Asian Journal of Information Technology 15 (24): 5051-5059, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

An Efficient Method for Big Data Classification

[1]S. Gayathri Devi and [2]M. Sabrigiriraj
[1]Department of Information Technology, Coimbatore Institute of Engineering and Technology, Coimbatore, 641109 Tamil Nadu, India
[2]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 inorder 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 optical 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-pint. 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

Corresponding Author: S. Gayathri Devi, Department of Information Technology, Coimbatore Institute of Engineering and Technology, Coimbatore, 641109 Tamil Nadu, India

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. Anbalagan 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.

Grolinger *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 Schaar (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 E^2LM based on the for the effective big data classification. The E^2LM 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 E^2LM 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 E^2LM. The problem with the approach is that the E^2LM 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. Inorder 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 Paakkonen and Pakkala (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. Miche *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 apriori 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 steam 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. Inorder 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 emergent 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 inorder 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 if 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_j and its i-th cut point is calculated. Here i= 1, 2, ... , N-1 and j = 1, 2, ..., n.

$$\text{Gain}(X, \text{cut}_{ij}) = \text{Info}(X) - I(\text{cut}_{ij}) \quad (1)$$

The information gain calculations $\{\text{Gain}_{1j}, \text{Gain}_{2j}, \text{Gain}_{3j}, \dots, \text{Gain}_{Mj}\}$ are assigned to the N computing nodes. The results are stored in a host node when the calculations are over. Then the remaining calculations $\{\text{Gain}_{M+1,j}, \text{Gain}_{M+2,j}, \dots\dots\dots \text{Gain}_{N-1,j}\}$ are assigned to free computing nodes in the next iteration. Then the calculation of gain ratio, Ratio_j is also done for each attribute to class them:

$$\text{Ratio}(X, A_j) = \frac{\text{Gain}(X, \text{cut}_{i(j)_j})}{\text{Split}(X, \text{cut}_{i(j)_j})} \quad (2)$$

Where:

$$\text{Split}((X, \text{cut}_{i(j)_j}) = -\left(\begin{array}{l} \frac{|(X_{i(j)_{j1}})|}{|X|}\log_2 \frac{|(X_{i(j)_{j1}})|}{|X|} + \\ \frac{|(X_{i(j)_{j2}})|}{|X|}\log_2 \frac{|(X_{i(j)_{j2}})|}{|X|} \end{array} \right) \quad (3)$$

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 computation 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}
Fitness = Computation time
Initialize chromosomes
        $C1 = \frac{T1\ T2\ T3\ T4\ T5}{N1N2N3N4N5}$
        $C2 = \frac{T2\ T3\ T4\ T5\ T6}{N1N2N3N4N5}$
        $C3 = \frac{T1\ T3\ T5\ T6\ T7}{N1N2N3N4N5}$
        $C4 = \frac{T4\ T5\ T6\ T7\ T8}{N1N2N3N4N5}$
Estimate fitness of each chromosome
Select two chromosomes as parents
        $Parent1 = \frac{T1\ T2\ T3\ T4\ T5}{N1N2N3N4N5}$
        $Parent2 = \frac{T4\ T5\ T6\ T7\ T8}{N1N2N3N4N5}$
Crossover to produce child by swapping the elements in the parents
        $Child1 = \frac{T7\ T8\ T3\ T4\ T5}{N1N2N3N4N5}$
        $Child2 = \frac{T4\ T5\ T6\ T1\ T2}{N1N2N3N4N5}$
Mutation is performed to reduce the duplication
        $Child1 = \frac{T3\ T4\ T5\ T7\ T8}{N1N2N3N4N5}$
        $Child2 = \frac{T1\ T2\ T4\ T5\ T6}{N1N2N3N4N5}$
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
        $P1 = \frac{T1\ T2\ T3\ T4\ T5}{N1N2N3N4N5}$
        $P2 = \frac{T2\ T3\ T4\ T6\ T7}{N1N2N3N4N5}$
        $P3 = \frac{T1\ T3\ T5\ T7\ T8}{N1N2N3N4N5}$
        $P4 = \frac{T4\ T5\ T6\ T7\ T8}{N1N2N3N4N5}$
Initialize particle position Pi
Particle velocity Vi = 0
Fitness = Computation time
Estimate fitness of each particle
Determine best particle as gbest
Choose random variables Rp and Rt
Determine particle velocity
$Vi = \omega i+(lbest\ position\times Rp)+(gbest\ position\times Rt)$
Update particles towards gbest
Pi = Pi+Vi
Repeat process with updated particles
Select gbest and lbest
If updated P>P
lbest = Updated P
```

```
else
lbest = 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
```

$$X1 = \frac{T1\ T2\ T3\ T4\ T5}{N1N2N3N4N5}$$

$$X2 = \frac{T2\ T3\ T4\ T6\ T7}{N1N2N3N4N5}$$

$$X3 = \frac{T4\ T5\ T6\ T7\ T8}{N1N2N3N4N5}$$

```
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  is the j-th firefly
Calculate Attractiveness for each firefly

// $\beta_0$ is the attractiveness of the firefly at r = 0; $\gamma$ is the media light absorption coefficient
If ($I_i < I_j$)
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
```

$$H_1 = Parent_1 + \beta(Parent_1 - Parent_2)$$

$$H_2 = Parent_2 + \beta(Parent_2 - Parent_1)$$

```
// $\beta$ is a scalar value ($0 < \beta < 1$)
Mutation is performed to reduce the duplication
```

$$H_1 = \Delta H_1$$

$$H_2 = \Delta H_2$$

```
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)
Cbest = Gbest
Repeat the process until Best cut-point selected
If new Cbest<Gbest
New Cbest = Gbest
Else
Gbest = Gbest
End if
End
// PSO
Initialize particles
```

P2 = C2

Pn = Cn

```
Initialize particle position Pi
Particle velocity Vi= 0
Fitness f(x) = Information gain
Estimate fitness of each particle
Call Algorithm 1
Particle with best f(x) as gbest
Choose random variables Rp and Rt
Determine particle velocity
Vi = $\omega$Vi+(lbest position×Rp)+(gbest position×Rt)
Update particles towards gbest
Pi = Pi+Vi
Repeat process with updated particles
Select gbest and lbest
If updated P>P
lbest = Updated P
else
lbest = 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
```

X1 = C1

X2 = C2

Xn = Cn

```
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 is the i-th firefly and is the j-th firefly
Calculate Attractiveness for each firefly
        $\beta(r) = \beta_0 e^{-\gamma r^2}$
// $\beta_0$ is the attractiveness of the firefly at r = 0; $\gamma$ is the media light absorption coefficient
If ($I_i$<$I_j$)
Move $X_i$ towards $X_i$
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
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.

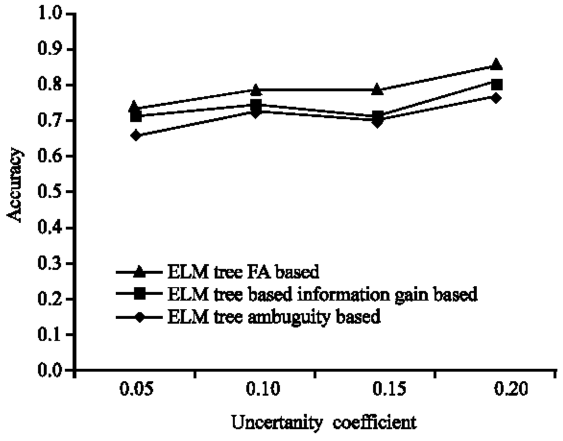

Fig. 1: Classification accuracy

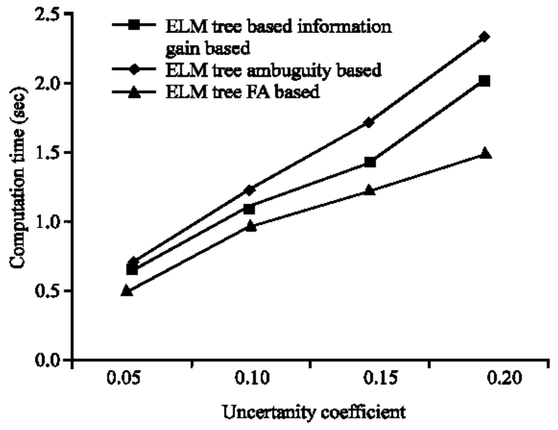

Fig. 2: Computation time

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

Dataset	Attribute	Class	Instance	Class distribution
Magic telescope	10	2	19 020 000	12 332 000/6 688 000
Page blocks	10	5	5 473 000	4 913 000/329 000/115 000/88 000/28 000
Wine quality-white	11	6	4 898 000	2 198 000/1 457 000/880 000/175 000/163 000/20 000

Table 2: Classification accuracy of magic telescope

Methods	Accuracy	MAE	TP	TN	Precision	F-measure
ELM tree ambiguity-based	0.7985	0.1588	0.8767	0.7112	0.8288	0.8554
ELM tree information gain	0.8694	0.1289	0.9273	0.7460	0.8862	0.9063
GA based proposed	0.8876	0.1167	0.9323	0.7568	0.8997	0.9223
PSO based proposed	0.9012	0.1044	0.9487	0.8123	0.9187	0.9345
FA based proposed	0.9324	0.0987	0.9565	0.8598	0.9261	0.9513

Table 3: Classification accuracy of page blocks

Methods	Accuracy	MAE	TP	TN	Precision	F-measure
ELM tree ambiguity-based	0.7780	0.3488	0.8345	0.7072	0.8078	0.8054
ELM tree information gain	0.7987	0.2345	0.8432	0.7230	0.8123	0.8263
GA based proposed	0.8176	0.1876	0.8675	0.7348	0.8467	0.8423
PSO based proposed	0.8488	0.1367	0.8876	0.7893	0.8812	0.8645
FA based Proposed	0.8793	0.1145	0.9067	0.8345	0.9045	0.9087

Table 4: Classification accuracy of wine quality-white

Methods	Accuracy	MAE	TP	TN	Precision	F-measure
ELM tree ambiguity-based	0.7456	0.2987	0.8298	0.7234	0.8342	0.8432
ELM tree Information gain	0.7787	0.2691	0.8475	0.7467	0.8534	0.8563
GA based proposed	0.7980	0.2076	0.8664	0.7691	0.8789	0.8645
PSO based proposed	0.8234	0.1675	0.8990	0.7901	0.8987	0.8821
FA based proposed	0.8567	0.1287	0.9124	0.8353	0.9234	0.9185

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

Variables	2 nodes	4 nodes	6 nodes	8 nodes
Magic telescope				
ELM tree ambiguity-based	66 615	38 604	10 776	3513
ELM tree Information gain	60123	34870	9765	3342
GA-GA	55780	31679	8967	3286
GA-PSO	53567	29786	8754	3156
GA-FA	48970	27643	8653	2897
PSO-GA	45890	25786	8456	2713
PSO-PSO	41456	22456	2145	2581
PSO-FA	39806	19756	7654	2340
FA-GA	37654	17650	7423	2216
FA-PSO	35234	15432	7016	2098
FA-FA	33245	12364	6578	1896
Page blocks				
ELM tree ambiguity-based	5756	2592	860	268
ELM tree Information gain	5523	2434	830	240
GA-GA	5478	2387	813	223
GA-PSO	5356	2213	800	210
GA-FA	4897	2123	780	198
PSO-GA	4765	2098	765	187
PSO-PSO	4345	1978	735	174
PSO-FA	4132	1867	710	155
FA-GA	3903	1734	676	135
FA-PSO	3654	1523	645	122
FA-FA	3345	1312	612	109
Wine quality-white				
ELM tree ambiguity-based	6239	4195	2738	1969
ELM tree Information gain	6012	4098	2534	1800
GA-GA	5867	3978	2390	1656
GA-PSO	5645	3765	2122	1532
GA-FA	5234	3567	1934	1423
PSO-GA	5016	3421	1789	1249
PSO-PSO	4656	3234	1567	1109
PSO-FA	4239	3024	1467	1023
FA-GA	4078	2809	1260	980
FA-PSO	3876	2690	1189	845
FA-FA	3654	2431	1007	800

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

Anbalagan, P. and R.M. Chandrasekaran, 2015. A parallel weighted decision tree classifier for complex spatial landslide analysis: Big data computation approach. Int. J. Comput. Appl., 124: 5-9.

Ding, L., Y. Liu, B. Song and J. Xin, 2015. Efficient ELM-based two stages query processing optimization for big data. Math. Prob. Eng., 2015: 1-12.

Divya, A.J. and S. Gagandeep, 2015. Classification of big data through artificial intelligence. Int. J. Comput. Sci. Mobile Comput., 4: 17-25.

Grolinger, K., M. Hayes, W.A. Higashino, L.A. Heureux and D.S. Allison *et al.*, 2014. Challenges for mapreduce in big data. Proceeding of the 2014 IEEE World Congress on Services, June 27-July 2, 2014, IEEE, Ottawa,Canada, ISBN:978-1-4799-5069-0, pp: 182-189.

Horta, E.G., C.L.D. Castro and A.P. Braga, 2015. Stream-based extreme learning machine approach for big data problems. Math. Prob. Eng., 2015: 1-17.

Hualong, Y.U., S. Changyin, Y. Wankou, Y. Xibei and X. Zuo, 2015. One uncertainty-based active learning algorithm using extreme learning machine. Neurocomputing, 166: 140-150.

Huang, G.B., D.H. Wang and Y. Lan, 2011. Extreme learning machines: A survey. Int. J. Mach. Learn. Cybern., 2: 107-122.

Huang, G.B., H. Zhou, X. Ding and R. Zhang, 2012. Extreme learning machine for regression and multiclass classification. Trans. Syst. Man Cybern. Part B (Cybernetics), 42: 513-529.

Lopez, V., D.S. Rio, J.M. Benitez and F. Herrera, 2014. On the use of map reduce to build linguistic fuzzy rule based classification systems for big data. Proceeding of the 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), July 6-11, 2014, IEEE, Granada, Spain, ISBN:978-1-4799-2072-3, pp: 1905-1912.

Miche, Y., A. Akusok, D. Veganzones, K.M. Bjork and E. Severin *et al.*, 2015. SOM-ELM-self-organized clustering using ELM. Neurocomputing, 165: 238-254.

Paakkonen, P. and D. Pakkala, 2015. Reference architecture and classification of technologies, products and services for big data systems. Big Data Res., 2: 166-186.

Rebentrost, P., M. Mohseni and S. Lloyd, 2014. Quantum support vector machine for big data classification. Phys. Rev. Lett., Vol. 113,

Rizk, Y. and M. Awad, 2015. On the distributed implementation of unsupervised extreme learning machines for big data. Procedia Comput. Sci., 53: 167-174.

Slavakis, K., G.B. Giannakis and G. Mateos, 2014. Modeling and optimization for big data analytics: (Statistical) learning tools for our era of data deluge. IEEE. Signal Process. Mag., 31: 18-31.

Sun, Y., Y. Yuan and G. Wang, 2011. An OS-ELM based distributed ensemble classification framework in P2P networks. Neurocomputing, 74: 2438-2443.

Tekin, C. and V.D.M. Schaar, 2013. Distributed online big data classification using context information. Proceeding of the 2013 51st Annual Allerton Conference on Communication, Control and Computing (Allerton), October 2-4, 2013, IEEE, California, USA., ISBN:978-1-4799-3410-2, pp: 1435-1442.

Triguero, I., D. Peralta, J. Bacardit, S. Garcia and F. Herrera, 2015a A map reduce solution for prototype reduction in big data classification. Neurocomputing, 150: 331-345.

Triguero, I., M. Galar, S. Vluymans, C. Cornelis and H. Bustince *et al.*, 2015bEvolutionary undersampling for imbalanced big data classification. Proceeding of the 2015 IEEE Congress on Evolutionary Computation (CEC), May 25-28, 2015, IEEE, Brussels, Belgium, ISBN: 978-1-4799-7492-4, pp: 715-722.

Xin, J., Z. Wang, L. Qu and G. Wang, 2015. Elastic extreme learning machine for big data classification. Neurocomputing, 149: 464-471.