IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-8, Aug 2018 E-ISSN: 2347-2693

A Mapreduce Approach To Deal with Big Data Pre Processing And Classification Problems Based On Evolutionary Algorithms

M.S. Saranya[1*], N. Jayaveeran[2]

[1]Computer Science, Khadir Mohideen College, Bharathidasan University, Thiruchirapalli, India
[2]Computer Science, Khadir Mohideen College, Bharathidasan University, Thiruchirapalli, India

[*]*Corresponding Author: sara88.mca@gmail.com, Mobile No: 9787873810*

Available online at: www.ijcseonline.org

Accepted: 14/Aug/2018, Published: 31/Aug/2018

***Abstract*—** The big data is a term which is used to describe the exponential growth in data that has occurred recently and it also represents an immense challenge for traditional learning techniques. In order to deal with big data pre processing and classification problems, a novel MapReduce-Neuro Ant Colony (MR-NAC) algorithm was proposed. The proposed algorithm used MapReduce framework to pre process and classify the large dataset which is found to difficult without using the MapReduce framework. The experimentation for the proposed work is carried on two different datasets and results obtained are discussed. The obtained results are much satisfactory which supports the proposed novel algorithm for big data pre processing and classification. AUC and execution time are the two metrics which were used to measure the performance of the proposed MR-NAC Algorithm

***Keywords*—**Big Data, Map Reduce, Neural Network, Ant Colony, Pre process, Classification, execution time.

I. INTRODUCTION

The major concern in most of the data mining and machine learning algorithms is that they are sometimes incompetent of learning from a very hefty database [1] which is also denoted by the term "Big Data". The term "Big Data" has attracted most of research areas like bio informatics, medicine, marketing or financial business, national intelligence applications, medical technology, cyber security data [2, 3]. Some standard data mining techniques are adopting the concept of Cloud Computing to discover knowledge from massive amounts of data [4, 5, and 6]. In order to adapt the data mining tools for the big data problems, the algorithms have to be redesigned and included in parallel environment. In order to address such problem, the MapReduce [7, 8] and its distributed file system [9] was introduced by Google. It was introduced as an effective and robust framework to address and analyze the big data set. Because of its fault tolerance and simplicity, the MapReduce is taken into consideration for implementing the data mining techniques [10] when compared with Message Passing Interface [11, 12] which is also a parallel implementation technique. Classification of big data has become an indispensable chore in wide selection of fields.

The main objective of this paper is to apply evolutionary approach for the classification of big data based on the MapReduce paradigm. This algorithm is called as MapReduce-Neuro Ant Colony (MR-NCA) Algorithm. More specifically the purpose of this paper is to

(a) To design an Evolutionary technique for classification over the MapReducer
(b) To analyze and illustrate the scalability of the proposed scheme based on classification accuracy and execution time.

To analyze the proposed approach, two big classification data sets with 32million instances and 2000 features is used for experimentation. The paper is organized as follows: Section I gives introduction about this work. Section II gives some background knowledge about the data classification and MapReduce. The MR-NAC Algorithm is discussed in section III and experimental setup for implementing the algorithm and the results obtained are discussed in section IV. Section V concludes the work with future enhancement.

II. CLASSIFICATION

The classification problem arises when then is a notable difference between the number of samples belonging to different classes [13,14]. This problem is found to be the most imperative one to be dealt with because it is seen in most of the real world applications. It is a major challenge because the stand search process that is embedded in most of the techniques is guided by a global search measure that does not consider the difference in the number of instances. Because of this the instances of the minority classes are not

been considered. In order to avoid such situations a different classification algorithm is needed which can overcome the disadvantages found in existing techniques.

A. MapReduce

MapReduce [9,10] is one of the most popular programming models to deal with big data. It was proposed by Google in 2004 and designed for processing huge amounts of data using a cluster of machines. The MapReduce paradigm is composed of two phases: map and reduce [17]. In general terms, in the map phase, the input dataset is processed producing some intermediate results. Then, the reduce phase combines them in some way to form the final output [18]. The flowchart of the MapReduce framework is given in Figure 1.

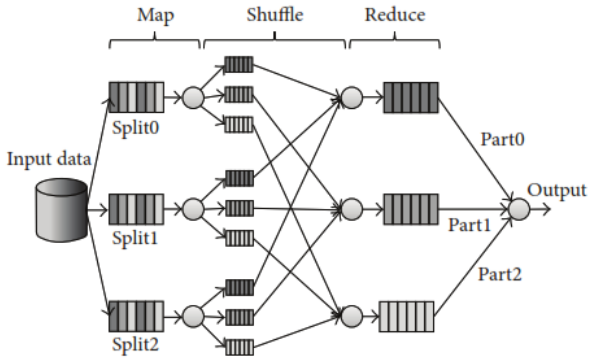

Figure 1: Framework of the MapReduce

B. Neuro Ant Colony (NAC) Algorithm

At the initial level, the pheromone level is initialized to zero. The information Gain of each attributes is calculated and the ants are generated. The subset is constructed and checked whether all the constructions is finished. If all the constructions have been finished, the ants have found the best solution for the problem. From the solution, select the local best and global best subset. Store the optimal dataset obtained. Update the τ and η value. The pheromone trail evaporated then.

Pheromone update $\tau_{ij} = \tau_{ij} + \Delta_{\tau}k$

Pheromone Evaporation $\tau_{ij} \leftarrow (1-p)\tau_{ij}, \forall (i,j) \in A$

The optimal dataset obtained is then classified using Multi Layer Perceptron Network [19]. The best features are assigned as input and transmitted to the hidden layer. In the hidden unit, the output is calculated. Each output unit receives a target pattern according to the input training pattern and error correction term is calculated. On the basis of calculated error, the weight and bias is also updated in the network. The bias and weight are updated. The working is given in Figure 2.

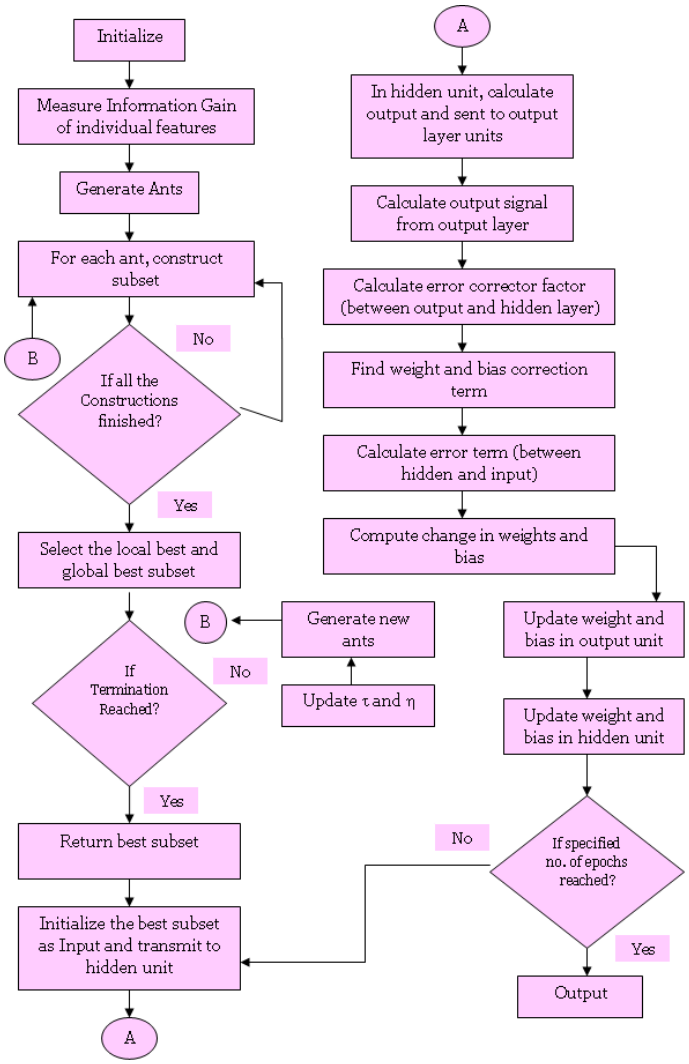

Figure 2: working of NAC Algorithm

III. MAPREDUCE – NEURO ANT COLONY (MR-NAC) ALGORITHM

This section explains the parallelization of the NAC Algorithm by using a MapReduce procedure. Let T be a training set, stored in HDFS and randomized. Let *m* be the number of map tasks. The splitting procedure of MapReduce divides *T* in *m* disjoint subsets of instances. Then, each T_i subset ($i \in \{1, 2 \ldots m\}$) is processed by the analogous Map_i task. All subsets will have approximately the same number of instances since the partitioning is performed in chronological basis. The Map phase over each T_i consists of NAC Algorithm. The output from each map task is a binary vector $f_i = \{f_{i1}, \ldots, f_{iD}\}$, where D is the number of features optimized by the NAC Algorithm. The Reduce phase averages all the binary vectors, obtaining a vector x which is defined by the (1).

$$x = \{x1, \ldots, xD\},$$

$$x_j = \frac{1}{m}\sum_{i=1}^{m} f_{ij},\ j \in \{1,2,\ldots,D\} \quad \ldots\ldots\ldots\ldots\ldots(1)$$

This vector is the consequence of the overall NAO process and is used to generate the classified output from the given dataset. The reduce phase is carried out by a single task to reduce the MapReduce overhead [16].

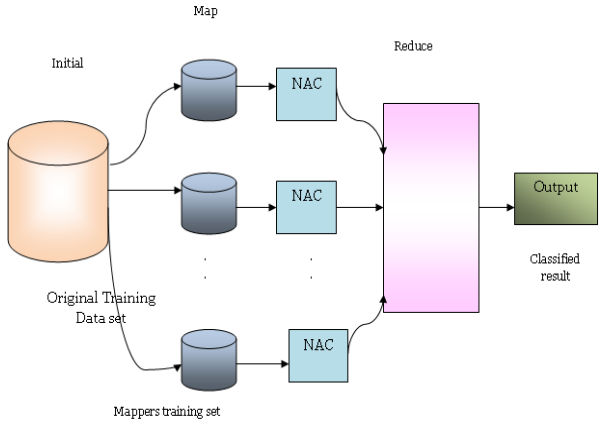

Figure 3: Working of MR-NAC Algorithm

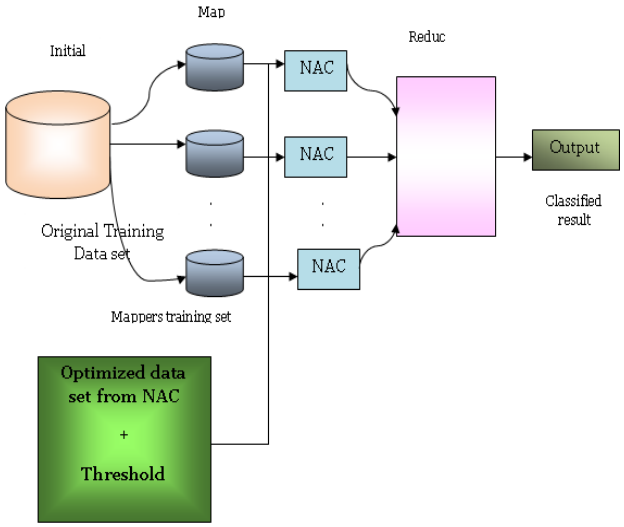

Figure 4: Working of MR-NAC Algorithm along with threshold

A. Dataset reduction with MapReduce:

Once vector **x** is calculated, the objective is to eliminate the less promising features from the original dataset to improve the exactness throughout the classification. An additional MapReduce was designed by binarizing the vector x is using a threshold θ:

$$b = \{x1, \ldots, xD\},$$

$$b_j = \begin{cases} 1, if\ x_j \geq \theta \\ 0, otherwise \end{cases}. \quad \ldots\ldots\ldots\ldots\ldots(2)$$

Vector b designates the features to be selected for the optimized dataset. Each map processes an instance and generates a new one that only contains the features selected in b. Finally, the instances generated are concatenated to form the final reduced dataset which is needed for the final classification. The dataset reduction process, using the result of MR-NAC and an arbitrary threshold is depicted in Figure 3. The final output from the framework is the classified result of the dataset.

IV. EXPERIMENTAL FRAMEWORK AND ANALYSIS

The experiments performed and their results were described in this section. The datasets used for experimentation, hardware and software support required for implementations and results obtained are presented.

A. Datasets and Methods:

In order to analyze the quality of the solution provided by MR-NAC Algorithm, two large binary classification datasets were used. Epsilon dataset which is composed of 500,000 instances with 2000 numerical features was used. Another dataset is the dataset which was used at the data mining competition held in 2014 [ECBDL14], which has 631 features and it is composed of approximately 32 million instances.

The main characteristics of these datasets are summarized in Table 1. For each dataset, the number of instances for both training and test sets and the number of attributes are shown, along with the number of splits in which MR-NAC divided each dataset.

Table 1: Description of the dataset used for experimentation

Dataset	Training instances	Testing instances	Features	Splits	Instances per splits
Epsilon	400 000	100 000	2000	512	~750
ECBDL14	31992815	2987415	631	32768	~1987

Table 2: Parameter values used for experimentation of NAC algorithm

Parameters	Value
Pheromone Level (τ)	0.5
Heuristic Value (η)	0.1
Strength of Pheromone level (α)	1
Strength of Heuristic level (β)	3
Pheromone decay parameter p	0.4
Learning rate	0.1
Momentum	0.7
No. of epochs	25

The parameters for the NAC algorithm are presented in Table 2. In this experimentation, the Original NAC Algorithm and MR-NAC Algorithms are compared based on metrics like Area Under Curve (AUC) and Execution time.

$$AUC = \frac{TPR + TNR}{2}$$

$$TPR = \frac{TP}{(TP + FN)}$$

$$TNR = \frac{TN}{(FP + TN)}$$

B. Hardware and Software Requirements:

The experimentation is carried on a cluster of twenty computing nodes and a master node. Each compute node has the following features:

(i) Processors: 2 x Intel Xeon CPU E5-2620.
(ii) Cores: 6 per processor (12 threads).
(iii) Clock speed: 2.00 GHz.
(iv) Cache: 15 MB.
(v) Network: QDR InfiniBand (40 Gbps).
(vi) Hard drive: 2TB.
(vii) RAM: 64 GB.

The Hadoop NameNode and JobTracker which is the master processes are hosted in the master node. The NameNode manages the HDFS by synchronizing the slave machines by means of their relevant DataNode process. TaskTracker, which execute the MapReduce Framework, is managed by JobTracker. Spark follows the related configuration. The software used for experimentation are:

(i) MapReduce implementation: Hadoop 2.0.0-cdh4.7.1. MapReduce 1 (Cloudera's open-source Apache Hadoop distribution).
(ii) Spark version: Apache Spark 1.0.0.
(iii) Maximum maps tasks: 320(16pernode).
(iv) Maximum reducer tasks: 20 (1 per node).
(v) Operating system: CentOS 6.6.

C. Experimentation:

This section explains the result obtained during the experimentation. The execution time taken over different datasets are given in Table 3 and Table 4. Figure 5 depicts the results for execution time for two different datasets which is tested on NAC Algorithm and MR-NAC Algorithm. The execution time for NAC Algorithm is increased with enhance in the number of instances and with the instance of 400000, the execution time is not listed as the system is unable to execute such large number of instances. The MR-NAC Algorithm has shown tremendous performance in executing different number of instances. Even with increase in the number of instances, the implementation time is increased noticeably and the system even performs with the 400000 instances. Similar results are also observed with ECBDL14 dataset.

Table 3: Execution time taken for Epsilon dataset with different number of instances

Instances	Time taken for execution (in seconds) (Epsilon Dataset)	
	NAC Algorithm	MR-NAC Algorithm
1000	254	185
2000	1458	196
5000	5667	251
10000	12546	352
20000	25142	378
50000	35782	395
400000	....	412

Table 4: Execution time taken for ECBDL14 dataset with different number of instances

Instances	Time taken for execution (in seconds) (ECBDL14)	
	NAC Algorithm	MR-NAC Algorithm
1000	312	214
2000	523	208
5000	1258	325
10000	12859	358
20000	27456	398
50000	35402	412
400000	125620	421
1000000	.....	452
5000000	.....	486
10000000	.....	492
30000000	.....	512

Table 5 and 6 displays the results obtained for AUC measurement. It is observed that with increase in threshold the features have been reduced also the AUC value is increased. From Table 5, the AUC value is 0.8299 for NAC Algorithm when the threshold is 0.65 for epsilon dataset. The number of feature is also reduced to 110 and MR-NAC algorithm has given an AUC value of 0.92 and 0.91 in training and testing respectively. For ECBDL14 Dataset, the features also decreased as same for the epsilon dataset and training and testing AUC value is also considerably increased with the reduction in the number of features to 46 with the threshold value of 0.65. Figure 6 shows the assessment of the dwindle in the number of features with the increase in the threshold value.

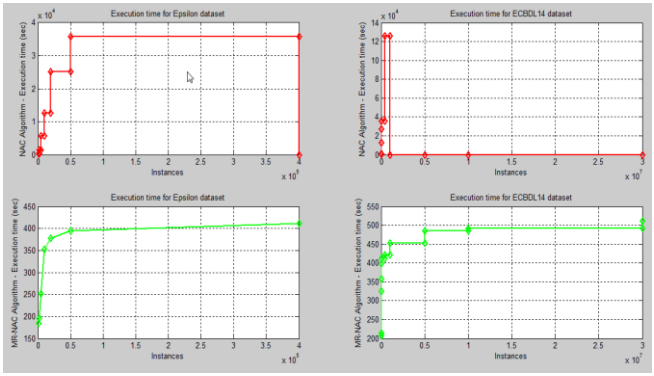

Figure 5: Execution time comparison for Epsilon and ECBDL14 Dataset

Table 5: AUC measurement for Epsilon dataset

Threshold	Features	NAC Algorithm		MR-NAC Algorithm	
		Training	Testing	Training	Testing
0.00	2000	0.7523	0.7546	0.792	0.798
0.55	721	0.8264	0.8321	0.8521	0.8545
0.60	332	0.8298	0.8256	0.8621	0.8921
0.65	110	0.8299	0.8299	0.9201	0.910

Table 6: AUC Measurement for EDBDL14 Dataset

Threshold	Features	NAC Algorithm		MR-NAC Algorithm	
		Training	Testing	Training	Testing
0.00	631	0.8263	0.8278	0.8621	0.8521
0.55	224	0.862	0.8697	0.8756	0.8723
0.60	107	0.856	0.842	0.8456	0.8624
0.65	46	0.881	0.8804	0.953	0.9523

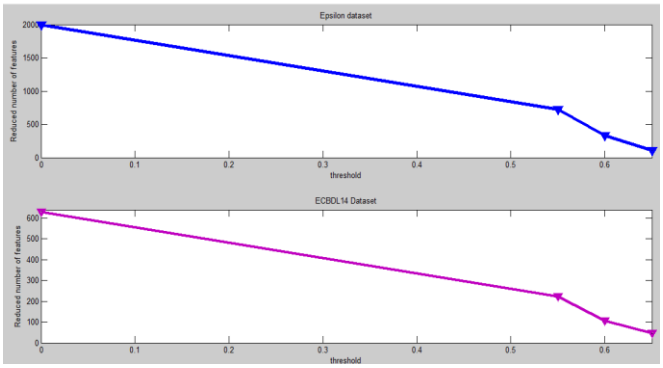

Figure 6: Comparison of reduction in the number of features with increase in threshold value for Epsilon and ECBDL14 Dataset

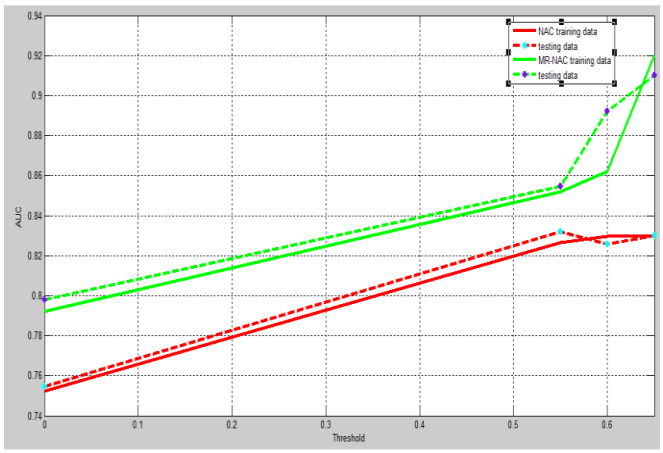

Figure 7: AUC measurement for Epsilon dataset

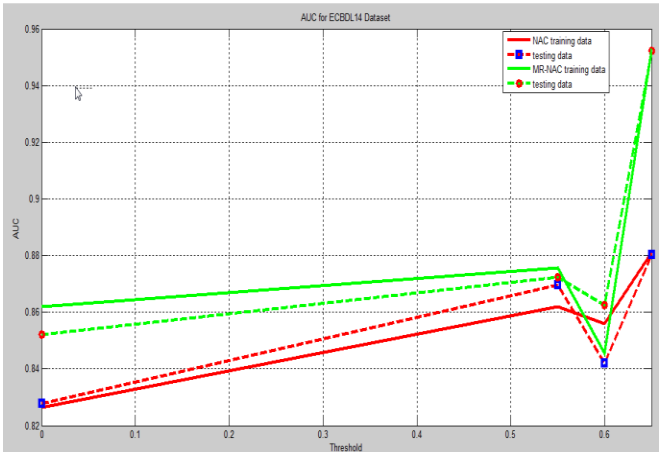

Figure 8: AUC measurement for ECBDL14 dataset

V. CONCLUSION

This paper presents MR-ACO, an Evolutionary Classification algorithm which is designed upon the MapReduce paradigm which is intended to preprocess and classify the big data sets. It was done so as to increase the scalability of the data set during the classification. This algorithm is implemented on Apache Hadoop. Two different large dataset have been used for experimentation. The measurement metrics which are used to evaluate performance have also shown that the scalability of the dataset is increased since the classification accuracy is increased and the execution time is decreased for increase in the number of instances. And also with increase in the threshold value, the number of features also decreased, which results in the more accuracy in classification. The future direction of this work is to implement unstructured data, (i.e) the text data in the MapReduce paradigm using different technique and to analyze the results obtained.

REFERENCES

[1] E. Alpaydin, "Introduction to Machine Learning", MIT Press, Cambridge Mass, USA, 2ND Edition, 2010.

[2] E. Merelli, M. Pettini and M. Rasetti, "Topology driven modelling: the IS metaphor", Natural Computing, Vol. 14, Issue 3, pp 421-430, 2015.

[3] Prakash Singh , "*Efficient Deep Learning for Big Data: A Review*", International Journal of Scientific Research in Computer Science and Engineering, Vol.4, Issue.6, pp.36-41, 2016.

[4] A. Fern´andez, S. del R´ıo,V.L´opez, "Big data with cloud computing: an insight on the computing environment, MapReduce, and programming frameworks," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. 4, Issue 5, pp.380–409,2014.

[5] S. Sakr, A. Liu, D. M. Batista, and M. Alomari, "A survey of large scale data management approaches in cloud environments," IEEE Communications Surveys and Tutorials, Vol.13,Issue.3, pp.311–336, 2011.

[6] Bacardit and X. Llor`a, "Large-scale data mining using genetics-based machine learning," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. 3, Issue.1, pp.37–61,2013.

[7] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM,Vol.51, Issue.1, pp. 107–113, 2008.

[8] J. Dean and S. Ghemawat, "Map reduce: a flexible data processing tool," Communications of the ACM,Vol.53, Issue.1,pp.72–77, 2010.

[9] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," In Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03), pp. 29–43, October 2003.

[10] M. Snir and S. Otto, "MPI—The Complete Reference: The MPI Core", MIT Press, Boston, Mass, USA, 1998.

[11] W. Zhao, H. Ma, and Q. He, "Parallel k-means clustering based on MapReduce, In Cloud Computing, M. Jaatun, G. Zhao, and C. Rong, Eds., Vol. 5931 of Lecture Notes in Computer Science, pp. 674–679, Springer, Berlin, Germany, 2009.

[12] A. Srinivasan, T. A. Faruquie, and S. Joshi, "Data and task parallelism in ILP using MapReduce," Machine Learning, Vol.86, Issue.1, pp.141–168, 2012.

[13] H. He, E.A. Garcia, "Learning from imbalanced data", IEEE Transaction of Knowledge Enginnering, Vol. 21, Issue. 9, pp 1263-1284, 2009.

[14] Y. Sun, A.K.C. Wong, M.S. Kamel, "Classification of imbalanced data: a review", International Journal of Pattern Recognition and Artificial Intelligence, Vol 23, Issue 4, pp 687-719, 2009.

[15] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM, Vol.51, Issue.1, pp. 107–113, 2008.

[16] J. Dean and S. Ghemawat, "Map reduce: a flexible data processing tool," Communications of the ACM, Vol.53, Issue.1, pp.72–77, 2010.

[17] Daniel Peralta, Sara del Río,Sergio Ramírez-Gallego, Isaac Triguero, Jose M. Benitez, and Francisco Herrera, "Evolutionary Feature Selection for Big Data Classification: A MapReduce Approach", Hindawi Publishing Corporation, Mathematical Problems in Engineering, Vol 2015, pp,. 1-11, 2015.

[18] Sara del Río , Victoria López, José Manuel Benítez, Francisco Herrera, "On the use of MapReduce for imbalanced big data using Random Forest", Information Sciences, Vol 285, pp 112–137, 2014.

[19] A. Yadav, V.K. Harit, "*Fault Identification in Sub-Station by Using Neuro-Fuzzy Technique*", International Journal of Scientific Research in Computer Science and Engineering, Vol.4, Issue.6, pp.1-7, 2016

Authors Profile

Mrs.M.S.Saranya pursed Bachelor of Computer Science from Bharathidasan University in Thiruchirapalli in 2008, and Master of Computer Application from SRM University Chennai in year 2011, and M.Phil in St.Peter's University chennai in the year of 2012..She has worked as an Assistant Professor in SRM University in the Department of Computer Science for 2 years. Now Currently doing her Ph.D in Khadir Mohideen College Adhirampattinam affliated to Bharathidasan University Thiruchirapalli. Her main research work focuses on Cloud Security and Privacy, Big Data Analytics, Data Mining, IoT and Computational Intelligence based on evolutionary algorithms education. .

Mr.N.Jayaveeran M.Sc.,M.phil.,Ph.D., He is working as an Associate Professor & HOD in Khadir Mohideen College Adhirampattinam Affliated with Bharathidasan UniversityThiruchirapalli .His main research work focuses on Cryptography Algorithms, Network Security, Cloud Security and Privacy, Big Data Analytics, Data Mining, IoT and Computational Intelligence based education. He has 30 years of teaching experience and 12 years of Research Experience.

