]. Among the classification algorithms, random forest (RF) has been used in text classification [2] and environmental prediction in recent years because of its strong stability and good tolerance to noise and outliers [3, 4]. Credit evaluation [5], bioinformatics [6], medical diagnosis [7], and other fields have received extensive attention. Random forest, as the name indicates, is a classifier that employs a decision tree based on diverse subsets of the supplied dataset and combines them to improve the dataset's forecasting accuracy. Instead, then relying on a single decision tree, the random forest takes forecasts from each tree and predicts the outcome based on the bulk of predictions' choices. The increasing number of trees in the forest prevents maximum reliability and generalization.

Big data systems nowadays are backed by a variety of processing, analytical, and dynamic visualization capabilities. These platforms make it possible to retrieve knowledge and information from dynamic environments that are complicated. Through suggestions and automatic identification of abnormalities, deviant behavior, or new trends, they also assist in decision-making [8]. Big data has become a research hotspot as information technology and network technologies have advanced. Big data has 4V characteristics compared to traditional data—volume (large quantity), variety (variety), velocity (fast speed), and value (low density) [8]—which requires a longer running time and more memory capacity when processing big data, and it is especially important to improve computer hardware to meet people's needs for big data analysis and processing difficulties. By refining the classic random forest technique and merging it with the distributed computing model, the notion of parallelized computing becomes highly relevant at this time, and it has become the major focus of current research.

In recent years, researchers and businesses have embraced Google's MapReduce parallel programming methodology in the area of large data processing owing to its ease of use, automated fault tolerance, and high scalability. Writing programs that can process big data in parallel on several nodes is possible using the MapReduce programming style. Large amounts of complicated data may be analyzed using analytical tools like MapReduce. The MapReduce concept is aimed at making the translation and analysis of huge datasets more straightforward while allowing programmers to concentrate on algorithms rather than data management. The paradigm makes data-parallel algorithms easy to build. This paradigm has been utilized in a number of ways, notably Google's (C++) technique and Apache's Hadoop implementation (written in Java). Both programs run in a peer-to-peer environment on massive hardware platforms. At the same time, Hadoop and Spark, which represent distributed computing systems, have gotten a lot of attention [9]. Many random forest techniques based on the MapReduce computing architecture have been effectively deployed to large data analysis and processing at this time. Among these, the MapReduce-based parallelized random forest method MR_RF [10] uses the divide-and-conquer approach, using the MapReduce paradigm to split the input and transfer it to several computing nodes to create a base classifier, then aggregating the output of each computing node. Create a model of a random forest.

MapReduce is extremely scalable and runs on a big cluster of common computers. Many terabytes of information are often processed on thousands of computers during a typical MapReduce calculation [11]. The MapReduce model is then called again, and the created random forest is utilized to forecast the test set in order to acquire classification accuracy, completing the random forest algorithm's parallelization. The parallelization framework is called twice before and after the algorithm, and the intermediate results are read out several times. It takes a lot of time to research and write. Literature developed a revised MR_RF method [12] to lower the temporal complexity of the MR_RF technique, which employs out-of-bag data to directly compute the classification model's generalization error in order to estimate the random forest's classification accuracy. The number of calls to parallel frameworks has been reduced. However, in a big data environment, a significant number of redundant and irrelevant characteristics in the dataset diminish the quality of the features picked by the decision tree while building the random forest model, which has an impact on the random forest model's overall classification accuracy.

The author devised a parallel random forest (PRF) approach to lessen the effect of redundant and irrelevant features in big datasets on the model [13]. A hybrid strategy integrating data-parallel and task-parallel optimization is used to optimize the PRF algorithm [14]. The out-of-bag data is utilized as the training set to determine the classification accuracy corresponding to each decision tree as the weight, which is then employed in the model prediction step. Although the PRF method increases the random forest's classification performance by optimizing the training features, it does not minimize the amount of redundant and irrelevant features in the dataset; therefore, the resulting training feature set includes greater redundancy and irrelevance. In light of this, the authors presented PRFMIC [15], a parallel random forest method based on the maximum information coefficient. The characteristics are separated into three intervals using the maximum information coefficient, the low correlation interval is eliminated, and the high correlation interval is chosen. Compared to a single decision tree method, the random forest approach is more accurate [16]. The random forest model is built in parallel as the features in the interval and the midcorrelation interval create feature subsets. Even though the method considers the impact of irrelevant characteristics on the random forest model, redundant data features cannot be given during the random forest modeling step.

The importance of creating big data applications has increased over the past several years. The information derived from enormous amounts of data is increasingly relied upon by several companies from various industries. Traditional data platforms and methodologies, on the other hand, perform poorly in the context of big data. They lack scalability, efficiency, and accuracy and have a slow reaction time. Much effort has been expended in addressing the tough big data issues. As a result, several distributions and technical advancements have occurred. It offers comparisons based on several system levels, including data storage layer, data processing layer, data querying layer, data access layer, and data management layer, in addition to providing a global overview of the primary big data technologies. It classifies and examines the primary technological aspects, benefits, and limitations.

The aforementioned approach does not consider the amount of information in the training features while producing the training feature set, but it does raise the correlation between decision trees and decision trees, which impacts the overall accuracy of the random forest model. The random forest's overall accuracy is influenced by the decision tree trained by the training feature set; however, owing to the load imbalance, the method takes too long in the prediction and classification stages, reducing the random forest's overall parallelization efficiency [17]. There are still pressing concerns to be addressed, such as how to minimize duplicate and unnecessary features in huge datasets, how to enhance the quantity of training feature information, and how to improve the parallel efficiency of algorithms. This work presents a parallel random forest method based on information theory and norms to address these issues (PRFITN). First, the method creates a dimensionality-reduced dataset using a hybrid dimensionality reduction approach called DRIGFN (dimension reduction based on information gain and Frobenius norm) based on information gain and Frobenius norm, successfully minimizing redundancy and irrelevance. Furthermore, the algorithm presents a feature grouping strategy based on information theory (FGSIT), which groups the features according to the FGSIT strategy and uses the stratified sampling approach to guarantee that the decision tree in the random forest is formed. The quantity of data in the feature subset enhances the classification results' accuracy. Finally, in the Reduce phase, a key-value pair redistribution strategy (RSKP) is proposed to obtain the global classification results, realize the fast and even distribution of key-value pairs, and thus improve the parallel efficiency of the cluster, taking into account the impact of the cluster load on the efficiency of the parallel algorithm. The experimental findings suggest that the algorithm performs better in a big data context, particularly when dealing with datasets with a high number of characteristics.

The next section describes the related concepts followed by PRFITN algorithms that are analyzed in this research. Then, the results of the algorithms are analyzed, and finally, conclusions are drawn.

2. Related Concepts

2.1. Introduction to Related Concepts

Definition 1 (information gain). Given discrete variable X and their corresponding category Z, the information gain $\mathrm{IG}(Z\,;X)$ is calculated by the following formula:

$$\mathrm{IG}(Z\,;X) = H(Z) - H(Z|X), \tag{1}$$

where $H(Z)$ is the information entropy about category Z and $H(Z \mid X)$ is the conditional entropy about variable X and category Z.

Definition 2 (mutual information). Given discrete variables X and Y, the mutual information $I(X\,;Y)$ is calculated by the following formula:

$$I(X\,;Y) = \sum_{i=1}^{l}\sum_{j=1}^{m} p\left(x_i, y_j\right) lb \frac{p\left(x_i, y_j\right)}{p(x_i)p\left(y_j\right)}. \tag{2}$$

Definition 3 (conditional mutual information). Given discrete variables X and Y and their corresponding category Z, the conditional mutual information $I(X\,;Y \mid Z)$ is calculated by the following formula:

$$I(X\,;Y|Z) = H(X|Z) - H(X|Y, Z), \tag{3}$$

where $H(X \mid Z)$ is the conditional entropy about variable X and category Z and $H(X \mid Y, Z)$ is the conditional entropy about variables X, Y and category Z.

Definition 4 (Frobenius norm). Given that $A = (a_{ij})_{m\times n} \in \mathbb{R}^{m\times n}$ is an $m \times n$-dimensional matrix and a_{ij} is an element in the matrix, then the Frobenius norm $\|X\|_F$ can be calculated by

$$\|X\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} {a_{ij}}^2}. \tag{4}$$

2.2. Principal Component Analysis Algorithm. Principal component analysis (PCA) [18] is a multivariate statistical method for dimensionality reduction. Its main purpose is to find a transformation matrix W and reduce the dimension of the dataset under the condition of maintaining the maximum variation. Principal components analysis (PCA) is the most widely used unsupervised dimensionality reduction technique for producing relevant characteristics by integrating the data points in linear (linear PCA) or nonlinear (kernel PCA) arrangements (features). Significant features are created by linearly lowering correlated data to a lower fraction of set of variables. This is accomplished by projecting (dot producing) the actual information into the simplified PCA space using the covariance/correlation matrix eigenvectors, also known as the principal components (PCs). PCA is a linear transformation of data into a sequence of uncorrelated variables existing in the simplified PCA space, where the first element explains the most variation and each successive component explains less. The PCA algorithm is mainly divided into four steps: (1) establish a data matrix and standardize the original dataset; (2) establish a correlation coefficient matrix and calculate the eigenvalue λ of each principal component and the corresponding number of eigenvectors; (3) according to the eigenvalues and the cumulative contribution rate, determine the number of principal components required; and (4) combine the eigenvectors corresponding to the principal components to obtain the transformation matrix W to reduce the dimension of the original dataset.

2.3. Support Vector Machine Algorithm. The support vector machine (SVM) algorithm [19] is a data mining algorithm based on statistical theory. It mainly selects an optimal classification hyperplane that meets the classification requirements, so that the hyperplane can guarantee the classification accuracy. At the same time, the blank areas on both sides of the hyperplane can be maximized [20]. SVMs are utilized in web pages, intrusion detection, face identification, email categorization, genre classification, and handwriting recognition, among other applications. We utilize SVMs in machine learning for several reasons, including this. Simultaneous categorization and extrapolation of linear and nonlinear data are supported by SVM. The SVM algorithm is mainly divided into three steps: (1) construct the classification hyperplane $f(x) = v^T X$, where v is the weight of the hyperplane and X is the data vector matrix; (2) use the kernel function to solve the classification hyperplane and obtain the hyperplane weight v; and (3) use the hyperplane weight v to predict the data classification. Large datasets are not a good fit for the SVM algorithm. When the targeted classes are overlapping and the dataset includes more noise, SVM does not really perform very well. The SVM will perform poorly when there are more training data samples than characteristics for each data point.

3. PRFITN Algorithm

The PRFITN algorithm mainly includes three stages: data dimensionality reduction, feature grouping, and parallel construction of random forests. (1) In the data dimension reduction stage, the DRIGFN strategy is proposed to accurately identify and delete redundant and irrelevant features in the dataset, and the dimensionality-reduced dataset DB*is obtained. (2) In the feature grouping stage, the FGSIT strategy is proposed to be used in order to measure the importance of features, and then, distribute features cyclically on this basis, to obtain two sets of feature subsets Q and S. (3) In the stage of parallel construction of random forests, the RSKP strategy is proposed to optimize the MapReduce computing model and improve its performance [21]. Parallelization efficiency and use the optimized MapReduce model to build random forests helps to predict and classify datasets, that obtain the accuracy of random forests.

3.1. Data Dimensionality Reduction. At present, dimensionality reduction algorithms mainly include feature selection and feature extraction. However, in the big data environment, due to the existence of a large number of redundant and irrelevant features in the dataset, the feature selection or feature extraction methods alone cannot achieve better results. Feature set for paper proposes a DRIGFN strategy to identify and filter redundant and irrelevant data in a big data environment [22]. First, combined with the MapReduce model, the feature information gain value is calculated in parallel to remove irrelevant features; then, the Frobenius norm is used to evaluate the amount of information loss, classification error, and control overfitting of the classifier, and on this basis, a global algorithm is proposed. The optimization function is used to iteratively optimize the dimensionality reduction parameters. Suppose $X = [x_1, x_2 \cdots, x_d] \in \mathbb{R}^{n\times d}$ represents n samples in the d-dimensional feature space of the original dataset DB, the dataset DB contains v different categories, and Y $\in \mathbb{R}^{n\times 1}$ represents the feature matrix X. In the corresponding label, the DRIGFN policy is as follows.

3.2. Feature Selection. For dataset DB, the main purpose of feature selection is to reduce the number of irrelevant features. The specific process is as follows: first, use the default file block strategy in Hadoop to divide the feature space of the original dataset into file blocks of the same size; then, the file blocks are used as input data [23]. According to Definition 1, the Mapper node calls Map. The function calculates the information gain of each feature in the form of key-value pair <key, value> (key is the feature name, and value is the information gain of the corresponding feature) and combines each key-value pair to obtain the feature information gain set $A = \{<\text{key}_1, \text{value}_1>, <\text{key}_2, \text{value}_2>, \cdots, <\text{key}_d, \text{value}_d>\}$. Finally, according to the information gain value corresponding to the feature, the elements in the set A are sorted in descending order, the features that are ranked later in the set A are removed, and the new feature matrix $X' = [x_1, x_2 \cdots, x_d] \in \mathbb{R}^{n\times m} (1 \le m \le d)$ is obtained by recombining and the dataset DB$'$ obtained by merging the feature matrix X' and label vector Y obtained after processing in columns is passed to the next stage. Feature selection is performed as follows.

3.3. Feature Extraction. In the feature extraction stage, to further optimize the dataset DB$'$ after feature selection, firstly, use principal component analysis and support vector machine algorithm to obtain the initial parameters, and use the received parameters to reconstruct the feature matrix; secondly, use the Frobenius norm loss of information [20]. The classification error and the degree of overfitting of the classifier are estimated; finally, to minimize the sum of information loss, classification error, and the degree of overfitting, a global optimization function is proposed to optimize the transformation matrix and the classification matrix. The specific process of feature extraction is described as follows:

(1) Initialization of parameters and reconstruction of feature matrix knowing the feature matrix X' and the dataset DB$'$

First, adopt the principal component analysis (PCA) to obtain the initial transformation matrix W $\in \mathbb{R}^{o\times m} (1 \le o \le m)$, that is,

$$X'' = X' W^T \in \mathbb{R}^{n\times o}, \tag{5}$$

where X'' is the feature matrix obtained after dimensionality reduction by PCA, which is merged with the label Y in columns to obtain the transformed dataset DB$''$.

Secondly, the support vector machine (SVM) algorithm is used to obtain the classification matrix $v \in \mathbb{R}^{o\times 1}$ about X'' with all the samples in the dataset DB$''$ as the training set, according to which the predicted label of X'' can be calculated as

$$f\left(X''\right) = X'' v = X' W^T v \Big] \in \mathbb{R}^{n\times 1}. \tag{6}$$

Next, in order to evaluate the loss of information in the feature extraction process, the transformation matrix is used to reconstruct the feature matrix X'', and the reconstruction matrix X_c of X'' can be expressed as

```
Input: original dataset DB.
Output: feature matrix $X'$, dataset $DB'$.
    1. Block ⟵ split the feature space of the original dataset
    2. Key: feature name
    3. Value: combine the feature space of the key with label Y
    4. For each feature $x_k$ in each block, do
    5. key$_k$⟵ feature name
    6. $IG(Y;x_k) \longleftarrow -\sum_{i=1}^{v} p_i lbp_i - (-\sum_{j=1}^{\alpha} |n_j|/|n|\sum_{i=1}^{v} (|n_{ij}|/|n_j|) lb(|n_{ij}|/|n_j|))$//according to Definition 1, calculate the information gain value of feature $x_k$; $p_i$ represents the proportion of category $i$ in the data set; $\alpha$ represents the tuple divided according to the value of feature $x_k$; |n|, $|n_j|$, and $|n_{ij}|$, respectively, correspond to the total number of data samples, the number of elements in tuple $j$ divided by the value of feature $x_k$, and the number of aspects of category $i$ in tuple $j$
    7. value$_k$ ⟵ IG (Y; $x_k$)//the information gain value of $xk$ is assigned to value $k$
    8. A ⟵ <key$_k$, value$_k$>//put the key-value pair <key$_k$, value$_k$> corresponding to the feature $x_k$ into the set A
    9. End for
    10. Sorted(A)//sort A in descending order of value
    11. Delete the later features in set A//delete the elements arranged later in A
    12. Return $X'$//the remaining features in $X$ are the feature matrix $X'$
    13. $DB'$ ⟵ combine $X'$ and $Y$ by column
    14. Return $DB'$
```

Algorithm 1: Feature selection.

$$X_c = X''W = X'W^TW \in \mathbb{R}^{n\times m} \tag{7}$$

(2) Estimation of the amount of information loss, classification error, and the degree of overfitting of the classifier

According to the transformation matrix W, classification matrix v, and reconstruction matrix X_c obtained in the previous step, this part will use the Frobenius norm to estimate the amount of information loss, classification error, and classifier degree overfitting.

Since the reconstruction matrix, X_c, is obtained by transforming the feature matrix X' through the matrix W, there will be more or fewer differences with the elements of X', so the Frobenius norm is used to process the differences between the elements in the two matrices and sum them up; the obtained result can reflect the matrix X'' and the moment after dimensionality reduction. The amount of information loss compared to the matrix X' is specifically defined as follows.

Definition 5 (information loss X_{error}). Given the known feature matrix X' and reconstruction matrix X_c, then according to Definition 4, the information loss X_{error} can be expressed as

$$X_{\text{error}} = \|X' - X_c\|_F^2. \tag{8}$$

Similarly, the difference between the predicted label $f(X'')$ and the label Y can also be measured by the Frobenius norm, which is defined as follows.

Definition 6 (classification error Y_{error}). It is known that Y is the label corresponding to the feature matrix X and $f(X'')$ is the predicted label predicted by the support vector machine; then according to Definition 4, the classification error Y_{error} can be expressed as

$$Y_{\text{error}} = \left\|Y - f\left(X''\right)\right\|_F^2. \tag{9}$$

Finally according to the Frobenius norm, v_{error} is designed, and the degree of overfitting of the classification is controlled by the v value, which is specifically defined as follows.

Definition 7 (overfitting degree v_{error}). Knowing that v is the classification matrix of the feature matrix X'', then according to Definition 4, the overfitting degree v_{error} can be expressed as

$$v_{\text{error}} = \|v\|_F^2. \tag{10}$$

According to the definition of the Frobenius norm, it can be inferred that the more uniform the distribution of elements of v, the smaller the value of v_{error}; on the contrary, the larger the value of individual elements in v, the larger the value of v_{error}.

(3) Global optimization function

To obtain the global optimal transformation function $W*$, it is necessary to satisfy the overall depreciation of X_{error}, Y_{error}, and v_{error} at the same time, so combined with the three Equations (8)~(10), the global optimization function $E(W, v)$ is defined as follows.

```
Input: dataset DB$'$, weight parameters $C_1$, $C_2$.
Output: feature matrix $X*$, dataset DB $*$.
1. W ⟵ the conversion matrix of $X'$ calculated according to PCA
2. $X'' = X' WT$
3. DB$''$ ⟵ combine $X''$ and $Y$ by column
4. $v$ ⟵ the classification matrix of DB$''$ according to SVM
5. $E(W, v) = \|X' - X' W^T W\|_F^2 + C_1 \|Y - X''v\|_F^2 + C_2 \|v\|_F^2$
6. Do//iteratively solve the global optimal transformation matrix W $*$
. $W'$ ⟵ conversion matrix solved by $\nabla E(W, v)_W = 0$
8. $v'$ ⟵ classification matrix solved by $\nabla E(W', v)_v = 0$
9. End do until convergence
10. Get $(W*, v*)$
11. $X* \longleftarrow X' WT$
12. Return $X*$
13. DB $*$ ⟵ combine $X*$ and $Y$ by column
14. Return DB $*$
```

Algorithm 2: Feature extraction.

Definition 8 (global optimization function $E(W, v)$. Knowing the feature matrix X' and Y labels, the corresponding transformation matrix and classification matrix are W and v, respectively, from which the information loss amount and classification error about X' can be obtained. If the degree of overfitting is X_{error}, Y_{error}, and v_{error}, then the global optimization function $E(W, v)$ can be expressed as

$$E(W, v) = X_{\text{error}} + C_1 Y_{\text{error}} + C_2 v_{\text{error}}, \tag{11}$$

where C_1 and C_2 are the weight parameters of Y_{error} and v_{error}, respectively.

Considering that the classification matrix v in the global optimization function $E(W, v)$ is affected by the transformation matrix W, it is divided into two steps when using the gradient to solve the minimum value of the function: (1) treat the classification matrix v as a constant to solve the transformation matrix W; (2) substitute the transformation matrix W into the solution classification matrix v. Then, according to the relevant definitions of X_{error}, Y_{error}, and v_{error}, the gradient of the function $E(W, v)$ to W is $\nabla E(W, v)_W$, and $\nabla E(W, v)_W = 0$, the transformation matrix can be solved W', and for $\forall \theta \in T\_1 (T\_1 = \{X | X \in \mathbb{R}^{o \times m}, X \neq W'\})$, all have $E(\theta, v) \geq E(W', v)$, so W' is such that $E(W, v)$ is a minor local optimal transformation matrix; in the same way, substituting W' into $E(W, v)$, the gradient of $E(W', v)$ to v is $\nabla E(W', v) v$. Let $\nabla E(W', v)_v = 0$; the classification matrix v' can be obtained, and for $\forall \varpi \in T_2 (T_2 = \{X | X \in \mathbb{R}^{o \times m}, v'\})$, all have $E(W', \varpi) \geq E(W', v')$, so v' is the local optimal classification matrix that minimizes $E(W', v)$.

According to Definition 8 and its solution process, the local optimal transformation matrix W' and the classification matrix v' can be obtained. To obtain the global optimal transformation matrix $W*$, the obtained local optimal transformation matrix W' and classification matrix v' are substituted into the function $E(W, v)$ to transform the matrix and classification matrix in turn. The class matrix is solved iteratively until it converges, and the returned transformation matrix is the global optimal transformation matrix $W*$. Finally, substituting the global optimal transformation matrix $W*$ into Equation (5) can obtain the feature matrix $X* = [x_1, x_2 \cdots, x_o] \in \mathbb{R}^{n \times o}$ after feature extraction, which is merged with the label Y by column; namely, the dimensionality reduction dataset DB $*$ can be obtained. The execution process of feature extraction is as follows.

3.4. Feature Grouping

3.4. Feature Grouping. In the current parallel random forest algorithm in the big data environment, training features are formed by randomly selecting the features of the dataset. Although the DRIGFN strategy reduces the redundant and irrelevant features in the dataset through data dimensionality reduction [24], there are still many low-informative features, and due to their existence, the resulting training features are low-informative. Therefore, a feature grouping strategy FGSIT based on information theory is proposed to solve the above problems. This strategy first uses the relevant knowledge of information theory to measure the influence degree of feature-label and feature-feature; secondly, based on obtaining the influence degree of feature-label and feature-feature, the feature evaluation function is proposed above; finally, the features are divided into two groups in an iterative manner. The specific process of feature grouping is described as follows:

(1) The degree of influence between feature-label and feature-feature

It is known that feature x is any feature in feature matrix $X*$ and Y is the label corresponding to the feature matrix $X*$; according to Definition 1, the information gain $IG(xi; Y)$ of the feature x is obtained as follows:

$$IG(x_i; Y) = H(Y) - H(Y|x_i) \quad (1 \le i \le 0). \qquad (12)$$

However, the information gain only measures the influence between features and labels, ignoring the impact between components and parts. Considering the effect of candidate features on selected elements in feature grouping, a calculation feature-feature relationship is proposed. The function $WR(i)$ of the degree of influence is defined as follows:

Definition 9 (feature-feature influence function $\mathrm{WR}(i)$). Y is the label, Q is the selected feature set, and x_j is the element in Q. According to Definition 2 and Definition 3, the candidate features x_i is the feature in Q. Therefore, the influence degree $\mathrm{WR}(i)$ can be expressed as

$$\mathrm{WR}(i) = \sum_{x_j \in Q} \left[I\left(x_j; Y | x_j\right) - I\left(x_j; Y\right) \right] (1 \le i \le 0). \qquad (13)$$

According to Definition 2 and Definition 3, the mutual information $I(x_j; Y)$ represents the correlation between the selected feature x_j and the label Y, and the conditional mutual information $I(x_j; Y \mid x_i)$ represents the feature under the condition of the feature x_i. There is a correlation between x_j and label Y, so the difference between conditional mutual information $I(x_j; Y \mid x_i)$ and mutual information $I(x_j; Y)$ can represent the influence of feature x_i on feature x_i and label Y, so in $\mathrm{WR}(i)$ in the function, the sum of the impact of feature x_i on all features in Q can be used to express the overall influence degree of feature x_j on Q.

(2) Feature evaluation function

To take into account the degree of influence between feature-label and feature-feature in the process of feature grouping, combined with the above two points, a feature evaluation function $\mathrm{FE}(i)$ is proposed, and its definition is as follows.

Definition 10 (feature evaluation function $\mathrm{FE}(i)$). Given candidate features x_i, label vector Y, and selected feature set Q, the evaluation function $\mathrm{FE}(i)$ about feature x_i can be expressed as

$$\mathrm{FE}(i) = \mathrm{WR}(i) + C \bullet \mathrm{IG}(x_i; Y), \qquad (14)$$

where C is the weight parameter of the function $\mathrm{IG}(x_i; Y)$.

Because the information gain can be used to measure the influence between features and labels and the function $\mathrm{WR}(i)$ can measure the impact between components and parts, the feature evaluation function $\mathrm{FE}(i)$ is obtained by combining Equations (12) and (13) simultaneously. Measure the degree of influence between feature-label and feature-feature.

(3) Feature grouping

The feature evaluation function $\mathrm{FE}(i)$ proposed by Definition 10 can divide the feature grouping process into three steps:

© Put the feature with the most significant information gain value in $X*$ into Q

© Calculate the $\mathrm{FE}(i)$ value of the candidate features in turn, and put the feature corresponding to the maximum value of $\mathrm{FE}(i)$ into Q

© Execute the step © iteratively until the number of features in Q reaches the threshold Thr, and the remaining elements form a part set S by themselves

According to the nature of the random forest, it can be inferred that the classification effect of a random forest is related to the correlation between decision trees in the forest and the classification ability of each decision tree. The stronger the correlation, the worse the classification effect of the random forest: decision tree classification. The stronger the command, the better the random forest classification effect. However, the choice of the threshold Thr affects the grouping of features, which involves the correlation of the decision tree and the classification ability of the decision tree. Therefore, the selection of the threshold Thr is essential, so the threshold function $\mathrm{Thr}(I)$ is proposed to determine the threshold, and its definition is as follows.

Definition 11 (threshold function $\mathrm{Thr}(I)$). Assuming that there are T decision trees in the random forest, Q contains I features, and S has U features and randomly extracts features from Q as high-information features in proportion, randomly removing b features from S to combine with an as a construction decision the training feature of the tree, the threshold function $\mathrm{Thr}(I)$ can be expressed as

$$\mathrm{Thr}(I) = \mathrm{acc} - \mathrm{cor} = \left(\frac{a}{a+b}\right)^T - \left(1 - \frac{C_I^a C_U^b C_{I-a}^a C_{U-b}^b}{C_{I+U}^{a+b} C_{I+U}^{a+b}}\right)^{T/2}, \qquad (15)$$

where acc is to use the proportion of features in Q to reflect the overall classification ability of the decision tree in the random forest and cor is to use the similarity of the selected components of the two decision trees to reflect the correlation of the decision trees in the random forest.

According to Definition 11, acc can be used to measure the overall classification ability of decision trees in a random forest, and cor can be used to measure the correlation between decision trees in a random forest. Therefore, according to the nature of the random forest, the classification effect of random forest can be measured by acc-cor. By observing the formula, it can be found that when all the features belong to Q, $a/(a+b) = 1$, then $\mathrm{acc} = 1$ and $\mathrm{cor} \approx 0$, and the maximum value can be taken, but the meaning of grouping is lost at this time, so it is discarded; when $a/(a+b) < 1$, since $T \gg T/2$ in the big data environment, so $\mathrm{acc} \approx 0$. When $I = U$ and $a = b$, that is, $\mathrm{Thr} = (U + I)/2$, the value of cor is the smallest. Acc-cor can reach the maximum value.

The execution of the FGSIT policy is as follows.

3.5. Building Random Forests in Parallel. After data dimensionality reduction and feature grouping, the classifier

```
Input: dataset DB $*$, weight parameter $C$.
Output: feature subset $Q$, $S$.
1. For each feature $x_i$ in $X*$ do//$1 \le i \le o$
2. Calculate the $\mathrm{IG}(x_i ; Y)$ by Equation (1)
3. B $\longleftarrow \mathrm{IG}(x_i ; Y)$//put the calculated information gained into set $B$
4. End for
5. $Q \longleftarrow$ the feature $x_{\mathrm{IG}}$ corresponding to the maximum value in B//put the feature with the largest information gain into the set $Q$
6. Remove $x_{IG}$ from $X*$
7. While the length of $X*$ >o/2 do
8. For each feature $x_j$ in $X*$ do//$1 \le j \le o-1$
9. Calculate the $\mathrm{FE}(xj)$ by Equation (14)
10. $C \longleftarrow \mathrm{FE}(x_j)$//put the calculated $\mathrm{FE}(x_j)$ into the set C
11. End for
12. $Q \longleftarrow$ the feature $x_{\mathrm{FE}}$ corresponding to the maximum mum value in $C$
13. Remove $x_{\mathrm{IG}}$ from $X*$
14. End while
15. S $\longleftarrow$ the remaining features in $X*$
16. Return $Q$, $S$
```

Algorithm 3: FGSIT strategy.

needs to be parallelized and trained according to the reduced dimensionality dataset DB $*$ and feature sets Q and S. At present, the parallel random forest algorithm in the big data environment mainly builds multiple decision trees based on the training data and training features as the output results.

And on this basis, the samples are predicted to obtain the model accuracy. However, in the prediction stage of this method, due to the different decision trees in each computing node, the predicted key-value pairs obtained for the dataset are also other, so after merging, the number of key-value teams on each Mapper node will be different. If there is a difference, it usually leads to an unbalanced load on the Reducer nodes in the next stage, which affects the parallelization efficiency of the algorithm. To deal with the above problems, this section first proposes the RSKP strategy to optimize the MapReduce computing model and balance the load of the Reducer nodes; then, it uses the optimized MapReduce model to build a random forest in parallel, predict the classification of the dataset, and obtain the accuracy of the random forest. Build random forests in parallel.

The specific process is described as follows:

(1) RSKP strategy

Given the set of key-value pairs P_1, P_2,..., P_q obtained after merging in each Mapper node, the process description of the RSKP strategy is shown in Figure 1.

(a) Aggregate all key-value pairs P_1, P_2,…, P_q into the intermediate file and sort them according to the keys in the key-value pairs

(b) According to the number of key-value pairs and the number of Reducer nodes, distribute the key-value teams in the intermediate folder to each node

(2) Parallel construction of random forest and prediction of dataset classification. The optimized MapReduce model is obtained through the RSKP strategy. Combined with this model, the parallel structure of the random forest is divided into four steps, as shown in Figure 2

(a) Call Hadoop's default data block strategy, divide the dataset DB $*$ into blocks of the same size, and transmit them to the Mapper node as input data

(b) According to the task assigned to each Mapper node by the primary node, call the Map function to extract the training set of the decision tree through bootstrap autonomous sampling and randomly remove features from the feature subsets Q and S as training features in proportion; based on the training set and the training feature, construct a decision tree in the form of a key-value pair $<\mathrm{key}_T, \mathrm{value}_T>$ (key_T is the decison tree model number; value_T is the decision tree model), all Mapper nodes are executed, and all decision trees are parsed and merged to obtain a random forest model

(c) Use the decision tree in the Mapper node to predict the dataset DB $*$ and form a new key-value pair $<\mathrm{key}', \mathrm{value}'>$ (key' is the combination array of the sample ID and the corresponding category; $\mathrm{value}' =$ 1 means the critical value number of occurrences of the pair); merge key-value teams with the same key' value (such as there are three key-value pairs $<k:1>$, $<k:1>$, and $<k:1>$ with all $\mathrm{key}'k$ locally; then, they will be merged into one key-value pair $<k:3>$)

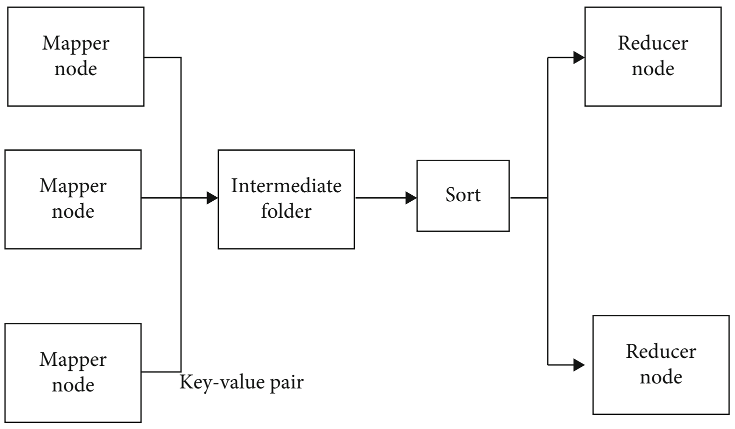

Figure 1: RSKP strategy to optimize the MapReduce computing model.

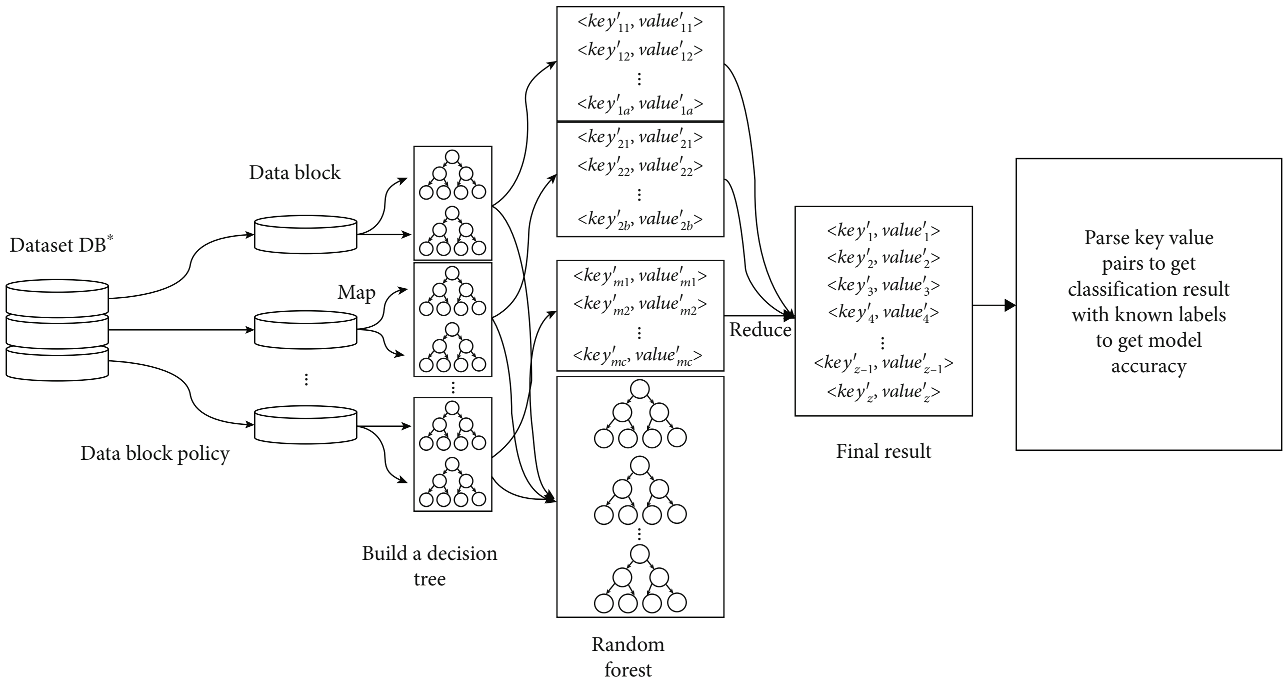

Figure 2: Random forests in parallel.

(d) The key-value pairs predicted in the Mapper node are distributed by the master node and then transferred to the corresponding Reducer nodes to merge again to obtain the global classification result and compare it with the label Y to get the model's accuracy. So far, the execution process of building a random forest in parallel is as follows

3.6. PRFITN Algorithm Steps. The specific implementation steps of the PRFITN algorithm are as follows.

Step 1. Divide the original dataset into file blocks of the same size through the default file block strategy of Hadoop, call a MapReduce task to calculate the information gain of the original data features in parallel, and select the features on this basis.

Step 2. Invoke the FEKFN strategy to extract new features from the feature-selected dataset in an iterative manner.

Step 3. Call the FGSIT strategy to group the features of the reduced dimensionality dataset.

Step 4. Start a new MapReduce task, call the Map function, use bootstrap and stratified sampling to extract training samples and features used for modeling, build a decision tree, and aggregate all decision trees to obtain a random forest; use the RSKP strategy to distribute Reducer node tasks

```
Input: dataset DB $*$, feature subset $Q$, $S$.
Output: random forest model and its accuracy.
Map stage
1. For each block corresponding to each Mapper node, do
2. T $\longleftarrow$ select training set randomly
3. F $\longleftarrow$ select training features randomly key$_T$ $\longleftarrow$ number the decision tree trained by $T$ and $F$value$_T$ $\longleftarrow$ train decision tree based
on $T$ and $F$
6. End for
7. Model $\longleftarrow$ collection of all decision trees
8. Return model//output random forest, model
9. For each decision tree in each Mapper node, do
10. Predict the category of each sample from the DB $*$
11. Key$'$ $\longleftarrow$ combine sample ID with sample category
12. Value$'$ $\longleftarrow$ number of key-value pairs
13. Output <key$'$, value$'$ >
14. End for
15. Combine key-value pairs with the same key$'$ in each Mapper
16. AK $\longleftarrow$ collection of all key-value pairs reduce stage
1. Sorted (AK)//sort all key-value pairs
2. RSKP(AK)//evenly distribute key-value pairs
3. Obtain the global classification results by combining key-value pairs with the same key$'$
4. Accuracy $\longleftarrow$ compare label $Y$ with global classification results//get classification accuracy
5. Return accuracy
```

Algorithm 4: Build random forest in parallel.

evenly, call the Reduce function to get the global classification results, and evaluate the model classification accuracy.

3.7. Analysis of Algorithm Time Complexity. The PRFITN algorithm mainly includes three stages: data dimensionality reduction, feature grouping, and parallel construction of random forests. Therefore, the algorithm's time complexity is primarily composed of three parts, denoted as T1, T2, and T3, respectively.

In the feature selection stage of data dimensionality reduction, the time complexity mainly depends on calculating the information gain value of each feature, which needs to traverse each data under the sample corresponding to each element in the dataset. Given that the number of samples in the known dataset is n, the number of features is d, and the number of Mapper nodes corresponding to executing MapReduce tasks is m. The time complexity of this stage is

$$T_s = O\left(\frac{n \times d}{m}\right). \tag{16}$$

In the feature extraction stage of data dimensionality reduction, the time complexity mainly depends on the process of iteratively optimizing the transformation matrix W and the classification matrix v. It is known that W is a matrix of order $n \times o$, and v is a matrix of order $n \times 1$. Therefore, it is assumed that this stage requires iteration. Calculate k times, and then, the time complexity is

$$T_E = O(k \times (n \times o + n)) \approx O(k \times n \times o). \tag{17}$$

Therefore, the time complexity of data preprocessing is

$$T_1 = O\left(k \times n \times o + \frac{n \times d}{m}\right). \tag{18}$$

The FGSIT strategy is mainly used in the feature grouping stage to divide the features. The feature evaluation function FE(i) between each candidate feature and the selected feature needs to be calculated each time the quality is screened. Knowing that the number of processed features is o and the number of samples is n, the time complexity of this stage is

$$T_2 = O\left(o \times n^2\right). \tag{19}$$

In the parallel construction of the random forest, the MapReduce task is mainly called to build the random forest model in parallel and predict all data classification to evaluate the accuracy. Assuming that the random forest model contains F decision trees, the number of Mapper nodes corresponding to the MapReduce task is M, and the number of Reduce nodes is R, the time complexity of this stage is

$$\begin{aligned} T_3 &= O\left(\frac{F \times n \times o \times lbn}{M} + \frac{F \times n \times o \times lbn}{R}\right) \\ &\approx O\left(\frac{F \times n \times o \times lbn}{M \times R}\right), \end{aligned} \tag{20}$$

$$T_{\text{PRFITN}} = O(T_1 + T_2 + T_3). \tag{21}$$

Table 1: Configuration of nodes in experiment.

CPU name	IP address	Role
Master	192.168.1.108	Master/JobTracker/NameNode
Slaver_1	192.168.1.109	Slaver/TaskTracker/DateNode
Slaver_2	192.168.1.110	Slaver/TaskTracker/DateNode
Slaver_3	192.168.1.111	Slaver/TaskTracker/DateNode

Table 2: PRFITN algorithm of the experimental datasets.

Dataset	Number of samples/bars	Number of attributes/species	Size (MB)
Farm Ads	1692088	5267660	1482
Susy	990005	41280	33
APS Failure at Scania Trucks	5000000	195	325

4. Experimental Results and Comparison

4.1. Experimental Environment. To verify the performance of the PRFITN algorithm, related experiments are designed in this paper. The experimentation includes four computing nodes in terms of hardware, including 1 Master node and 3 Slaver nodes. The CPUs of all nodes are AMD Ryzen 7, each with eight processing units and 16 GB of memory. The four nodes in the experimental environment are in the same local area network and connected by 200 Mbit/s Ethernet. In terms of software, the Hadoop version installed on each node is 2.7.4, the Java version is 1.8.0, and the operating system is Ubuntu 16.04. The specific configuration of each node is shown in Table 1.

4.2. Experimental Data. The experimental data used by the PRFITN algorithm are three real datasets from the UCI public database (https://archive.ics.uci.edu/ml/index.php), namely, Farm Ads, Susy, and APS Failure at Scania Trucks. The Farm Ads dataset is various farm animal-related data collected from text ads on 12 websites.

The dataset contains 4143 samples and 54877 attributes, which have a small sample size and many features. Susy is a dataset that records the detection of particles using particle accelerator dataset containing 5000000 records. There are 18 attributes in total, which have the characteristics of a large sample size and a small number of features: APS Failure at Scania Trucks dataset. It is a dataset that records Scania truck APS faults and operations. The dataset contains 60,000 samples and a total of 171 attributes. It has the characteristics of moderate sample size and a moderate number of features. The specific information of the dataset is shown in Table 2.

4.3. Performance Analysis of PRFITN Algorithm. To verify the feasibility of the PRFITN algorithm in the big data environment, this paper selects 50, 100, and 150 decision trees in the random forest. It applies the PRFITN algorithm to the three datasets of Farm Ads, Susy, and APS Failure at Scania Trucks, runs ten times independently, takes the average of the ten running results, and compares the running time and accuracy of the algorithm to achieve an overall evaluation of the performance of the PRFITN algorithm. Figure 3 shows the execution results of the PRFITN algorithm under three datasets.

As can be seen from Figure 3 and Table 3, when the number of decision trees changed from 50 to 100 to 150, the running time of the algorithm on the Farm Ads dataset increased by 8700 s and 9000 s, and the accuracy increased by 3.8 percentage points and 3.8 percentage points, respectively, 1.5 percentage points; running time on the Susy dataset increased by 4250 s and 6000 s, and accuracy increased by 2.5 percentage points and decreased by 0.7 percentage points; runtime on the APS Failure at Scania Trucks dataset increased by, respectively, 750 s and 4500 s, and the accuracy increases by 3.0 and 1.1 percentage points, respectively. It can be seen from the data reflected in the picture that the time complexity and accuracy of the PRFITN algorithm on the three datasets gradually increase, and the increase of time complexity gradually increases, but the increase of accuracy decreases slowly. The former is mainly due to the increase in the number of tasks assigned to the computing nodes in the modeling stage as the number of decision trees increases, and the number of key-value pairs also increases exponentially, so it takes more time to process them. The main reason is that with the increase of the number of decision trees, the difference between trees will decrease, and then, the impact on the classification results of the random forest will become smaller and smaller, so the increase in accuracy will increase with decrease as the decision tree increases.

4.3.1. Time Complexity Comparison of PRFITN Algorithm. This study conducts tests based on the three datasets of Farm Ads, Susy, and APS Failure at Scania Trucks to confirm the temporal complexity of the PRFITN method. The PRFMIC algorithm carries out a thorough comparison. The PRFITN algorithm without the RSKP strategy—referred to as PRFITN-ER—is also run in order to investigate the effects of load balancing on the PRFITN algorithm. The specific time complexity is shown in Figure 4 and Table 4.

As can be seen from Figure 5, on the Farm Ads dataset, the running time of the PRFITN algorithm is 2300 s,

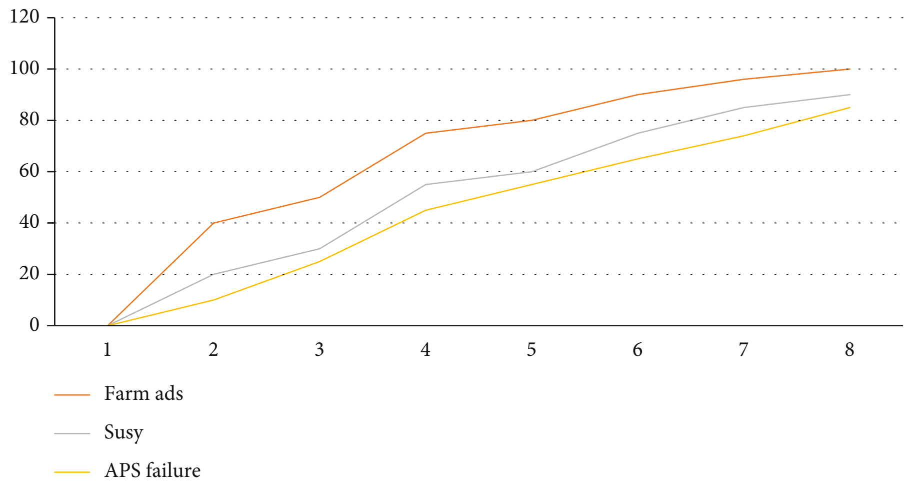

Figure 3: Number of DT build up by PRFITN algorithm.

Table 3: Execution results of the PRFITN algorithm under three datasets.

Running time/10^3	Number of decision tree Farm Ads	Susy	APS Failure
0	0	0	0
5	40	20	10
10	50	30	25
15	75	55	45
20	80	60	55
25	90	75	65
30	96	85	74
35	100	90	85

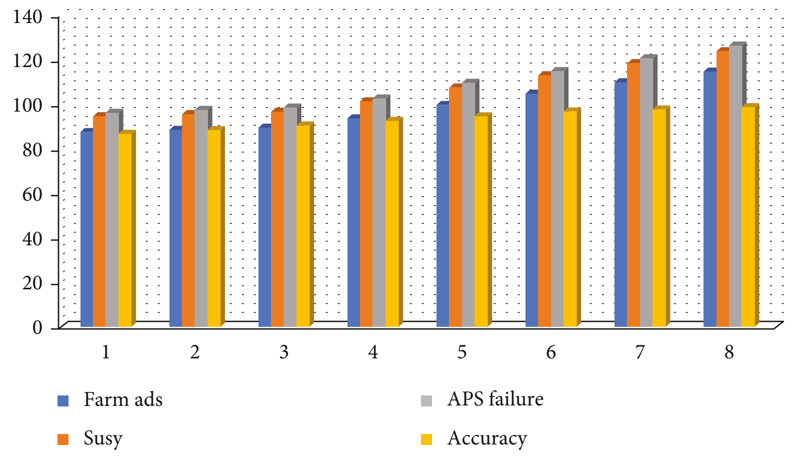

Figure 4: Graph for accuracy of PRFITN algorithm.

3833.3 s, and 8666.7 s higher than that of the PRFMIC algorithm, the PRF algorithm, and the improved MR_RF algorithm, respectively. On the APS Failure at Scania Trucks dataset, the running time of the PRFITN algorithm is on average 200 s, 416.7 s, and 733.3 s higher than that of the PRFMIC algorithm, the PRF algorithm, and the improved MR_RF algorithm, respectively. The above two situations occur because the PRF algorithm uses data dimensionality reduction processing for training features when building the random forest model. On the other hand, the PRFMIC algorithm assumes hierarchical processing for parts, and the PRFITN algorithm adopts dimensionality reduction

Table 4: Accuracy of PRFITN algorithm.

Accuracy %	Number of decision tree		
	Farm Ads	Susy	APS Failure
87	88	95.04	96.8
89	89	96.12	97.9
91	90	97.2	99
93	94	101.52	103.4
95	100	108	110
97	105	113.4	115.5
98	110	118.8	121
99	115	124.2	126.5

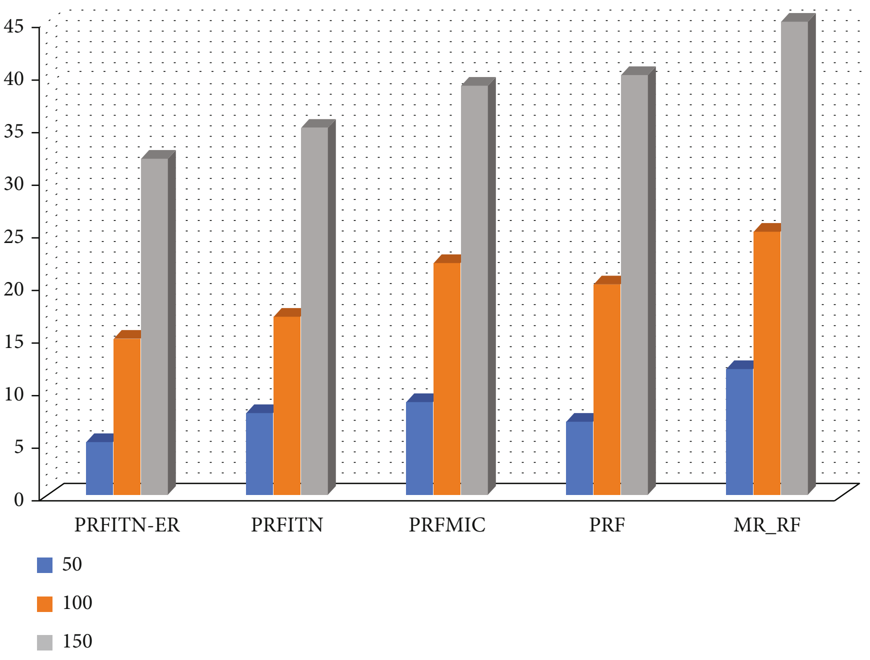

Figure 5: Running time of five algorithms on different scenarios.

and feature layering for details. In addition, the dimensionality reduction and layering strategy of the PRFITN algorithm focus on directly evaluating the elements themselves. Therefore, when dealing with the Farm Ads and APS Failure at Scania Trucks datasets with relatively many features, the PRFITN algorithm is significantly better than PRFMIC, PRF, and the improved MR_RF. However, the algorithm takes a lot of time.

On the contrary, when dealing with the Susy dataset with large sample size and a small number of features, the running time of the PRFITN algorithm is 6783.4 s and 3750 slower than that of the PRFMIC algorithm and the PRF algorithm, respectively. When the number of features is small, the PRFITN algorithm takes less time in the data dimensionality reduction and feature layering stages. The PRFMIC algorithm uses the RSKP strategy to balance the load of each node and reduce the time complexity. In addition, to more intuitively judge the impact of load balancing on the model, that is, the optimization effect of the RSKP strategy on the model, comparing the running time of the PRFITN algorithm and the PRFITN-ER algorithm on the three datasets, it can be seen that in the Farm Ads data On the dataset, Susy dataset, and APS Failure at Scania Trucks dataset, the running time of the PRFITN algorithm is 1733.33 s, 1583.33 s, and 295 s more minor than that of the PRFITN-ER algorithm on average, so the adoption of the RSKP strategy will save to a particular extent model learning time.

5. Conclusion

To solve the shortcomings of the parallel random forest algorithm in the big data environment, this paper proposes a parallel random forest algorithm PRFITN based on information theory and norms. Parallel Random Forest (PRF) method using Spark to boost the effectiveness of the RF technique and alleviate the data connection cost and workload imbalances concerns of massive data in a distributed and parallel environment. The PRF method is optimized using a hybrid parallel technique that combines data and job parallelization. A vertical data-partitioning approach and a data-multiplexing method are used for data-parallel optimization. These strategies lower the amounts of data and the frequency of data transfer operations in a distributed setting while maintaining algorithm correctness. These strategies lower the amounts of data and the frequency of data transfer operations in a distributed setting while maintaining algorithm correctness. First, the algorithm fully considers the problem of redundant and irrelevant features in large datasets and proposes a hybrid dimensionality reduction strategy, DRIGFN. This strategy can effectively reduce the dimension of the dataset and significantly reduce the amount of information lost during data dimension reduction. Secondly, to improve knowledge of the features used for training decision trees in random forests, a feature grouping strategy FGSIT is proposed, which fully considers the relationship between feature-feature and feature-label. The weighted vote technique and dimension reduction are used to optimize the PRF algorithm's accuracy. Then, using Apache Spark, a hybrid parallel PRF technique incorporating data-parallel and task-parallel optimization is carried out. The training dataset is reused, and the amount of data is greatly decreased, thanks to the data-parallel optimization. The task-parallel optimization has the advantage of significantly lowering the cost of data transmission and enhancing algorithm performance. According to experimental findings, PRF is superior to other algorithms and has distinguishing advantages over them in terms of accuracy, efficiency, and scalability. On this basis, the features are divided into two groups; the training features are extracted proportionally, ensuring the information amount of the selected features when constructing the decision tree. Finally, considering the impact of cluster load on the efficiency of parallel algorithms, a key-value pair redistribution strategy RSKP is designed to evenly group the intermediate results obtained by the similar algorithm, balancing a load of reducer nodes in the cluster and reducing the time complexity of the algorithm Spend. At the same time, to verify the classification performance of the PRFITN algorithm, this paper compares and analyzes the three algorithms of the improved MR_RF algorithm, the PRF algorithm, and the PRFMIC algorithm on the three datasets of Farm Ads, Susy, and APS Failure at Scania Trucks. The experimental results show that the PRFITN algorithm has high accuracy in the big data environment, especially for classifying datasets with a large number of features.

6. Future Scope

The authors explain the topics that warrant additional investigation and expect that these issues may contain the potential to contribute to future research studies, building on the solid foundation of the research findings reported and the general knowledge attained in this work. Due to the majority of the studied publications adopting an analytical methodology, it is possible to enhance empirical research based on an extensive case study using a qualitative and quantitative approach based on surveys. As a cross-cutting theme, business and management big data have numerous ties to well-established subjects in the fields of computing, engineering, mathematics, business, and social sciences, among others.

Data Availability

The data shall be made available on request.

Conflicts of Interest

The authors declare that they have no conflict of interest.

References

[1] R. Mittal, V. Malik, V. Singh, J. Singh, and A. Kaur, "Integrating genetic algorithm with random forest for improving the classification performance of web log data,," in *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pp. 177–181, Waknaghat, India, 2020.

[2] J. Chen, K. Li, Z. Tang et al., "A parallel random forest algorithm for big data in a spark cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 919–933, 2017.

[3] M. C. M. Oo and T. Thein, "Hyperparameters optimization in scalable random forest for big data analytics," in *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, pp. 125–129, Singapore, 2019.

[4] Y. Xu, "Research and implementation of improved random forest algorithm based on Spark," in *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pp. 499–503, Beijing, China, 2017.

[5] S. Wang, J. Zhai, S. Zhang, and H. Zhu, "An ordinal random forest and its parallel implementation with MapReduce," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2170–2173, Hong Kong, China, 2015.

[6] H. Chen, P. Chang, Z. Hu, H. Fu, and L. Yan, "A Spark-based Ant Lion algorithm for parameters optimization of random forest in credit classification," in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 992–996, Chengdu, China, 2019.

[7] G. Cong and I. Tanase, "Composable locality optimizations for accelerating parallel forest computations," in *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 190–197, Sydney, NSW, Australia, 2016.

[8] A. Oussous, F.-Z. Benjelloun, A. Ait Lahcen, and S. Belfkih, "Big data technologies: a survey," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 4, pp. 431–448, 2018.

[9] A. Hamad, S. N. Saleh, and A. Abouelfarag, "Improving performance of distributed data mining," in *2017 27th International Conference on Computer Theory and Applications (ICCTA)*, pp. 111–116, Alexandria, Egypt, 2017.

[10] M. P. Lokhande, D. D. Patil, L. V. Patil, and M. Shabaz, "Machine-to-machine communication for device identification and classification in secure telerobotics surgery," *Security and Communication Networks*, C. Chakraborty, Ed., vol. 2021, Article ID 5287514, 16 pages, 2021.

[11] P. L. Springer, T. Schibler, G. Krawezik, J. Lightholder, and P. M. Kogge, "Machine learning algorithm performance on the Lucata computer," in *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, pp. 1–7, Waltham, MA, USA, 2020.

[12] V. Jagota, M. Luthra, J. Bhola, A. Sharma, and M. Shabaz, "A secure energy-aware game theory (SEGaT) mechanism for coordination in WSANs," *International journal of swarm intelligence research*, vol. 13, no. 2, pp. 1–16, 2022.

[13] W. Lin, Z. Wu, L. Lin, A. Wen, and J. Li, "An ensemble random forest algorithm for insurance big data analysis," *IEEE Access*, vol. 5, pp. 16568–16575, 2017.

[14] D. Jeon and W. Kim, "Random forest algorithm for linked data using a parallel processing environment," *IEICE Transactions on Information and Systems*, vol. E98.D, no. 2, pp. 372–380, 2015.

[15] D. A. Bader and G. Cong, "Fast shared-memory algorithms for computing the minimum spanning forest of sparse graphs," in *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings*, p. 39, Santa Fe, NM, USA, 2004.

[16] D. Chen, N. Zhang, Z. Qin et al., "S2m: a lightweight acoustic fingerprints-based wireless device authentication protocol," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 88–100, 2017.

[17] S. Sanober, I. Alam, S. Pande et al., "An enhanced secure deep learning algorithm for fraud detection in wireless communication," in *Wireless Communications and Mobile Computing*, V. Shanmuganathan, Ed., vol. 2021, Article ID 6079582, 14 pages, 2021.

[18] F. Hossain, M. Akter, and M. N. Uddin, "Cyber attack detection model (CADM) based on machine learning approach," in *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, pp. 567–572, Dhaka, Bangladesh, 2021.

[19] M. Kang, S. K. Gonugondla, S. Lim, and N. R. Shanbhag, "A 19.4-nJ/decision, 364-K decisions/s, in-memory random forest multi-class inference accelerator," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 7, pp. 2126–2135, 2018.

[20] C. Li, H. Niu, M. Shabaz, and K. Kajal, "Design and implementation of intelligent monitoring system for platform security gate based on wireless communication technology using ML," *International Journal of System Assurance Engineering and Management*, vol. 13, Supplement 1, pp. 298–304, 2022.

[21] S. Horng, X. Hu, B. Li, and N. Xiong, "Personal identification via heartbeat signal," in *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, pp. 152–156, Taipei, Taiwan, 2018.

[22] B. Xin, W. Yang, S. Wang, and L. Huang, "Differentially private greedy decision forest," in *ICASSP 2019-2019 IEEE international conference on acoustics, Speech and Signal Processing (ICASSP)*, pp. 2672–2676, Brighton, UK, 2019.

[23] E.-H. Yeh, P. Lin, X.-X. Lin, J.-Y. Jeng, and Y. Fang, "System error prediction for business support systems in telecommunications networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2723–2733, 2020.

[24] U. Sivarajah, M. M. Kamal, Z. Irani, and V. Weerakkody, "Critical analysis of big data challenges and analytical methods," *Journal of Business Research*, vol. 70, pp. 263–286, 2017.