An Efficient Method for Big Data Classification

S. Gayathri Devi and M. Sabrigiriraj

Department of Information Technology, Coimbatore Institute of Engineering and Technology, Coimbatore, 641109 Tamil Nadu, India

Department of Electronics and Communication Engineering, SVS College of Engineering, Coimbatore, 642109 Tamil Nadu, India

Abstract: Big data classification is an important process that helps in efficient analysis of larger datasets. The designing of highly parallelized learning algorithms provides efficient big data classification which can be done by applying parallel computation on Extreme Learning Machine Tree (ELM-Tree) model. In the ELM-tree Model, the decision tree nodes are split based on uncertainty measures such as information entropy and ambiguity. The over partitioning problem caused due to compression of data to a fixed amount of memory and high computation time due to more iterations are needed to be sorted. The problem of over partitioning can be overcome by embedding ELMs as the leaf nodes when the gain ratios of all the available splits are below a given threshold. The input weights in ELM are assigned randomly in order to approximate the training instances but this approach can further be improved by optimizing the weights that can be achieved only by identifying the optimal cut-points for each attribute. Similarly, the calculation of information gain and gain ratio of all attributes and their cut points increases the computation time. This can be reduced by optimally scheduling the computation tasks to the available host nodes so that the computation of the information gain and the gain ratio takes less time. In this paper, efficient optimization algorithms are first utilized for optimizing the cut-points for each attribute that helps in determining the optimal weights of the attributes. The genetic algorithm, Particle Swarm Optimization (PSO) and firefly optimization algorithms are used in this approach to determine the optimal cut-points. The available cut-points of an attribute are randomly assigned to the efficient host nodes using the optimal scheduling algorithm and the cut-point with best gain values is chosen as the optimal cut-point. The scheduling algorithm is called in between for the efficient scheduling of the cut-point gain computation tasks. The computation of the information gain and gain ratio of the attributes takes more time which can be reduced by the efficient scheduling. The task scheduling algorithm utilizes the optimization algorithms to decide which node performs the task efficiently and allocates the task randomly to the nodes. This strategy does not send all the data to the host nodes instead sends the randomly selected data to the nodes which reduces the overall iteration of tasks thus reducing the computation time. Experimental results also show that the presented technique effectively selects the optimal cut-points and also reduces the computation time.

Key words: ELM tree, genetic algorithm, particle swarm optimization, firefly algorithm, computation tasks

INTRODUCTION

Big data is the collection of data in huge volumes that provides more than sufficient informations related to the field (Slavakis et al., 2014). Big data has caused the most of the developing industries to use advanced techniques to analyze the large size datasets. There are three challenging problems with big data causing researchers to find solutions. The first problem is the velocity problem that creates huge amounts of data to be handled at a high speed. The next problem is the variance in data as the data are collected from different sources and are formatted differently. The last problem is the memory problem that is the major issue among the three as the storage and processing requires large memory. These problems paved way for wider researches to be performed to provide efficient analysis of data.

Extreme Learning Machine (ELM) (Huang et al. (2011)) approach has been presented as an efficient method to analyze the large amount of data. ELM is generally used for the classification and regression of data with a single layer of hidden nodes. The machine learning approach has fast learning speed and best generalization scheme. The approach efficiently solves
the problems but cannot be termed as the best method due to the reason that it cannot be effective for all types of datasets (Huang et al., 2012). The complex datasets requires wide analysis techniques to efficiently determine the attributes. The cloud computing techniques like MapReduce can be used to analyze the big data without any scalability problems. The method uses distributed computing to train multiple models with large data blocks and combine them using ensemble algorithms. As the method seems efficient the big data can be processed by implementing MapReduce based machine learning approaches.

The main objective of this study is the determination of optimal cut-points and the efficient scheduling of computation tasks to different nodes in the ELM tree. An attribute has many cut-points which all are send to host nodes to compute the information gain and gain ratio. This approach takes more time to determine the best possible results of an attribute due to using all the cut-points in the computation process. Hence it is necessary to select the optimal cut-points to reduce the computation time. In this paper, the efficient optimization algorithms, genetic optimization, particle swarm optimization and firefly optimization algorithms are utilized to determine the optimal cut-points. The approach, instead of selecting all cut-points, randomly selects the cut-points and computes the information gain. After some iteration, the cut-points with best information gain can be found by the optimization algorithms. The computation of the information gain and gain ratio takes more time and hence a scheduling algorithm is used to schedule these computation tasks to efficient host nodes by estimating which node can process which tasks with higher efficiency using the optimization algorithms. The node data capacity is considered while allocating the tasks and only few randomly selected data (i.e., attribute with cut-point) are computed. Thus the computation tasks can be performed efficiently with less time and reduced computation complexity.

**Literature review:** In this study, various techniques are discussed for efficient big data classification. Anbulagan and Chandrasekaran (2015) presented Parallel Weighted Decision Tree Classifier for classifying the complex spatial landslide big data in the MapReduce Model. The approach includes the different degrees of importance to the different landslide factors. The three data structures such as attribute table, count table, hash table are used to build parallel decision tree classifier. This improves the classification by selecting a best splitting attribute which has best gain ratio. Divya and Gagandeep (2015) suggested the implementation of artificial intelligence to improve the big data classification. On the basis of pollination based optimization, the big data classification is performed. The approach with unique blend of classification algorithm, parallelism and artificial intelligence, improves the big data classification efficiently.

Grobling et al. (2014) presented an approach of big data classification using MapReduce Model. The research focuses on the challenges involved in the implementation of MapReduce for the big data and introduces the MapReduce based classification model to overcome the problems. The approach improves the parallelism in computing and determines the classes. Rebentrost et al. (2014) suggested the use of quantum support vector machines for the efficient classification of the big data. The approach uses support vector machine in the quantum computer to analyze complexity logarithmic in the size of the vectors and the number of training examples. The efficient evolutionary under sampling techniques by Triguero et al. (2015 ab) has been a greater promise in big data classification. The problem of less number of instances is needed to be sorted and hence the MapReduce scheme is introduced that distributes the functioning of evolutionary under sampling algorithms in a cluster of computing elements.

Tekin and Skaar (2013) suggested a distributed online big data classification method using the context information. In this method, the data is gathered by distributed data sources and processed by a heterogeneous set of distributed learners. To solve the problem of joint classification by the distributed and heterogeneous learners in which data is obtained from multiple sources is taken as a distributed contextual bandit problem and each data is characterized by a specific context. Lopez et al. (2014) presented linguistic fuzzy rule based classification system build on the MapReduce framework for efficient classification of big data. The approach is denoted as Chi-FRBCS-Big data provides a competitive predictive accuracy in big data. Chi-FRBCS-Big data has been developed under two versions Chi-FRBCS-Big data-Max and Chi-FRBCS-Big data-Ave to provide efficient data clustering for different degrees using fuzzy rule based data classification.

Xin et al. (2015) presented the elastic extreme learning machine also known as Elastic ELM or ELM based on the for the effective big data classification. The ELM approach performs better than normal ELM which has weak learning ability for the updating large scale training dataset. Moreover the matrix multiplication part in the ELM is the most computation expensive part. The matrix multiplication problem is resolved in ELM by calculating it by incrementing, decrementing or correctional
calculation. The intermediate matrices are calculated and then the old matrices are compared with them to provide effective learning for the updating large datasets in ELM. The problem with the approach is that the ELM utilized the weight vector for rapid learning which may reduce the convergence speed.

Triguero et al. (2015) proposed a novel distributed partitioning methodology for prototype reduction techniques in nearest neighbor classification. The approach effectively reduces the number of instances and thus increases the speed of the classification which greatly reduces the storage requirements and the noise sensitive measures. In order to reduce prototypes in larger datasets, the MapReduce based framework is introduced with strategies to integrate multiple partial solutions into single solution and thus avoids drop in classification accuracy rates. This approach can be applied to various applications which require conceptual work integration. But the architecture of the big data systems has to suit for integration and hence the reference architecture models for the classification of the products and services has been presented by Paakonon and Pakkalna (2015) for the efficient test case classification in big data.

Sun et al. (2011) presented Online Sequential Extreme Learning Machine (OS-ELM) for the effective classification of the ensemble in the P2P networks. The OS-ELM tree can be easily implemented for distributed ensemble learning and classification. A two-layer index structure is also presented to efficiently support peer selection for the introduction Quad tree for ensemble learning. The drawback in the approach is that it cannot be utilized for high dimensional ensembles. To overcome the uncertainty, Hualong et al. (2015) presented active learning ELM (AL-ELM). The approach can estimates the uncertainty of each unlabeled instance by creating a mapping relation between the actual outputs of the instance in ELM and the approximated membership probability of the same instance.

The clustering performance of the ELM approach can be analyzed by studying the ELM in state-of-the-art methods. Michele et al. (2015) presented an approach for clustering the Self-Organizing Maps (SOM) using the ELM. The approach forms two problems and proposes two clustering methods based on the features of the ELM approach with the a priori knowledge. The optimization of the big data after classification has to be dealt with the two stage query processing optimization (Ding et al. 2015). As optimization is out of scope for our research, the classification is considered which is obtained by ELM classifier. ELM classifier trains the configuration and prediction parameters and classifies the big data.

Guanwu Zhou proposed the use of ELM approach for the classification of the big data and also presented the parallel approach of ELM and MapReduce for the betterment of the big data classification. The approach also enables provision for the implementation of Hadoop MapReduce with ELM through accurate and fast learning. Horta et al. (2015) presented the concept of active learning approach by including data stream based ELM and Hebbian learning for the effective classification of big data using the linearization of the input data in the ELM tree. The problem with the approach is that the focus is on many big data problems so that the classification performance is not satisfactory. In order to solve the problem (Rizk and Awad, 2015), presented unsupervised ELM approach in three forms for the classification. The parallel, hierarchical and ensemble unsupervised ELM has been presented to provide effective classification for all types of ELM of big data. The problem with the approach is that the cluster formation is not efficient due to the weak matrix operations performed in the ELM.

MATERIALS AND METHODS

The ELM approach is an emerging technique for training Single-hidden Layer Feed Forward Neural Networks (SLFNs). Extreme Learning Machine Tree (ELM-Tree) model has extremely fast training speed and hence have the great potential for learning from big data. The parallelized learning algorithm provides efficient big data classification which can be done by applying parallel computation on ELM-Tree Model. In the ELM-tree Model, the decision tree nodes are split based on uncertainty measures such as information entropy and ambiguity. The ELM tree is generated usually by random selection of the input weights and then the analytical determination of output weights. The performance can be improved by input weight optimizing of the ELM nodes. But in order to determine optimal weights, the optimal cut-points are needed to be determined which can be done by using the optimization techniques. The selection of optimal cut-points is based on the computation of information gain and gain ratio. But these computation tasks consume more time and hence a scheduling algorithm comprised of optimization techniques is employed which determines which task to send to which node in a random approach that reduces the overall computation time.

Scheduling computation tasks based on optimization techniques: The computation of different parameters of
attributes in a larger datasets is needed to be distributed to different computing nodes inorder to process the data in less time. The information gain, split ratio and gain ratio calculations are done to estimate the optimal cut points for each attribute. These calculations are performed in the host computing nodes by allocating all the data associated with each cut-point to the available nodes and determining the better results. This approach consumes more time as all the cut-points are analyzed and also the fact that only few cut-points will be optimal. Hence, in our approach, the available nodes are analyzed with the available tasks using the optimization algorithms. The node to which a task is allocated can perform better would have better data capacity. Thus the nodes with better data capacity are optimally selected and the tasks are allocated randomly to the nodes. The random selection is such that not all the cut-point computations are carried out and only few cut-points are computed. This reduces the number of iterations of computation.

The scheduling process by determining the optimal nodes for each specific task are allocated by using the optimization techniques. The genetic algorithm, Particle Swarm Optimization (PSO) and firefly algorithms are employed to optimize which node should process which task based on the data capacity of the nodes. The genetic algorithm assigns the population of solutions and selects the nodes with high data rate using cross over and mutation of the chromosomes. The nodes with high data rate are assigned with larger tasks or even more than one task can be assigned to it. Similarly, the nodes with low data handling are assigned with suitable tasks.

A set of computers N is taken as computing nodes. The tasks to be assigned are also taken as T. The information gain of attribute A and its i-th cut point is calculated. Here i = 1, 2, ..., N-1 and j = 1, 2, ..., n.

\[
\text{Gain}(X, \text{cut}_i) = \text{Info}(X) - I(X, \text{cut}_i)
\]  \hspace{1cm} (1)

The information gain calculation \{Gain_{x_1}, Gain_{x_2}, \ldots, Gain_{x_n}\} are assigned to the N computing nodes. The results are stored in a host node when the calculations are over. Then the remaining calculations \{Gain_{y_{1x_1}}, Gain_{y_{2x_2}}, \ldots, Gain_{y_{nx_n}}\} are assigned to free computing nodes in the next iteration. Then the calculation of gain ratio, Ratio, is also done for each attribute to class them:

\[
\text{Ratio}(X, A_j) = \frac{\text{Gain}(X, \text{cut}_i)}{\text{Split}(X, \text{cut}_i)}
\]  \hspace{1cm} (2)

Where:

\[
\text{Split}(X, \text{cut}_i) = - \left( \frac{\log_2 \left( \frac{|X_{x_1}|}{|X|} \right)}{|X|} + \frac{\log_2 \left( \frac{|X_{x_2}|}{|X|} \right)}{|X|} \right)
\]  \hspace{1cm} (3)

The genetic, PSO and firefly optimization algorithms are utilized simultaneously to analyze the available computing nodes and determine which task should assign to each node. Thus the tasks can be optimally scheduled to suitable computing nodes.

**Estimation of optimal cut-points:** The ELM tree is based on the heuristics of uncertainty reduction and hence the decision tree nodes are split based on uncertainty measures such as information entropy and ambiguity. The over partition problem that occur in the induction stage can be reduced by embedding the ELMs as leaf nodes when the gain ratios of all the available splits are below a given threshold. Thus, the approach can be efficient but the ELM weight estimation is very important in the induction stage. The weights are optimized and the optimal weighted ELMs are selected. This aids in the reduction of over partition problem. But inorder to determine optimal weights, first the optimal cut-points of each attribute must be estimated.

In our approach, the larger datasets with distinct instances are trained and three efficient optimization techniques are used to find the optimal cut-points of each attribute. The information gain is used as the objective function for the determining the optimal cut-points. An attribute may contain many number of cut-points based on the data values. In general, the computations of information gain, gain ratio and split ratio are carried out for all the data associated with every cut-point. Then the cut-points with better results are selected for optimizing the weight. But this general approach consumes more time to compute the gain values as all the cut-points are computed. Hence, three optimization algorithms are used in this research with randomly selecting the cut-points which reduces the number of iterations and reduces the time for computation. The genetic algorithm, Particle Swarm Optimization (PSO) and firefly algorithms are used in this approach.

Given a training set X that contains distinct instances with n inputs and m outputs is taken with N hidden nodes. The cut-points c of every attribute is estimated from the population of cut-point solutions. The fitness value is the information gain calculated for each cut-point. The available cut-points in the population are analyzed as chromosomes and cross over and mutation is performed. Then, few of the cut-points with the data associated with
them selected in random and are forwarded to the host nodes for the information gain computation. This is repeated to determine the cut-points with the better information gain. The new results are saved in the population and the fitness of each is calculated. This process continues until the optimal cut-points of each attribute are determined or until the assigned numbers of iterations are completed.

The PSO algorithm assigns the cut-points as particles. The particle moves around the entire search space of the datasets with the uniform random vector. The motion of the particles is guided by their own best known position in the search-space. When positions are changed the new position will be used to guide the movements of the swarm. The process is repeated to discover a satisfactory solution of optimal cut-points. The local best and global best particles are selected based on the value of information gain. When the best values are selected, the other particles move towards the best with velocity \( v \) and update the position. When the selected particle is not the best, the previous best particle continues as the best particle. Thus the optimal cut-points are detected. The firefly algorithm is a novel method inspired from the nature to optimize the best solutions. The concept of firefly is that the firefly with the low brightness selects the firefly with the highest brightness. Each cut-point is assigned as a firefly and the firefly chooses the brighter one. If the newly selected cut-point has information gain that is not better than the previously selected weight then the firefly moves randomly until finding better solutions. This process continues till the brightest firefly, i.e., cut-point with better information gain is selected. The firefly algorithm reduces the total number of iterations and hence the optimal cut-points are selected with less computation time.

The optimal cut-points are selected only by the computation of the information gain which is performed at the host nodes. Hence the scheduling algorithm is called in between to determine the nodes to which the computation tasks are needed to be allocated:

**Algorithm 1: Task Scheduling**

- **Begin**
  - Align all the available nodes and tasks
  - Computing nodes = \{N1, N2, N3, N4, N5\}
  - Computation Tasks = \{T1, T2, T3, T4, T5, T6, T7, T8\}
- // Genetic Optimization
- Initialize Population of solutions \{T1, T2, T3, T4, T5, T6, T7, T8\}
- \{N1, N2, N3, N4, N5\}
- \text{Fitness} = \text{Computation time}
- Initialize chromosomes

\[
\begin{align*}
  C1 &= T1 \times T2 \times T3 \times T4 \times T5 \\
      &= N1 \times N2 \times N3 \times N4 \times N5 \\
  C2 &= T2 \times T3 \times T4 \times T5 \times T6 \\
      &= N1 \times N2 \times N3 \times N4 \times N5
\end{align*}
\]

- Estimate fitness of each chromosome
- Select two chromosomes as parents

- Parent 1 = T1 T2 T3 T4 T5
- Parent 2 = T4 T5 T6 T7 T8

- Crossover to produce child by swapping the elements in the parents

- Child1 = T7 T8 T3 T4 T5
- Child2 = N1 N2 N3 N4 N5

- Mutation is performed to reduce the duplication

- Child 1 = T3 T4 T5 T7 T8
- Child 2 = N1 N2 N3 N4 N5

- Estimate fitness of child 1 and child 2
- Place the child 1 and 2 in the population of solutions
- Repeat the process until best nodes determined
- // PSO
- Initialize particles

\[
\begin{align*}
  P1 &= T1 \times T2 \times T3 \times T4 \times T5 \\
      &= N1 \times N2 \times N3 \times N4 \times N5 \\
  P2 &= T2 \times T3 \times T4 \times T6 \times T7 \\
      &= N1 \times N2 \times N3 \times N4 \times N5 \\
  P3 &= T1 \times T2 \times T3 \times T5 \times T7 \times T8 \\
      &= N1 \times N2 \times N3 \times N4 \times N5 \\
  P4 &= T4 \times T5 \times T6 \times T7 \times T8 \\
      &= N1 \times N2 \times N3 \times N4 \times N5
\end{align*}
\]

- Initialize particle position \( P_i \)
- Particle velocity \( V_i = 0 \)
- Fitness = Computation time
- Estimate fitness of each particle
- Determine best particle as \( g_{best} \)
- Choose random variables \( R_p \) and \( R_t \)
- Determine particle velocity
- \( V_i = V_i + (g_{best} \cdot R_p) - (p_{best} \cdot R_t) \)
- Update particles towards \( g_{best} \)
- \( P_i = P_i + V_i \)
- Repeat process with updated particles
- Select \( g_{best} \) and \( p_{best} \)
- If updated \( P_i = P_i \)
- Best = Update \( P_i \)
else
    Ibest = P
end if
If Updated P gbest>P gbest
Gbest = Updated P
Else
Gbest = P
End if
Repeat until Gbest = Gbest
// Firefly Optimization
Initialize Population of feasible solutions
Iteration parameter k = 0
{T1, T2, T3, T4, T5, T6, T7, T8, N1, N2, N3, N4, N5}
Initialize fireflies X

\[ X_{1} = T1 T2 T3 T4 T5 \]
\[ X_{2} = T2 T3 T4 T6 T7 \]
\[ X_{3} = T4 T5 T6 T7 T8 \]
\[ X_{4} = N1 N2 N3 N4 N5 \]
Fitness f(x) = computation time
Estimate brightness I for each firefly
Calculate distance r between each two fireflies
\[ r = \| X_i - X_j \| \]
// where is the i-th firefly and j is the j-th firefly
Calculate Attractiveness for each firefly
// \( \beta_i \) is the attractiveness of the firefly at \( r = 0 \); \( \gamma \) is the media light absorption coefficient
If (I=1)
Move \( X_i \) towards \( X_j \)
Update brightness I
Rank fireflies to find gbest
k = k+1
Repeat until termination condition satisfied
After iteration, the same is repeated with other cut-points

Algorithm 2: Determination of cut-points
Begin
Align all the cut-points
// Genetic Optimization
Initialize Population of cut-points \( \{C1, C2, C3, C4...Cn\} \)
Fitness \( f(x) = \) Information gain
Initialize chromosomes

Chromosome 1=C1
Chromosome 2=C2
Chromosome n=Cn

Estimate \( f(x) \) of each chromosome
Call Algorithm 1
Select two chromosomes as parents
Parent 1=C1
Parent2=C2
Crossover to produce child H1 and H2

\[ H_{1} = \text{Parent}_1 + \beta(\text{Parent}_1 - \text{Parent}_2) \]
\[ H_{2} = \text{Parent}_2 + \beta(\text{Parent}_1 - \text{Parent}_2) \]
// \( \beta \) is a scalar value \( 0 < \beta < 1 \)
Mutation is performed to reduce the duplication
\[ H_{3} = \Delta H_{1} \]
\[ H_{4} = \Delta H_{2} \]
Estimate \( f(x) \) of H1 and H2
Call Algorithm 1
Place the H1 and H2 in the population of solutions
Chromosome with best \( f(x) \)
Gbest = Gbest
Repeat the process until Best cut-point selected
If new Gbest < Gbest
New Gbest = Gbest
Else
Gbest = Gbest
End if
End
// PSO
Initialize particles

\[ P_{2} = C_{2} \]
\[ P_{n} = C_{n} \]
Initialize particle position \( P_i \)
Particle velocity \( V_i = 0 \)
Fitness \( f(x) = \) Information gain
Estimate fitness of each particle
Call Algorithm 1
Particle with best \( f(x) \) as gbest
Choose random variables \( R_{P1} \) and \( R_{P2} \)
Determine particle velocity
\[ V_i = \omega V_i + (\text{Best position} \cdot R_{P1}) + (\text{gbest position} \cdot R_{P2}) \]
Update particles towards gbest
\[ P_i = P_i + V_i \]
Repeat process with updated particles
Select gbest and Ibest
If updated P>P
best = Updated P
else
best = P
End if
If Updated P gbest>P gbest
Gbest = Updated P
Else
Gbest = P
End if
Repeat until Gbest = Gbest
// Firefly Optimization
Initialize Population of feasible solutions
Iteration parameter k = 0
Initialize fireflies X

\[ X_{1} = C_{1} \]
\[ X_{2} = C_{2} \]
\[ X_{n} = C_{n} \]
Fitness \( f(x) = \) computation time
Estimate brightness I for each firefly
Call Algorithm 1
Calculate distance $r$ between each two fireflies

$$r = |X_i - X_j|$$

where $i$ is the $i$-th firefly and $j$ is the $j$-th firefly.

Calculate Attractiveness for each firefly

$$\beta(i) = \beta_0 e^{-\gamma r}$$

where $\beta_0$ is the attractiveness of the firefly at $r = 0$, $\gamma$ is the medium light absorption coefficient.

If $\beta(i) < \beta(j)$

Move $X_i$ towards $X_j$.

Update brightness $I$.

Rank fireflies to find safest.

$k = k+1$

Repeat until termination condition satisfied.

After iteration, the same is repeated with other cut-points.

Select the cut-points with better gain.

End

RESULTS AND DISCUSSION

The experiments are performed on three big datasets, Magic Telescope, Page Blocks and Wine Quality-White which are constructed from the UCI benchmark datasets. The datasets contain more number of instances than other datasets shown in Table 1. The computation time and the accuracy of classification using the proposed Optimization (FA) based ELM tree has to be estimated to evaluate the performance of the method. The proposed optimization based ELM tree is compared with the ELM tree ambiguity-based and ELM tree Information gain-based. Figure 1 shows the comparison of the ELM trees in terms of computation time while Fig. 2 shows the comparison in terms of accuracy for the Magic telescope dataset. The computation time for the Magic telescope dataset using the FA based ELM tree is very less compared to the ambiguity-based and the information gain based ELM tree approaches. Similarly the proposed approach also provides better accuracy than the ambiguity based and the information gain based ELM tree approaches.

The selected datasets are computed using ELM tree ambiguity-based and ELM tree Information gain-based and the proposed method with the three optimization techniques genetic algorithm, Particle Swarm Optimization (PSO) and firefly optimization algorithms. The methods are compared with each other in terms of accuracy parameters and the computation time. The accuracy of classifying the datasets is evaluated for each method and it is found that the proposed method achieves better results than other methods. It is further evaluated for determining which optimization based method is better. The cut-points $c$ of every attribute is estimated from the population of cut-point solutions using the optimization algorithms.

From the analysis in Table 2-4, it is clear that the firefly optimization based approach provides high accuracy of big data classification.

Table 1 describes the dataset specification which clearly shows that the datasets built are large in size which can be run with Pentium 4 Xeon, 3.06GHz CPU, 5 12MB RAM and Red Hat Linux9.0 operating system. Table 2-4 shows the comparison of the methods in terms of accuracy, mean absolute error, true positive, true negative, precision and f-measure for each of the selected datasets.

The methods are compared in terms of computation time with each possible combination of 2, 4, 6, and 8 computing nodes. The computation time is the time taken by the computing nodes to calculate the information gain, gain ratio, split ratio calculations of each cut point of the attributes.

The computation of gain parameters of each cut-point with the data associated with them takes more time in general. From Table 5, it is clear that for all datasets, the proposed method using Firefly-Firefly optimization has better computation time than other approaches.
Table 1: Dataset description

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Attribute</th>
<th>Class</th>
<th>Instance</th>
<th>Class distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magic telescope</td>
<td>10</td>
<td>2</td>
<td>19,020,000</td>
<td>12,323,000/6,888,000</td>
</tr>
<tr>
<td>Page blocks</td>
<td>10</td>
<td>5</td>
<td>5,473,000</td>
<td>4,913,000/3,529,000/1,113,000/88,000/28,000</td>
</tr>
<tr>
<td>Wine quality-white</td>
<td>11</td>
<td>6</td>
<td>4,898,000</td>
<td>2,198,000/1,457,000/38,000/4,175,000/163,000/20,000</td>
</tr>
</tbody>
</table>

Table 2: Classification accuracy of magic telescope

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>MAE</th>
<th>TP</th>
<th>TN</th>
<th>Precision</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM tree ambiguity-based</td>
<td>0.7985</td>
<td>0.1588</td>
<td>0.8767</td>
<td>0.7112</td>
<td>0.8288</td>
<td>0.8554</td>
</tr>
<tr>
<td>ELM tree information gain</td>
<td>0.8949</td>
<td>0.1289</td>
<td>0.9273</td>
<td>0.7460</td>
<td>0.8862</td>
<td>0.9063</td>
</tr>
<tr>
<td>GA based proposed</td>
<td>0.8876</td>
<td>0.1167</td>
<td>0.9323</td>
<td>0.7568</td>
<td>0.8997</td>
<td>0.9223</td>
</tr>
<tr>
<td>PSO based proposed</td>
<td>0.9012</td>
<td>0.1044</td>
<td>0.9487</td>
<td>0.8123</td>
<td>0.9187</td>
<td>0.9345</td>
</tr>
<tr>
<td>FA based proposed</td>
<td>0.9224</td>
<td>0.0987</td>
<td>0.9565</td>
<td>0.8598</td>
<td>0.9261</td>
<td>0.9513</td>
</tr>
</tbody>
</table>

Table 3: Classification accuracy of page blocks

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>MAE</th>
<th>TP</th>
<th>TN</th>
<th>Precision</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM tree ambiguity-based</td>
<td>0.7780</td>
<td>0.3488</td>
<td>0.8345</td>
<td>0.7072</td>
<td>0.8078</td>
<td>0.8054</td>
</tr>
<tr>
<td>ELM tree information gain</td>
<td>0.7987</td>
<td>0.2345</td>
<td>0.8432</td>
<td>0.7230</td>
<td>0.8123</td>
<td>0.8263</td>
</tr>
<tr>
<td>GA based proposed</td>
<td>0.8176</td>
<td>0.1876</td>
<td>0.8675</td>
<td>0.7348</td>
<td>0.8467</td>
<td>0.8423</td>
</tr>
<tr>
<td>PSO based proposed</td>
<td>0.8488</td>
<td>0.1367</td>
<td>0.8876</td>
<td>0.7893</td>
<td>0.8812</td>
<td>0.8645</td>
</tr>
<tr>
<td>FA based proposed</td>
<td>0.8793</td>
<td>0.1145</td>
<td>0.8967</td>
<td>0.8345</td>
<td>0.9045</td>
<td>0.9087</td>
</tr>
</tbody>
</table>

Table 4: Classification accuracy of wine quality-white

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>MAE</th>
<th>TP</th>
<th>TN</th>
<th>Precision</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM tree ambiguity-based</td>
<td>0.7456</td>
<td>0.2987</td>
<td>0.8298</td>
<td>0.7234</td>
<td>0.8342</td>
<td>0.8432</td>
</tr>
<tr>
<td>ELM tree Information gain</td>
<td>0.7787</td>
<td>0.2691</td>
<td>0.8475</td>
<td>0.7467</td>
<td>0.8534</td>
<td>0.8563</td>
</tr>
<tr>
<td>GA based proposed</td>
<td>0.7980</td>
<td>0.2076</td>
<td>0.8664</td>
<td>0.7691</td>
<td>0.8789</td>
<td>0.8645</td>
</tr>
<tr>
<td>PSO based proposed</td>
<td>0.8234</td>
<td>0.1675</td>
<td>0.8990</td>
<td>0.7901</td>
<td>0.8987</td>
<td>0.8821</td>
</tr>
<tr>
<td>FA based proposed</td>
<td>0.8567</td>
<td>0.1287</td>
<td>0.9124</td>
<td>0.8535</td>
<td>0.9234</td>
<td>0.9185</td>
</tr>
</tbody>
</table>

Table 5: Comparison in terms of computation time (sec)

<table>
<thead>
<tr>
<th>Variables</th>
<th>2 nodes</th>
<th>4 nodes</th>
<th>6 nodes</th>
<th>8 nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magic telescope</td>
<td>66,615</td>
<td>38,694</td>
<td>10,776</td>
<td>3,513</td>
</tr>
<tr>
<td>ELM tree Information gain</td>
<td>60,123</td>
<td>34,870</td>
<td>9,765</td>
<td>3,342</td>
</tr>
<tr>
<td>GA-GA</td>
<td>53,780</td>
<td>31,679</td>
<td>8,967</td>
<td>3,286</td>
</tr>
<tr>
<td>GA-PSO</td>
<td>53,567</td>
<td>29,786</td>
<td>8,754</td>
<td>3,156</td>
</tr>
<tr>
<td>GA-FA</td>
<td>48,087</td>
<td>27,443</td>
<td>8,653</td>
<td>2,897</td>
</tr>
<tr>
<td>PSO-GA</td>
<td>45,890</td>
<td>25,786</td>
<td>8,456</td>
<td>2,713</td>
</tr>
<tr>
<td>PSO-PSO</td>
<td>41,456</td>
<td>22,456</td>
<td>2,145</td>
<td>2,591</td>
</tr>
<tr>
<td>PSO-FA</td>
<td>39,086</td>
<td>19,756</td>
<td>7,654</td>
<td>2,340</td>
</tr>
<tr>
<td>FA-GA</td>
<td>37,654</td>
<td>17,650</td>
<td>7,423</td>
<td>2,216</td>
</tr>
<tr>
<td>FA-PSO</td>
<td>35,234</td>
<td>15,432</td>
<td>7,016</td>
<td>2,098</td>
</tr>
<tr>
<td>FA-FA</td>
<td>33,815</td>
<td>12,664</td>
<td>6,678</td>
<td>1,896</td>
</tr>
<tr>
<td>Page blocks</td>
<td>57,560</td>
<td>25,920</td>
<td>8,609</td>
<td>2,688</td>
</tr>
<tr>
<td>ELM tree Information gain</td>
<td>55,323</td>
<td>24,344</td>
<td>8,300</td>
<td>2,440</td>
</tr>
<tr>
<td>GA-GA</td>
<td>54,784</td>
<td>23,878</td>
<td>8,137</td>
<td>2,273</td>
</tr>
<tr>
<td>GA-PSO</td>
<td>53,567</td>
<td>22,133</td>
<td>8,000</td>
<td>2,107</td>
</tr>
<tr>
<td>GA-FA</td>
<td>48,977</td>
<td>21,233</td>
<td>7,800</td>
<td>1,987</td>
</tr>
<tr>
<td>PSO-GA</td>
<td>47,651</td>
<td>20,980</td>
<td>7,650</td>
<td>1,873</td>
</tr>
<tr>
<td>PSO-PSO</td>
<td>43,457</td>
<td>19,780</td>
<td>7,350</td>
<td>1,747</td>
</tr>
<tr>
<td>PSO-FA</td>
<td>41,324</td>
<td>18,670</td>
<td>7,100</td>
<td>1,557</td>
</tr>
<tr>
<td>FA-GA</td>
<td>39,003</td>
<td>17,347</td>
<td>6,767</td>
<td>1,353</td>
</tr>
<tr>
<td>FA-PSO</td>
<td>36,564</td>
<td>15,233</td>
<td>6,465</td>
<td>1,225</td>
</tr>
<tr>
<td>FA-FA</td>
<td>33,457</td>
<td>13,121</td>
<td>6,122</td>
<td>1,099</td>
</tr>
<tr>
<td>Wine quality-white</td>
<td>62,309</td>
<td>41,959</td>
<td>2,773</td>
<td>1,969</td>
</tr>
<tr>
<td>ELM tree Information gain</td>
<td>60,121</td>
<td>40,988</td>
<td>2,543</td>
<td>1,800</td>
</tr>
<tr>
<td>GA-GA</td>
<td>58,672</td>
<td>39,708</td>
<td>2,390</td>
<td>1,656</td>
</tr>
<tr>
<td>GA-PSO</td>
<td>56,454</td>
<td>37,650</td>
<td>2,122</td>
<td>1,532</td>
</tr>
<tr>
<td>GA-FA</td>
<td>52,344</td>
<td>35,667</td>
<td>1,944</td>
<td>1,425</td>
</tr>
<tr>
<td>PSO-GA</td>
<td>50,516</td>
<td>34,213</td>
<td>1,789</td>
<td>1,249</td>
</tr>
<tr>
<td>PSO-PSO</td>
<td>46,562</td>
<td>32,458</td>
<td>1,507</td>
<td>1,109</td>
</tr>
<tr>
<td>PSO-FA</td>
<td>42,390</td>
<td>30,248</td>
<td>1,467</td>
<td>1,023</td>
</tr>
<tr>
<td>FA-GA</td>
<td>40,707</td>
<td>28,089</td>
<td>1,260</td>
<td>980</td>
</tr>
<tr>
<td>FA-PSO</td>
<td>38,876</td>
<td>26,900</td>
<td>1,189</td>
<td>845</td>
</tr>
<tr>
<td>FA-FA</td>
<td>36,654</td>
<td>24,313</td>
<td>1,007</td>
<td>806</td>
</tr>
</tbody>
</table>
The FA-FA approach improves the computation speed of the nodes as the approach reduces the iterations to a great extent.

CONCLUSION

The big data classification is a wider area of research that provides vast informations to various fields. In this study, the over-partitioning problem in using the decision tree technique is overcome by the ELM tree approach. But the problem of high computation time and optimal cut-point selection is not considered at a centre stage. Hence, an approach is proposed with the efficient optimization algorithms, genetic optimization, particle swarm optimization and firefly optimization algorithms to determine the optimal cut-points. The approach, instead of selecting all cut-points, randomly selects the cut-points and computes the information gain. The computation of the information gain and gain ratio takes more time and hence a scheduling algorithm is used to schedule these computation tasks to efficient host nodes by estimating which node can process which tasks with higher efficiency based on the node data capacity using the optimization algorithms. Thus, the optimal cut-points are identified and the computation time is also reduced considerably especially using the Firefly optimization approach in the proposed method.

REFERENCES