ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IMPROVED ERROR REDUCED EXTREME LEARNING MACHINE (IERELM) CLASSIFIER FOR BIG DATA ANALYTICS

B. RAJENDRAN[1] and Prof (Dr.) SARAVANAN VENKATARAMAN[2]

[1]Research and Development Centre, Ph. D Scholar, Bharathiar University, Coimbatore, Tamil Nadu, India

[2]Department of Computer Science, College of Computer and Information Sciences, Majmaah University, Kingdom of Saudi Arabia.

E-Mail: [1]rajendran.bhojan@gmail.com, [2] s.tirumalai@mu.edu.sa

ABSTRACT

Nowadays the term 'Big Data Analytics' has been talked everywhere due to the advancement of information technology, evolution of computer applications, mobile communications and much more. This paves the way for doing lot of research in this big data arena. It is common to known that classification is one among the thrust research dimension in the field of big data particularly in the field of data analytics. Machine learning algorithms have lot of scope on analytics; particularly extreme learning machine is a kind of feed-forward neural network that is commonly applied for performing classification task. This machine learning ELM algorithm has single layer hidden nodes with randomly assigned weights for connecting with hidden layer. It is to be noted that feed-forward neural networks in extreme learning machine are poor in updating the weights that leads to performance degradation. Computational complexity is certainly more when applying ELM for big data analytics. This part of doctoral research work inclines high motivation for improving the performance of ELM in terms of reducing the error and we named it as Improved Error Reduced ELM shortly coined as IERELM. The performance of the proposed IERELM mechanism is applied for performing the classification task in KDD Cup 99 multivariate dataset that contains 40,00,020 instances with 42 attributes. Obtained results portrays that the proposed IERELM performs better in terms of detection rate, false alarm rate and elapsed time to perform classification.

Keywords: *Big Data, HACE Theorem, Classification, Extreme Learning Machine, KDD Cup Data Set, Neural Networks.*

1. INTRODUCTION

Big Data analytics fit into spot of handling blended/assortment of data from an assortment of data sources creating corresponding datasets [1]. Thus, the data sets are arranged by their part volumes as well as by their heterogeneity and the conveyed capacity of data. A significant number of data mining systems have been proposed in the related works so as to process such data sets [1]. Tangentially, from conventional brought together data mining frameworks where a solitary learner has full access to the worldwide dataset [2], data mining frameworks typically makes utilization of group learning strategies comprising of a hierarchy of leadership of numerous nearby learners working on subsets of the worldwide dataset [3]. It is critical that analytics assumes a huge part to mine data and to get concealed learning/data about the fundamental dataset. In [16] the creators expressed that HACE theorem produced for Big Data begins with substantial volume, heterogeneous, self-ruling sources with disseminated and decentralized control, and looks to investigate complex and advancing connections among data. The HACE theorem exhibits the above said elements that make an exceptional test for finding valuable information from the Big Data.

This part of doctoral research work deals with network traffic data that has similar characteristics of big data which is the primary task for addressing big data analytics to be more cost effective. As of late, a lot of utilizations experience the ill effects of the big data issue that incorporates network traffic risk investigation, geospatial classification and big business anticipating. Interruption recognition and forecast are thought to be time responsive applications furthermore it needs profoundly productive big data systems to set out upon the issue on the go.

A percentage of the as of late rising innovations are likewise help to perform big data investigation on a few applications, for example, Hadoop Distributed File Systems (HDFS) and Hive database [5] are actualized to determine research issues like big data classification. Then again the applications likewise needing constant development in the big data space those are most likely experience the ill - effects of the big data issues.

The proposed work aims in design and development of improved error reduced extreme learning machine to perform classification tasks in big data. In this research the KDD cup 99 dataset is chosen. The first important deficiency in the KDD data set is the huge number of redundant records. Analyzing KDD train and test sets, we found that about 78% and 75% of the records are duplicated in the train and test set, respectively. This large amount of redundant records in the train set will cause learning algorithms to be biased towards the more frequent records, and thus prevent it from learning unfrequented records which are usually more harmful to networks such as U2R attacks. The existence of these repeated records in the test set, on the other hand, will cause the evaluation results to be biased by the methods which have better detection rates on the frequent records. Hence there is a wide scope of research for reducing the training errors and hence improved error reduced extreme learning machine classifier is proposed in this research work.

This paper is organized as follows. Section 1 briefs the introduction to big data, problem statement and scope of research. Section 2 shortly describes the related works carried out. Section 3 emphasizes the proposed research work. Section 4 examines the experimental results. Section 5 concludes the paper with future scope of research work.

2. RELATED WORKS

In 2011, Mckinsey's report [17] defined big data as ''datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze.'' This definition is subjective and does not define big data in terms of any particular metric. Big data comes with lot of scope for researchers due to its formidable challenges in dealing with large-scale data sets. Primarily, the perpendicular volume and dimensionality of data make it often impossible to run analytics and traditional inferential or knowledge based methods using standalone processors, e.g., [12] and [13]. Decentralized learning with parallelized multicores is preferred [14], [15], while the data themselves are stored in the cloud or distributed file systems as in MapReduce/Hadoop [16]. As a result, there is an imperative need to explicitly account for the storage, query, and communication number. Machine learning algorithms are proposed for the classification task of network intrusion traffic [6-10] which is an example of big data analytics. In this research, an auxiliary development on the performance of ELM is chosen that tends for the proposed big data analytics problem. The basic short coming for Big Data applications is to see the insights the large volumes of data and extract useful information or knowledge for future actions [11]. In many situations, the knowledge extraction process has to be very efficient and close to real time because storing all observed data is nearly infeasible.

The proposed improved error reduced extreme learning machine classifier The Error Minimized ELM, namely IERELM, is designed to update in an iterative manner. Though the IERELM takes more time for training the overall complexity is reduced by adding one new node is to the existing k hidden nodes network.

3. PROPOSED WORK

3.1. Improved Error Reduced ELM (IERELM)

IERELM is based on the **H** which is full column rank, then $\mathrm{H}^{\dagger} = (H^T H)^{-1} H^T$. However, there exist situations when **H** is ill-conditioned ("almost" not full column rank) as *K* increases though *K* < *N* still holds. One consequence of it is that, even if the matrix $\mathrm{H}^{\mathrm{T}}\mathrm{H}$ is invertible, a computer algorithm may be unsuccessful in obtaining an approximate inverse, and if it does obtain one it may be numerically inaccurate. Divergence will inevitably occur if EM-ELM or QRI-ELM continues to be applied in this case.

The hidden-layer output matrix **H** can be decomposed as **H** = **Q** · **R**, where **Q** is an orthogonal matrix and **R** isan upper triangular matrix, respectivelyThe proposed improved error reduced ELM is designed for updating $\mathrm{H}^{\dagger}{}_{k+1}$ in an iterative manner by making use of $\mathrm{H}^{\dagger}{}_{k}$, instead of H_{k+1}. This update will be conducted when one new node is added to the existing *k* hidden nodes network. It is presumed that h_{k+1} is the new column in H_{k+1}. It can be found in the position from *(k +* 1*)*th neuron, U_{k+1} is located at the upper part of

$H^{\dagger}{}_{k+1}$, and is lower part of $H^{\dagger}{}_{k+1}{}^{2}$. The important steps during the process of IERELM are portrayed below:

$$D_{k+1} = \frac{h^{T}{}_{k+1}(I - H_k H^{\dagger}{}_{K})}{h^{T}{}_{k+1}(I - H_k H^{\dagger}{}_{K})h_{k+1}} \quad (1)$$

$$U^{k+1} = H^{\dagger}{}_{K}(I - h_{k+1}D_k) \quad (2)$$

$$H^{\dagger}{}_{k+1} = \begin{bmatrix} U_{k+1} \\ D_{k+1} \end{bmatrix} \quad (3)$$

It is noteworthy that the training time of IERELM will be comparatively lesser than that of conventional ELM. On the other hand, by making slight modification in IERELM the computational complexity can further be reduced. This is done by QR factorization method.

$$D^{\dagger}{}_{k+1} = \frac{h^{t}{}_{k+1} - h^{t}{}_{k+1}H_k H^{\dagger}{}_{k}}{h^{t}{}_{k+1}h_{k+1} - h^{t}{}_{k+1}H_k H^{\dagger}{}_{k}h_{k+1}} \quad (4)$$

$$U_{K+1} = H^{\dagger}{}_{k} - H^{\dagger}{}_{k}h_{k+1}D_k \quad (5)$$

$$\beta_{K+1} = \begin{bmatrix} U_{K+1} \\ D_{K+1} \end{bmatrix} T \quad (6)$$

Although it is claimed that the training time of IERELM is less than that of ELM, it can be observed that is not true with simple analysis if using the above formula directly. The most computational consuming step of IERELM is multiplication of H_k and $H\dagger_k$, with complexity, $O(kN^2)$, even more than $O(k^2N)$ in conventional ELM. Here, the hidden-layer output matrix H can be decomposed as H = Q · R, where Q is an orthogonal matrix and R is an upper triangular matrix. Hence $R^{-1}{}_{k+1}$ is calculated based on $R^{-1}{}_{k}$ as the number of hidden nodes *k* increases, followed by getting output weights $\hat{\beta}_{k+1}$ by $\hat{\beta}_{k}$ and $R^{-1}{}_{K}$. By deploying the above said logic computational complexity will then be reduced. The QR decomposition is carried out as shown below

Let $H = [h_1, h_2, \ldots, h_k], Q = [q_1, q_2, \ldots, q_k]$ and

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1k} \\ & r_{22} & \cdots & r_{2k} \\ & & & \vdots \\ & & \ddots & \\ & & & r_{kk} \end{bmatrix} \quad (7)$$

Since H=Q.R, we have,

$[h_1, h_2, \ldots, h_k], Q = [q_1, q_2, \ldots, q_k]$.

$$\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1k} \\ & r_{22} & \cdots & r_{2k} \\ & & & \vdots \\ & & \ddots & \\ & & & r_{kk} \end{bmatrix} \quad (8)$$

Hence,

$$h_1 = q_1.r_{11}$$
$$h_2 = q_1.r_{12} + q_2.r_{22} \quad (9)$$
$$\vdots$$
$$h_k = q_1.r_{1k} + \ldots + q_k.r_{kk} \quad (10)$$

It is also to be noted that

$$Q^T Q = I \ \Rightarrow \ q^T i \, q_i = 1 \text{ and } q^T i \, q_j = 0 \text{ when } i \neq j,$$

$$\vdots$$

$$so \quad h_1^T h_1 = r_{11} q_1^T q_1 r_{11} = r_{11}^2 (11)$$

At the final stage it is considered to set

$$\delta h_k = q_k r_{kk} = h_k - (q_1 r_{1k} + \cdots + q_{k-1} r_{k-1,k}),$$

$$\text{So } \ r_{kk} = \sqrt{\delta h^T k}\, \delta h_k \text{ and } q_k = \delta h_k / r_{kk} (12)$$

It is presumed that *(k* + 1*)*th hidden node is added to the *k* nodes network. Once and while input weights are generated, an additional column h_{k+1} added to H_k to form $\mathrm{H}_{k+1} = [\mathrm{H}_k \mid \mathrm{h}_{k+1}]$. Accordingly, the QR factorization of H_{k+1} becomes $\mathrm{H}_{k+1} = \mathrm{Q}_{k+1}\,\mathrm{R}_{k+1}$, where $\mathrm{Q}_{k+1} = [\mathrm{Q}_k \mid \mathrm{q}_{k+1}]$ and

$$R_{k+1} = \left[\frac{R_K \mid \delta r_{K+1}}{0 \quad \mid r_{k+1,k+1}}\right] = \left[\begin{array}{ccccc|c} r_{11} & r_{12} & \cdots & r_{1k} & & r_{1,k+1} \\ & r_{22} & \cdots & r_{2k} & & r_{2,k+1} \\ & & \ddots & \vdots & & \vdots \\ & & & r_{kk} & & r_{k,k+1} \\ \hline & & & & & r_{k+1,k+1} \end{array}\right]$$

As per the Equation (12), we have

$$\delta r_{k+1} = Q^T{}_k\, h_{k+1} \,(13)$$

$$\delta h_{k+1} = q_{k+1}, r_{k+1,k+1}$$

$$= h_{k+1} - (q_1 r_{1,k+1} + \cdots + q_k, r_{k.k+1})$$

$$= h_{k+1} - Q_k \delta r_{k+1} (14)$$

$$r_{k+1,k+1} = \sqrt{\delta h^T{}_{k+1}\,\delta h_{k+1}} \quad (15)$$

$$q_{k+1} = \delta h_{k+1} / r_{k+1,k+1} \qquad (16)$$

$$R_{k+1}^{-1} = \left[\frac{R_k|\delta r_{k+1}}{o|r_{k+1,k+1}}\right]^{-1}$$

$$= \left[\frac{R_K^{-1}|-R_k^{-1}\delta r_{K+1} r_{k+1,k+1}^{-1}}{o|r_{k+1,k+1}^{-1}}\right](17)$$

Finally,

$$\hat{\beta}_{k+1} = \mathrm{H}^{\dagger}{}_{k+1}\mathrm{T} = \mathrm{R}^{-1}{}_{k+1}\mathrm{Q}^{\mathrm{T}}{}_{k+1}\mathrm{T}$$

$$= \left[\frac{R^{-1}{}_K \quad |-R^{-1}{}_k\,\delta\, r_{K+1} r^{-1}{}_{k+1,k+1}}{0|\, r^{-1}{}_{k+1,k+1}}\right] \cdot \left[\frac{Q^T{}_k}{q^T{}_{k+1}}\right] .T$$

$$= \left[\frac{\hat{\beta}_K \quad - R^{-1}{}_K\,\delta\, r_{K+1}\beta_{K+1}}{\beta_{k+1}}\right](18)$$

$$\text{Where } \beta_{k+1} = \mathrm{q}^{\mathrm{T}}{}_{k+1}\mathrm{T}/\mathrm{r}_{k+1,k+1}$$

At this juncture the proposed IERELM is summarized as follows.

Given a set of training data, it is presumed that a single hidden layer neural network is to be trained, starting with 1 hidden node up to maximum number of hidden nodes $\mathrm{K}_{\max}$, and the expected learning accuracy $\in$. Note that it is $\mathrm{R}^{-1}{}_k$ instead of R_k used as an intermediate variable in the whole recursive process, hence $\mathrm{P}_k = \mathrm{R}^{-1}{}_k$ is introduced in the procedure. Thewhole process is demonstrated in Algorithm.

Algorithm - IERELM.

1. Randomly generate the single hidden node input weights set $\{\omega_{i1}\}^{\mathrm{I}}{}_{i=0}$
2. Calculate the hidden-layer output matrix $\mathrm{H}_1 (= \mathrm{h}_1)$
3. Calculate the inverse of $\mathrm{R}_1 : \mathrm{P}_1 (= \mathrm{p}_{11} = \dfrac{1}{\mathrm{r}_{11}}) = (\mathrm{h}^{\mathrm{T}}{}_1 \mathrm{h}_1)^{-\frac{1}{2}}$
4. Calculate $\mathrm{Q}_1 = \mathrm{q}_1 = \mathrm{p}_{11}\mathrm{h}_1$
5. Calculate the output weight $\hat{\beta}_1 (= \beta_1) = \mathrm{P}_1 \mathrm{Q}^{\mathrm{T}}{}_1 \mathrm{T}$
6. while k = 1 to K_{max} And $E(H_k) = \left|\left|H_k\beta_k - T\right|\right| > \in do$
7. A new hidden node is added, the corresponding input weights-set are generated $\{\omega^i{}_{,k+1}\}^{\mathrm{I}}{}_{i=0}\, i$=0, and the corresponding h_{k+1} are calculated.
8. Update the following variables in sequence:

$$\delta r_{k+1} = Q^T{}_k\, h_{k+1} \,(19)$$

$$\delta h_{k+1} = h_{k+1} - Q_k \delta\, r_{k+1} \,(20)$$

$$p_{k+1,k+1}\left(= \frac{1}{r_{k+1,k+1}}\right) = (\delta h^T{}_{k+1,k+1}\delta h_{k+1,k+1})^{-\frac{1}{2}} (21)$$

$$\mathrm{q}_{k+1} = \mathrm{p}_{k+1,k+1}\delta\mathrm{h}_{k+1,k+1} \qquad (22)$$

$$\beta_{k+1} = \mathrm{p}_{k+1,k+1}\mathrm{q}^{\mathrm{T}}{}_{k+1}\mathrm{T} \qquad (23)$$

$$\beta_{k+1} = \left[\frac{\hat{\beta}_K \quad - P_K\,\delta\, r_{K+1}\beta_{K+1}}{\beta_{k+1}}\right] (24)$$

$$p_{k+1} = \left[\frac{P_K \quad | -P_k \, \delta \, r_{K+1} p_{k+1,k+1}}{0 | \, p_{k+1,k+1}} \right] (25)$$

$$Q_{k+1} = \left[Q_k \mid q_{k+1} \right]$$

```
9.  $k \leftarrow k + 1$
10. End while.
```

4. EXPERIMENTAL RESULTS

This section presents experimental results that signify classification efficiency over the conventional KDD Cup dataset. KDD Cup dataset is present in UCI KDD archive. This dataset has four gigabytes of compressed binary TCP dump data from seven weeks of network traffic, which was processed into about five million connection records, among which we randomly select 50000 records asthe training dataset. Each connection record is labelled as a “normal” connection or as an “attack”. The performance of the algorithms such as ELM and the proposedIERELM algorithms are evaluated using the metrics such as detection rate, false alarm rate and time taken for classification.

The proposed IERELM and conventional ELM has been implemented in a personal computer that has 2.4 GHz processor, 2 GB RAM with L2 cache. MATLAB tool is used to write the source code for the both IERELM and ELM algorithms.

Since the proposed IERELM automatically determine the number of hidden nodes in generalized single-hidden-layer feed forward networks (SLFNs), the proposed mechanism is capable enough to add random hidden nodes to SLFNs one by one. The computational complexity of this approach is analyzed. This proposed mechanism thus improves the performance of the IERELM classifier.

The values of simulation results have been presented in Table1.It can be perceived that the proposed IERELM classifier has better detection rate(as shown in Figure 1), lesser false alarm rate (as shown in Figure 2) with comparably reduced timed taken for classification (as shown in Figure 3).It is very significant that from the Figure 4 and Figure 5 that the proposed IERELM has better true positive rate (sensitivity) that measures the proportion of positives that are correctly identified as such and true negative rate (specificity) measures the proportion of negatives that are correctly identified as such.

It can be observed from the results that the overall performance of the IERELM is improved than that of traditional ELM. As far as detection rate is concerned, ELM is obtains 71.3 % of detection rate whereas the proposed IERELM obtains 85.7 % detection rate of attacks. It is evident that around 14.4 % of the detection rate is improved. This is because of the error reduction in training the IERELM classifier. The false alarm rate of ELM is 28.7 % and for IERELM it is 14.3 %. A significant reduction of false alarm rate is done by IERELM which proves that the error rate has been reduced and significant difference of 14.4% is achieved. Even though IERELM consumes more complexity for training the dataset, the overall elapsed time for performing the classification task is reduced when compared to ELM. From the results it is clear that IERELM consumes 3092 seconds which is lesser than that of ELM which consumes 4863 seconds. The sensitivity and specificity metrics are also taken into account and it is evident that IERELM (sensitivity – 91.62, specificity – 28.57) performs better than that of ELM (sensitivity - 81.21, specificity - 22.86).

5. Conclusions and Future Scope of Research

Big data deals about large volume data sets that are complex in nature which also do contain multiple autonomous sources. Previous available technologies could not cope up the storage and processing of such huge data and hence it leads to the concept of big data which is a tough task for the stakeholders for to identifying accurate data from huge datasets. As a result a mechanism is required to the users from large datasets in less complex way. Extreme learning machine is a machine learning classifier that makes use of single layer feed-forward neural network architecture. This part of doctoral research proposed improved error reduced extreme learning machine classifier. The proposed classifier is applied to the of the big data analytics problem called network intrusion detection. Performance metrics such as detection rate, false alarm rate and time taken for classification.

The proposed IERELM has better generalization performance than the ELM. The proposed IERELM can be commercially built for network intrusion detection logs for identifying the attacks over the networks. The primary advantage of IERELM is that it is suitable for erroneous big datasets such as KDD Cup 99 dataset etc. IERELM also has disadvantages. It takes more time for training but the overall classification time is reduced.

Simulations are carried out and the results portrays that the proposed IERELM performs better when compared to the ELM.

REFERENCES

[1] B. Park and H. Kargupta, "Distributed data mining: Algorithms, systems, and applications", *Distributed data mining handbook*, pp. 341-358, 2002.

[2] M. Kantardzic, "Data mining: Concepts, models, methods, and algorithms", *Wiley-IEEE Press*, 2011.

[3] P. Chan and S. Stolfo, "Experiments on multistrategy learning by meta-learning", *CIKM* '93, 1993.

[4] S. Agrawal, V. Narasayya and B. Yang, "Integrating vertical and horizontal partitioning into automated physical database design", *SIGMOD '04*, 2004.

[5] T. White, Hadoop: The definitive guide. *O'Reilly Media*, 2012.

[6] S. Carlin, K. Curran, "Cloud computing technologies", *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, Vol.1, No. 2, pp. 59-65, 2012.

[6] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica* 31, 249-268, 2007.

[7] P. Laskov, C. Schafer and I. Kotenko, "Intrusion detection in unlabeled data with quarter-sphere support vector machines," *In Proc. of the DIMVA Conference*, 71-82, 2004.

[8] G. Huang, H. Chen, Z. Zhou, F. Yin and K. Guo, "Two-class support vector data description," *Pattern Recognition*, 44, 320-329, 2011.

[9] I. Corona, G. Giacinto and F. Roli, "Intrusion detection in computer systems using multiple classifier systems," *Studies in Computational Intelligence (SCI)* 126, 91-113, 2008.

[10] G. Giacinto, R. Perdisci and F. Roli, "Network intrusion detection by combining one-class classifier," *In: F. Roli and S. Vitulano (Eds.) ICIAP* 2005, LNCS 3617, 58-65, 2005.

[11] A. Rajaraman and J. Ullman, Mining of Massive Data Sets.*Cambridge Univ. Press*, 2011.

[12] T. Bengtsson, P. Bickel, and B. Li, "Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems," in Probability and Statistics: Essays in Honor of David A. Freedman. Beachwood, OH: IMS, 2008, vol. 2, pp. 316–334.

[13] M. I. Jordan, "On statistics, computation and scalability," *Bernoulli*, vol. 19, no. 4, pp. 1378–1390, 2013.

[14] D. P. Bertsekas and J. N. Tsitsiklis, "Parallel and Distributed Computation: Numerical Methods". *Belmont, MA: Athena Scientific*, 1999.

[15] P. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *J. Mach. Learn. Res.*, vol. 11, pp. 1663–1707, May 2010.

[16] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in *Proc. Symp. Operating System Design and Implementation, San Francisco, CA*, 2004, vol. 6, p. 10.

[17] J. Manyika et al., Big data: The Next Frontier for Innovation, Competition, and Productivity. San Francisco, CA, USA: *McKinsey Global Institute*, 2011, pp. 1–137.

Table 1: Simulation Results

Detection Rate (%)		False Alarm Rate (%)		Time Taken for Classification (Seconds)		Sensitivity (%)		Specificity (%)	
ELM	IERELM	ELM	IERELM	ELM	IERELM	ELM	IERELM	ELM	IERELM
71.3	85.7	28.7	14.3	4863	3092	81.21	91.62	22.86	28.57

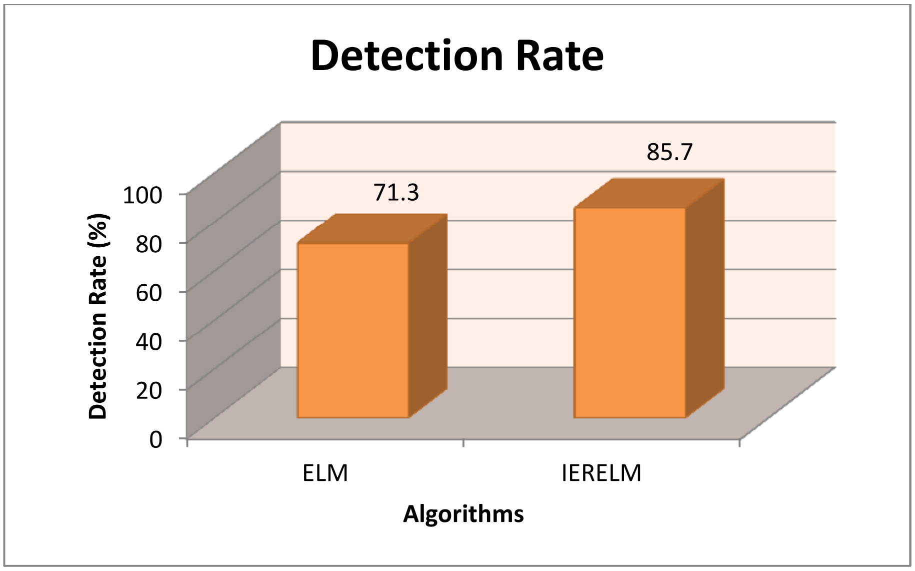

Figure 1: Detection Rate

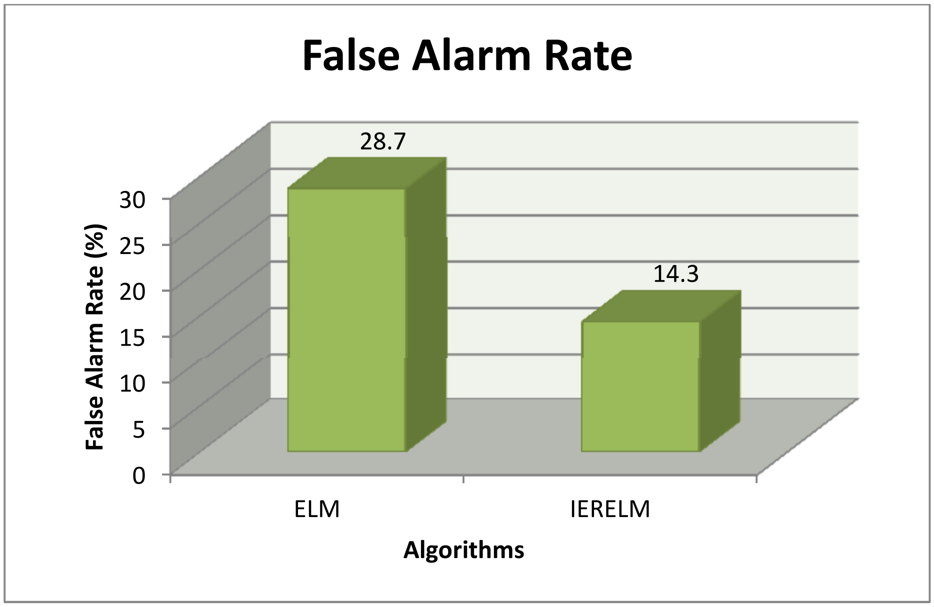

Figure 2: False Alarm Rate

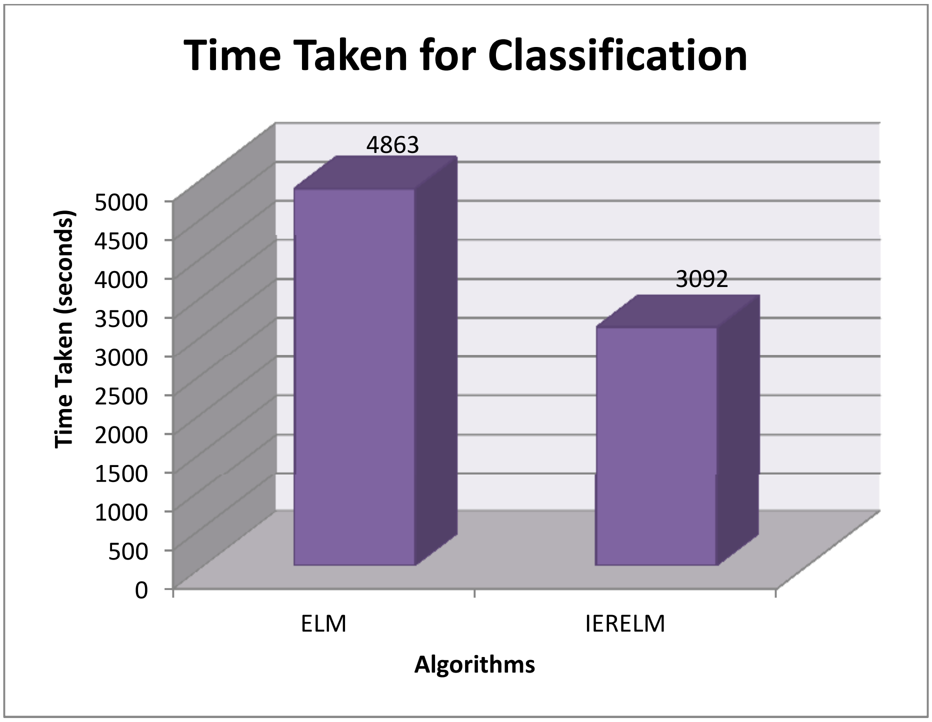

Figure 3: Time Taken for Classification

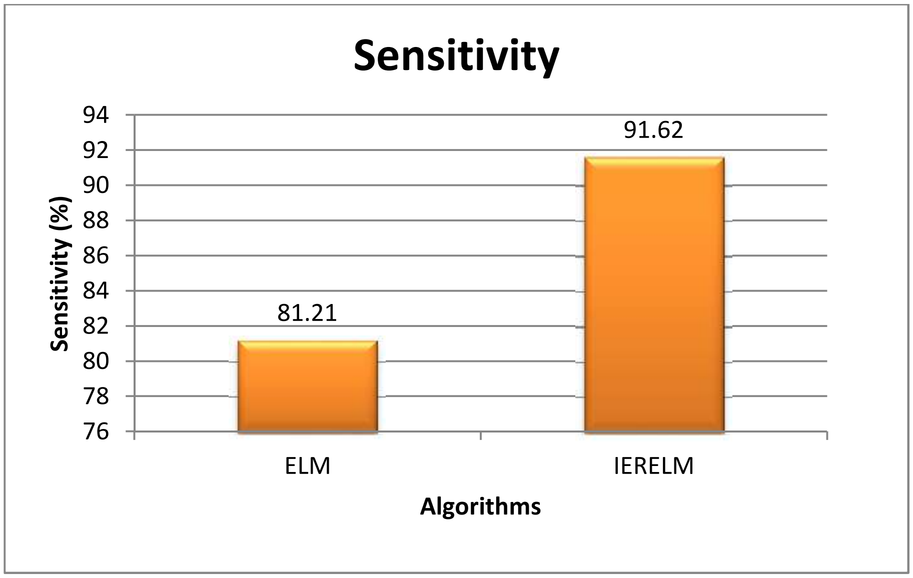

Figure 4: Sensitivity

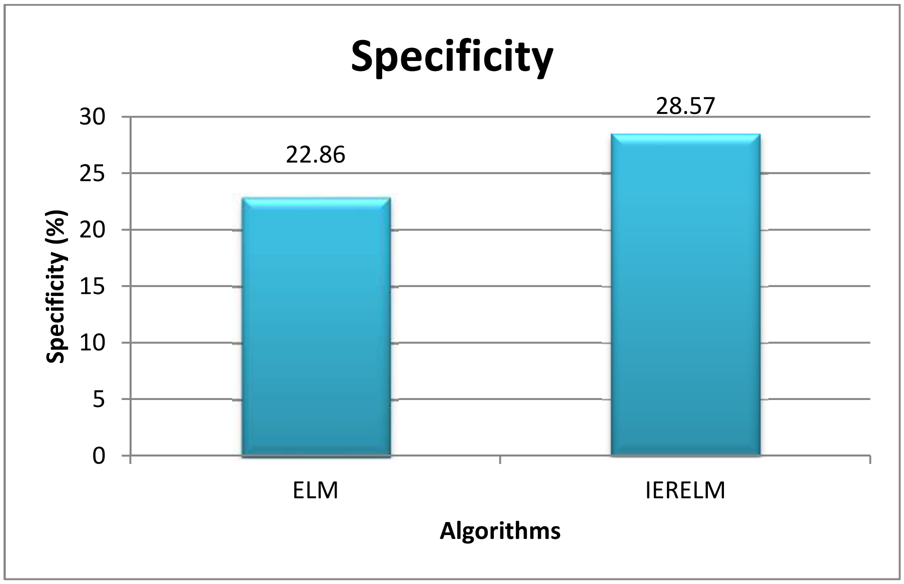

Figure 5: Specificity