An Enhancement of Big Data Classification with Minimum Consistent Subset and Virtual Machine Mapping

S. Gayathri Devi*, M. Sabrigiriraj**

**Department of Information Technology ,Coimbatore Institute of Engineering and Technology, Coimbatore, Tamil Nadu-641109, India (Tel: 7305255620; e-mail: gayathrideviphd789@gmail.com)*
*** Department of Electronics and Communication Engineering, SVS College of Engineering, Coimbatore, Tamil Nadu-642 109, India (e-mail: sabari_giriraj@yahoo.com)*

Abstract: Big data classification is primarily required for education, business, engineering, medical and science. The main issue of design and development of big data classification is parallelizing learning algorithms. Many algorithms were parallelized; one of them was the Decision Tree (DT) model. Information entropy and ambiguity were used for splitting the DT nodes. The over partitioning problem in DT induction was resolved by embedding Extreme Learning Machine (ELMs) as leaf nodes where the gain ratios of splits were lesser than a specified threshold. The ELM embedded DT is known as ELM-Tree. Then ELM-Tree was parallelized. This parallel ELM-tree model was further improved by finding optimal cut points for attributes using optimization algorithms which were called as Optimized parallel ELM-Tree (OPELM-Tree). In this paper, the finding of optimal cut points for all attributes is considered as unnecessary overhead. A Minimum Consistent Subset (MCS) is introduced to select optimal cut points for only optimal subsets to avoid unnecessary computation and improper resource utilization. MCS is formed based on hyper surface model, which is used to select optimal subsets for each class in the datasets. A hyper surface representation which is rectangular in shape is utilized to hold the samples and feature subsets. The proper boundary for MCS is found by optimization algorithms like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Firefly algorithm (FA). MCS is implemented using the Hadoop MapReduce (MR) framework, which is one of the current and powerful parallel programming models. A Dynamic Data partitioning (DDP) and Virtual Machine Mapping(AMM) is used in this paper for improving the performance of mapper and reducer respectively in MR. Additionally, Artificial Neural Network (ANN) is used to estimate the required number of mappers and reducers in MR for executing OPELM-Tree with better resource utilization. The performance of the proposed classifier is evaluated on three datasets and proved that the performance of MCS primarily based OPELM-Tree running on enhanced MR performed better than OPELM-Tree in terms of accuracy, precision and computation time.

Keywords: ELM-Tree model, Minimum Consistent Subset, Hyper Surface, Virtual Machine Mapping, Dynamic Data Partition, MapReduce, Genetic Algorithm, Particle Swarm Optimization, Firefly Algorithm.

1. INTRODUCTION

The characteristics of big data like huge volume, heterogeneity and fast growing of data from autonomous and multiple sources makes Big data mining very popular (Xindong Wu et al., 2014). The improved Frequent Pattern Mining (FPM) mining (Devi and Sabrigiriraj 2017) was proposed for big data classification by optimizing levy flight bat algorithm (LFBA). But the parallel big data processing was not processed in Map Reduce (MR).Because of data size is far beyond the processing capacity of a single personal computer, MR is used. MR is a parallel and high-performance computing tool which relies on cluster computers. The huge data is partitioned and produces intermediate key value pairs for each partitioned data in mapping phase. The partitioned data are then processed in parallel and the output of mapping phase is given as input to the number of reducers to get the final output. The author suggested that parallel programming using MR is being applied to data mining algorithms.

The parallel ELM-Tree (RanWang et al., 2015) is built by hybridization of DT and ELM. ELM is embedded as leaf node where splitting of DT has not met minimum criteria. The hybridization of ELM avoided the creation of over partitions while generating DT. However, the parallel ELM-tree is implemented using C programming language cannot be provided efficient output than parallel computing Tools like MR. The parallel ELM-Tree is implemented in MR framework and further enhanced by finding optimal cut points of attributes with work distribution scheduling (Devi and Sabrigiriraj 2016). The optimization algorithms GA, PSO and FA are utilized for both finding optimal cut points and scheduling nodes in MR framework. The Optimized Parallel ELM-Tree (OPELM-Tree) performed efficiently than parallel ELM-Tree in terms of accuracy and computation overhead.

The first proposal of this paper is improving the classification accuracy of OPELM-Tree by using MCS. In OPELM-Tree, considering all attributes for an optimum split is caused computation overhead that is solved by MCS. The computation overhead is reduced by removing irrelevant attribute and instances from the dataset by proposed MCS approach.MCS selects only reliable attributes and samples from the datasets for efficient classification. The minimal subset is selected by randomly formed rectangles on the dataset. A new MCS approach is proposed in this paper based on hyper surface classification (He et al., 2007). The boundary of the rectangles is adjusted by same optimization algorithms which are used in OPELM-Tree for finding an optimal split of attributes and work distribution schedule. The selected attributes and samples using MCS are further processed for OPELM-Tree classification.

The second proposal of this paper is improving the effectiveness of OPELM-Tree classification by enhancing MR framework. The enhancement of MR includes DDP, VMM (Slagter et al., 2013). The distance between data location and node location in Hadoop cause different data transferring between nodes.DDP is proposed to partition the data for each node based on the number of processing unit in VM. In VMM, the VM from the physical node which is nearest to mapper is allocated for reducers. The performance of Map reduce is improved certain level by DDP and VMM. The predetermined number of mappers and reducers irrespective of data and resource parameters causes many nodes either remains in an idle state or in the busy state of MR for many executions. To improve resource utilization of MR further, ANN based machine learning algorithm is proposed to estimate the required number of mappers and reducers to execute OPELM-Tree algorithm for data and resource parameters. The exact number of mapper and reducer selected for MR maximize resource utilization efficiently.

The organization of this paper is given as follows: In Section 2, previous research works on big data classification is analyzed and summarized. In Section 3 and 4, the methodology of proposed work is discussed in detail. In Section 5, performance evaluation for an outcome of proposed work is discussed briefly. In Section 6, a conclusion about results obtained for proposed approaches is explained.

2. LITERATURE SURVEY

In this section, various techniques proposed for big data classification are discussed. The brief discussion of existing methods provides clear ideas about the problems faced in big data classification.

A machine learning algorithm like Support Vector Machine (SVM) and DT algorithm were analyzed for supporting big data classification (Raikwal and Saxena, 2014). The analysis of machine learning algorithms provides a clear idea about the requirements for classifying a huge volume of data efficiently. A node selection in predictive models (Mahmood et al., 2015) was proposed to find the optimal number of hidden nodes in ELM with the help of SVM to improve the classification performance. In this approach, a median and mean of ELM was used as the threshold value to remove the inactive hidden nodes in ELM. A light-weight feature selection technique using accelerated PSO (Fong et al., 2016) was proposed for improving big data classification. This technique has found the best combination of selected features and classification algorithm to improve the data mining process of big data. A Parallel Sampling method based on Hyper Surface (PSHS) (He et al., 2015) was proposed to select the subset to improve big data classification. The uncertainties in big data were eliminated during sampling that uses MCS. The hyper surface was constructed by using fuzzy set.

The techniques were proposed (Xie et al., 2010) to distribute heterogeneous data based on the computing capacities of each node in the Hadoop Distributed File System (HDFS). The data skew problem of HDFS was overcome by using data reorganization and data redistribution algorithm. The data placement of a heterogeneous cluster was achieved by reducing data movement in a network. A virtual network mapping using PSO (Abedifar et al., 2013) was proposed to find an optimized map of the virtual network. The resources were allocated based on the position and velocity updates of each particle and updates their position based on particle best value and global best value.

A method was proposed to enhance the performance of MR execution in heterogeneous network. In this method, the heterogeneous data in a network was dynamically partitioned in the Map phase of MR framework. Then, the maximum resource utilization of big data was achieved by using VMM in Reduce phase. The GA based scheduler (Kune et al., 2014) was proposed for the big data cloud. The GA was processed iteratively to find the optimal resource allocation model to reduce the turnaround time of jobs. The GA based job scheduling algorithm (Lu et al., 2015) was proposed to enhance the performance of big data analysis. This approach utilizes the estimation module to define the optimized solution for job scheduling process. Based on the optimized solution, the process was scheduled to reduce the computation time and cost for data processing jobs.

3. AN OPTIMIZED MINIMUM CONSISTENT SUBSET SELCTION

The performance of the parallel ELM-Tree model was improved (Devi and Sabrigiriraj 2016) by selection of optimal cut-points using optimization algorithms based on the computation of information gain and gain ratio of partitioned data. Moreover, a scheduling algorithm was also proposed to allocate data from Master nodes to Slave Nodes. In this paper, the classification of big data is further improved by boundary optimized MCS.

3.1 Minimum Consistent Subset Selection

The performance of the parallel ELM-Tree model was improved (Devi and Sabrigiriraj 2016) by selection of optimal cut-points using optimization algorithms based on the computation of information gain and gain ratio of partitioned data. Moreover, a scheduling algorithm was also proposed to allocate data from Master nodes to Slave Nodes. In this paper, the classification of big data is further improved by boundary optimized MCS.

Algorithm 1: MCS

Input: Number of samples S, Number of attributes (dimensions) N, number of classes C, Number of rectangles R,Units U, minimum attributes a and minimum samples s.

Output: minimum consistent subset X_{MCS}

1. Draw rectangle R on dataset which covers S samples and N attributes

2. Divide R into units (U), SEU = B_Optimize(R, alg)

//SEU – Size of each unit, alg ∈ {GA, PSO, FA) to call Algorithm2

3. For all U

4. if (U_i contains more s for different C)

5. split U_i into sub units , SEU = B_Optimize(U, alg)

6. else

7. Lable U_i ∈ {1, C}

8. End If

9. End For

10. If (U_i && U_j have same C)

11. Combine U_i

12. End If

13. Collect equivalent subset samples from all combined Units

14. Selected subsets and samples are stored in X_{MCS}.

Consider a sample set *T* that contains a subsets and s samples. The hyper surface representation Y defines samples and subsets those are consistent at least availed in two rectangles in same classes. The equivalent class is defined as the attributes come under the same rectangles. Thus the MCS is build by

$$X_{MCS}|_{a,s} = X_{MCS} \cup Y \epsilon T \quad (1)$$

Where a=1...N, s=1…S and $X_{MCS} = \emptyset$ initially

Thus the minimum (consistent) subset is formed by hyper surface representation with the help of the given equation (1). In this the upper and lower bound region of the rectangle is optimized by the optimization algorithms like GA, PSO and FA.

3.2 A Boundary Optimization of MCS

The boundary optimization of rectangles is the selection of optimal regions for MCS. The GA, PSO and FA algorithms are used separately to select the optimal regions of upper and lower bounds. The optimal bounds are selected to obtain equivalent class samples in each rectangle units. The optimal selection of most representative subsets using MCS decrease training time, memory usage and increase the accuracy of OPELM-Tree. The rectangles boundaries are adjusted until all the rectangle regions contain same classes. Finally, the similar regions are combined to form larger equivalent class regions. The samples from equivalent class regions are enough to train ELM-Tree classifier more accurately. The optimal boundary value of MCS rectangles is found by optimization algorithms with the objective of maximizing information gain.

Algorithm 2: B_Optimize(R,alg) // MCS Boundary Optimizer

Input: R with N attributes and S samples or U with a attributes and s samples and C

Output: optimal boundary region oa and os //optimized s and optimized a

Initialize Fitness $f(U)$ as information gain (IG) of Units

$$f(U) = IG(\mathrm{C},\mathrm{U}) = \mathrm{E}(\mathrm{C},\mathrm{R}) - \mathrm{E}(\mathrm{C},\mathrm{U})$$

$\mathrm{E}(\mathrm{C},\mathrm{R})$- Entropy of instances in R

$\mathrm{E}(\mathrm{C},\mathrm{U})$-Entropy instances in U.

$$\mathrm{E} = \sum_{\mathrm{i}=1}^{\mathrm{C}} \mathrm{p_C} \log_2 \mathrm{p_C}$$

where p_C denotes the probability of instances belonging to class C

//Genetic optimization

1. If (alg = = GA)

2. Initialize population of boundaries {P_1, P_2, P_3,….,P_p} as P chromosomes, Each P_i select boundary of rectangles s ± $rand$ ∈ S for samples and a ± $rand$ ∈ N for attributes ,mutation rate

3. Calculate Fitness $f(U)$ for each P_i

4. Sort P in descending order of fitness

5. Position based Cross Over operation

Select two chromosomes as parents

$$Parent_m = P_m$$
$$Parent_n = P_n$$

Where n=1.. p; m =1 to p and n ≠ m

Crossover to produce child $Ch_n$and Ch_m

$$Ch_n = Parent_m + \mu(Parent_m - Parent_n)$$
$$Ch_m = Parent_n + \mu(Parent_n - Parent_m)$$

//μ is a scalar value ($0 < \mu < 1$)

6. Mutation operation

Change the s and a value of any child as per mutation rate

7. Calculate Fitness $f(U)$ for each Ch_i

8. Consider each Ch_i as P_i then go to 4

9. Repeat until all chromosomes have same $f(U)$

10. Return s and a value of chromosome as os and oa

11. End If

12. If (alg = = PSO)

13. Initialize the P particles (Pl_1, Pl_2, Pl_3,…., Plp} as particles , each Pl_i select boundary of rectangles $s \pm rand \in S$ for samples and $a \pm rand \in N$ for attributes randomly and velocity of particle V

14. Iteration = 1

15. Calculate Fitness $f(U)$ for each Pl_i

16. Select particle with best fitness as g_{best}

17. If (Iteration = = 1)

18. Assign initial fitness as p_{best} for all particles

19. else

20. If ($p_{best} < f(Pl_i)$

21. $p_{best} = f(Pl_i)$

22. Iteration = Iteration +1

23. End if

24. End if

25. Choose random variables r_1 and r_2

26. Velocity update, let ω is inertia of particles

$$V(i) = \omega V(i-1) + (p_{best}(i-1) \times r_1) + (g_{best}(i-1) \times r_2)$$

27. $Pl_i\ (i) = Pl_i\ (i-1) + V(i)$

28. Repeat steps 15 to 27 until all particles have fitness

29. Return s and a value of particles as os and oa

30. End If

31. If (alg = = FA)

32. Initialize Iteration parameter $q = 0$

33. Initialize P number of fireflies (F_1, F_2…, F_p). Each F_n select boundary of rectangles $s \pm rand \in S$ for samples and $a \pm rand \in N$ for attributes randomly.

34. Calculate Fitness $f(U)$ (brightness b) of each of firefly

35. Find distance d between two fireflies

$$d = \|F_x - F_y\|$$

//where F_x is the position of x^{th} firefly and F_y is position of y^{th} firefly. position have selected s and a values

36. Calculate Attractiveness for each firefly

$$\beta(d) = \beta_0 e^{-\gamma d^2}$$

//β_0 is the attractiveness of the firefly at $d = 0$; γ is the media light absorption coefficient

37. If ($b_x < b_y$)

Move F_x towards F_y

38. Update brightness b using $f(U)$

39. $q = q + 1$

40. Repeat steps 34 to 40 until all firefly have same Fitness

41. Return s and a value of particles as os and oa

42. End If

From Algorithm 2, the optimal boundaries for MCS are found and returned to Algorithm1. In GA, optimal boundaries are found by crossover and mutation function. Then their boundary values are updated based on the fitness function. In PSO algorithm, the number of particles is initialized with initial boundary values. The optimal boundary is estimated based on the position update and velocity update function. In firefly algorithm, optimal boundary values are updated based on the attractiveness of each firefly to others. Thus the optimal boundaries are found and it is returned to Algorithm1.The features and samples obtained from MCSs are given for OPELM-Tree. The reduced attributes and samples increase the overall performance of OPELM-Tree Classifier.

4. ENHANCED MAPREDUCE

DDP and VMM methods are used to improve the performance of MR in heterogeneous environments. DDP is applied in mapper and VMM in reducer phase to maximize resource utilization. The exact number of mappers and reducers required to classify partitioned data greatly improve the performance of MR (EMR) further. ANN is used to estimate the exact number of mappers and reducers based on the trained ANN model. The ANN is trained with the node and data characteristics of previous MR executions.

Initially in DDP, the processing speed of VM is found by allocating same amount of data to all nodes. The response time of node is the speed of each VM. This is based on the number of virtual processing units (VPU) of that VM, and the PM it is running on. Then data is repartitioned on each PM. Let considered in PM1, four virtual machines has processing rate of 2, 5, 8 and 10.The initial split size of each VM is 50 GB. So the total size of four VM is 200GB and total speed of four VM is 25 units. The fragment size is found by (2)

$$F_S = \frac{T_{ds}}{T_s} \quad (2)$$

Where F_S is fragment size, T_{ds} is total data size T_s is Total speed of all VM .

The split size for each VM is found by multiplying the VPU speed of each VM by theF_S. For PM1 the F_S is 200/25 = 8.

$$S_S = VM_{PS} \times F_S \quad (3)$$

Where S_S is split size, VM_{PS} is VM processing speed.

Therefore, for VM1 in PM1 has a VPU value of 5, it's S_S to be 5 × 8 = 40GB using equation (3).The S_S for all nodes is found by the same method.

In VMM, the reducer is allocated to right PM based on the split size and the availability of a VM. The reducers are assigned to PM which stored the larger portion of data for reducers. Always the reducer is assigned to fastest VM in PM. The first reducer is assigned to fastest VM and the second reducer is assigned to next fastest VM in PM where more than one reducer is allocated. If VM is not available in PM to allocate more reducers, the reducer is assigned to VM

in another PM based on the best fit method by VMM. VMM assigned reducers to PM by reviewing the data size contributed by each PM to reducers.

Dynamically determining the number of mappers and reducers in MR improves the resource utilization more efficient. The machine learning algorithm ANN is used to estimate the expected number of mappers and reducers. The files Size, Attribute length, Number of Attributes, Number of Samples are the parameters of Dataset. Network Bandwidth, Number of VM in PM and Speed of PM are the parameters of PM. Number of VPU,Mips of VPU,RAM and Storage Space are the parameters of VM. The parameters of Dataset, PM and VM of previous execution in MR with corresponding no of mapper, reducer used are given as input to the input layer of ANN. The execution time is converted to nominal value as less, normal and high. The execution time is given as input to the output layer of ANN. The trained ANN model is generated from training process of ANN for the given inputs to both layers. The trained ANN model is used to predict the number of mapper and reducer required for the current parameters of Dataset, PM and VM. The current parameters with possible no of mappers and reducers are given as input to the input layer in prediction stage of ANN. The number of mappers and reducers are decided from the instance which is predicted as less execution time by ANN.

Algorithm 3 Estimation of number of mapper and reducer using ANN

Input: Number of parameters X and Execution Time T

Output: Expected number of mappers and reducers

//Training Phase

1.Initialize number of layer 3, number of node: input layer 6, hidden layer 3 and output layer 1, minimum error arte 0.01.

2. Calculate weighted sum of input layer values in hidden layer

$$V_i = \sum_{i=1}^{n} W_i X_i$$

//W_i is ith input weight, X_i input value

3.Calculate the sigmoid activation function of hidden nodes

$$f(V_i) = \frac{1}{1 + \exp^{(-\sum_i W_i X_i - b_i)}}$$

// b is the bias value

4. The output function

$$O_i = \sum_{i=1}^{n} W_{hi} f(V_i) + b_i$$

//W_h is the hidden layer weight

5.Evaluate the error function

$$\varepsilon_o = \frac{W_i f(X_i) + b_i}{\varepsilon_H}$$

//ε_o is error rate on output nodes and ε_H is error rate at hidden layer nodes

6. Repeat step2 by adjusting W_i by adding $\varepsilon_o$and adjust W_h by adding ε_h until obtain expected error rate

7.Save generated W_i , W_{hi} and b as trained model

//Testing Phase

8. Initialize number of testing samples Y_i and target class T_i

9. Calculate activation function same as training data

10. Predict test instances label

$$T_i = \sum_{i=1}^{n} W_i f(Y_i) + W_{hi} f\big(f(Y_i)\big) + b_i$$

// W and b are obtained from ANN training

11. Select test Sample predicted as less execution time

12.Obtain the number of mappers and reducers from this sample.

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

The experiments are conducted for different datasets for existing and proposed big data classification. In order to prove the effectiveness of the proposed big data classification, the performance is measured in terms of True positive, True negative, Accuracy, Precision, F-Measure, Mean Absolute Error (MAE) and computation time. Three big datasets namely Page Blocks, Wine Quality-White and Magic Telescope are used which includes the number of instances constructed from UCI benchmark datasets. The description of these datasets is shown in Table 1.

Table 1. Datasets.

Dataset	Attribute	Class	Instance	Class Distribution
Page Blocks	10	5	5473 000	4 913 000/329 000/115 000/88 000/28 000
Wine Quality-White	11	6	4898 000	2198000/1 457 000/880 000/175 000/163 000/20 000
Magic - Telescope	10	2	19020 000	12332000/6 688 000

The experiment is conducted in MR Hadoop environment. ELM-tree ambiguity-based and ELM-tree Information gain based are parallel ELM-Tree classifier. The ELM-tree Information gain is superior in performance than other. The ELM tree with information gain is optimized (Devi & Sabrigiriraj, 2016) by GA, PSO and FA optimization algorithms. It is also proved that FA-FA-OPELM-Tree is obtained best results in all measures with less computation time.

Table 2. Acronym List.

Methods	List
PELM-Tree ambiguity-based	1
PELM-Tree IG based	2
FA-FA-OPELM-Tree for GA-MCS	3
FA-FA-OPELM-Tree for PSO-MCS	4
FA-FA-OPELM-Tree for FA-MCS	5

In Table 2 FA-FA-OPELM-Tree represent FA based optimal cut point and FA based optimal scheduling.MCS is optimized by GA, PSO and FA which is represented after the representation of Tree model . The PELM stands for parallel ELM-Tree and OPELM stands for Optimized parallel ELM-Tree. This was explained in abstract and introduction part of this article.

In this experiment, The MCS obtained from GA, PSO and FA based algorithms are given as input to the FA-FA-OPELM-Tree classifier. The result of FA-FA-OPELM-Tree for MCS running in MR frame work shown in Table 3, Table 5 and Table 7. From the analysis, FA based MCS improves the Accuracy FA-FA-OPELM-Tree classifier than GA based MCS and PSO based MCS from 6% to 10 % for all three datasets.

Table 4, Table 6, and Table 8 shows the results of same FA-FA-OPELM-Tree for optimized MCS running on EMR framework. The result shows that FA-FA-OPELM-Tree for FA-MCS in EMR is given 20% better accuracy than FA-FA-OPELM-Tree for FA-MCS in MR.FA-FA-OPELM-Tree for FA-MCS in EMR is 3 % better than FA-FA-OPELM-Tree for PSO-MCS in EMR. The result from all tables for all dataset it can be concluded FA-FA-OPELM-Tree for FA-MCS in EMR provides high Accuracy, Precision, F-measure, True positive and less True negative, Mean absolute error than other MCS.

Table 3. Classification Measures of ELM-Tree classifiers for MCS in MR-Page Blocks.

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.779	0.347	0.835	0.708	0.808	0.806
2	0.799	0.233	0.844	0.724	0.813	0.827
3	0.818	0.186	0.868	0.735	0.847	0.844
4	0.849	0.135	0.888	0.790	0.882	0.865
5	0.880	0.113	0.907	0.835	0.905	0.909

Table 4. Classification Measures of ELM-Tree classifiers for MCS in EMR-Page Blocks

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.780	0.346	0.836	0.709	0.809	0.807
2	0.800	0.232	0.845	0.725	0.814	0.828
3	0.819	0.185	0.869	0.736	0.848	0.844
4	0.850	0.134	0.889	0.791	0.883	0.866
5	0.881	0.112	0.908	0.836	0.906	0.910

Table 5. Classification Measures of ELM-Tree Classifiers for MCS in MR-Wine Quality-White

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.746	0.297	0.830	0.724	0.835	0.844
2	0.779	0.268	0.848	0.747	0.854	0.857
3	0.799	0.206	0.867	0.770	0.879	0.865
4	0.824	0.166	0.900	0.791	0.899	0.883
5	0.857	0.127	0.912	0.836	0.924	0.919

Table 6. Classification Measures of ELM-Tree classifiers for MCS in EMR-Wine Quality-White

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.747	0.296	0.831	0.725	0.836	0.845
2	0.780	0.267	0.849	0.748	0.855	0.858
3	0.800	0.205	0.868	0.771	0.880	0.866
4	0.825	0.165	0.901	0.792	0.900	0.884
5	0.858	0.126	0.913	0.837	0.925	0.920

Table 7. Classification Measures of ELM-Tree classifiers for MCS in MR-Magic Telescope

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.799	0.157	0.877	0.712	0.829	0.856
2	0.869	0.127	0.928	0.747	0.887	0.907
3	0.888	0.115	0.933	0.757	0.900	0.923
4	0.902	0.104	0.949	0.813	0.919	0.935
5	0.933	0.097	0.957	0.860	0.927	0.951

Table 8. Classification Measures of ELM-Tree classifiers for MCS in EMR-Magic Telescope

Alg	Accuracy	MAE	TP	TN	Precision	F-measure
1	0.800	0.156	0.878	0.713	0.830	0.857
2	0.870	0.126	0.929	0.748	0.888	0.908
3	0.889	0.114	0.934	0.758	0.901	0.924
4	0.903	0.103	0.950	0.814	0.920	0.936
5	0.934	0.096	0.958	0.861	0.928	0.952

Table 9 and Table 10 shows the computation time of all ELM-Tree algorithms for MCS running in MR and EMR for number of 2, 4, 6 and 8 computing nodes in Hadoop system. FA-FA-OPELM-Tree-MCS represent FA based optimal cut points and FA based optimal scheduling with simple MCS feature selection. FA-FA-OPELM-Tree-GA-MCS represents PELM-Tree with GA optimized MCS feature selection, FA based optimal cut points and FA based optimal scheduling. FA-FA-OPELM-Tree-PSO-MCS represents ELM Tree with PSO optimized MCS feature selection, FA based optimal cut points and FA based optimal scheduling. FA-FA-OPELM-Tree-FA-MCS represents PELM Tree with FA optimized MCS feature selection, FA based optimal cut points and FA based optimal scheduling. The experimental results from

Table 9 and Table 10 proves that the FA-FA-OPELM-Tree-FA-MCS is taking less computational time for all three datasets and for any number of nodes used in Hadoop system.

Table 9. Comparison in terms of computation time (secs) of ELM-Tree algorithms for MCS running in MR

Alg	2 nodes	4 nodes	6 nodes	8 nodes
Page blocks				
1	5746	2582	850	258
2	5513	2424	840	230
3	3298	1261	590	102
4	3134	1154	583	97
5	3063	1062	572	93
Wine Quality-White				
1	6229	4185	2728	1959
2	6002	4088	2524	1790
3	3473	2289	975	768
4	3318	2195	968	762
5	3299	2052	960	757
Magic Telescope				
1	66 605	38 594	10 766	3503
2	60113	34860	9755	3332
3	33156	12276	6479	1765
4	32998	12180	6382	1685
5	32887	12999	6269	1591

Table 10. Comparison in terms of computation time (sec) of ELM-Tree algorithms for MCS running in EMR

Alg	2 nodes	4 nodes	6 nodes	8 nodes
Page blocks				
1	5736	2572	840	248
2	5503	2414	835	220
3	3288	1251	580	99
4	3124	1144	573	95
5	3053	1052	562	90
Wine Quality-White				
1	6219	4175	2718	1949
2	5992	4078	2514	1780
3	3463	2279	970	764
4	3308	2185	963	759
5	3289	2042	955	754
Magic Telescope				
1	66 595	38 574	10 756	3493
2	60103	34850	9745	3322
3	33146	12266	6469	1755
4	32988	12170	6372	1675
5	32877	12989	6259	1581

5. CONCLUSIONS

The big data classification with MCS is proposed to overcome the problem of unnecessary computation, memory and resource utilization of OPELM-Tree Classifier. MCS is used to determine the minimum number of samples required for classification. It is achieved through hyper surface representation that modeled the samples into the root region of rectangular shape. The upper and lower bound on the hyper surface is optimized by optimization algorithms like GA, PSO and FA. The EMR is also proposed to enhance MR with DDP and VMM. The number of mappers and reducers required for big data classification is dynamically determined. Finally proved that optimized MCS and EMR improve the accuracy and time of Big data classification.

REFERENCES

Abedifar, V., Eshghi, M., Mirjalili, S., and Mirjalili, S. M. (2013). An optimized virtual network mapping using PSO in cloud computing. In *2013 21st Iranian Conference on Electrical Engineering (ICEE)* (pp. 1-6). IEEE.

Devi, S.G. and Sabrigiriraj, M(2017). Swarm intelligent based online feature selection (OFS) and weighted entropy frequent pattern mining (WEFPM) algorithm for big data analysis. *Cluster Computing.Springer Science+Business Media,LLC, part of Springer Nature (pp.1-13)*

Devi, S.G., and Sabrigiriraj, M. (2016) . An efficient method for big data classification. *Asian Journal of Information Technology, 15 (24).(pp. 5051-5059).Medwell journals*

Fong, S., Wong, R., and Vasilakos, A. V. (2016). Accelerated PSO swarm search feature selection for data stream mining big data. *IEEE Transactions on Services Computing*, *9*(1), 33-45.

He, Q., Zhao, X. R., and Shi, Z. Z. (2007). Sampling Based on Minimal Consistent Subset for Hyper Surface Classification. In *Machine Learning and Cybernetics, 2007 International Conference on* (Vol. 1, pp. 12-18). IEEE.

He, Q., Wang, H., Zhuang, F., Shang, T., and Shi, Z. (2015). Parallel sampling from big data with uncertainty distribution. *Fuzzy Sets and Systems*, *258*, 117-133.

Kune, R., Konugurthi, P. K., Agarwal, A., Chillarige, R. R., and Buyya, R. (2014). Genetic Algorithm based Data-aware Group Scheduling for Big Data Clouds. In *Big Data Computing (BDC), 2014 IEEE/ACM International Symposium on* (pp. 96-104). IEEE.

Lu, Q., Li, S., and Zhang, W. (2015). Genetic Algorithm Based Job Scheduling for Big Data Analytics. In *2015 International Conference on Identification, Information, and Knowledge in the Internet of Things (IIKI)* (pp. 33-38). IEEE.

Mahmood, S. F., Marhaban, M. H., Rokhani, F. Z., Samsudin, K., and Arigbabu, O. A. SVM–ELM: Pruning of Extreme Learning Machine with Support Vector Machines for Regression. *Journal of Intelligent Systems*.

Raikwal, J. S., and Saxena, K. (2014). Weight based Classification Algorithm for Medical Data.*International Journal of Computer Applications, 107*(21).

RanWang,Yu-LinHe,,Chi-YinChow,Fang-FangOu and JianZhang (2015). Learning ELM-Tree from big data based on uncertainty reduction. *Fuzzy Sets and Systems 258 (2015) 79–100. Elsevier B.V.*

Slagter, K., Hsu, C. H., and Chung, Y. C. (2013). Dynamic Data Partitioning and Virtual Machine Mapping: Efficient Data Intensive Computation. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on* (Vol. 2, pp. 220-223). IEEE.

Xie, J., Yin, S., Ruan, X., Ding, Z., Tian, Y., Majors, J.,and Qin, X. (2010). Improving mapreduce performance through data placement in heterogeneous hadoop clusters. In *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on* (pp. 1-9). IEEE.

Xindong Wu, Xingquan Zhu and Gong-Qing Wu (2014). Data mining with big data.*IEEE transactions on knowledge and data engineering*, *26*(1), 97-107.