Predictive Ensemble Modelling - Experimental Comparison of Boosting Implementation Methods

Vincent F. Adegoke, Daqing Chen

Computer Science and Informatics, School of Engr'g
London South Bank University
London, United Kingdom
adegokev@lsbu.ac.uk, chend@lsbu.ac.uk

Safia Banissi, Ebad Banissi

Computer Science and Informatics, School of Engr'g
London South Bank University
London, United Kingdom
barikzas@lsbu.ac.uk, banisse@lsbu.ac.uk

Abstract — **This paper presents the empirical comparison of boosting implementation by reweighting and resampling methods. The goal of this paper is to determine which of the two methods performs better. In the study, we used four algorithms namely: Decision Stump, Neural Network, Random Forest and Support Vector Machine as base classifiers and AdaBoost as a technique to develop various ensemble models. We applied 10-fold cross validation method in measuring and evaluating the performance metrics of the models. The results show that in both methods the average of the correctly classified and incorrectly classified are relatively the same. However, average values of the RMSE in both methods are insignificantly different. The results further show that the two methods are independent of the datasets and the base classier used. Additionally, we found that the complexity of the chosen ensemble technique and boosting method does not necessarily lead to better performance.**

Keywords - AdaBoost; ensemble based system; machine learning; resampling; reweighting

I. INTRODUCTION

Adaptive boosting is a family of boosting algorithm. It is a meta-learning technique that operates on several weak classifiers. Boosting has its roots in Probably Approximately Correct (PAC) learning framework that was first introduced by Valiant [1]. However, AdaBoost as a technique was first proposed by Freud and Schapire [2] in 1995.It has received significant attention in recent years due to its ability to train and improve the performance of other classifiers also referred to as weak classifiers.

AdaBoost forms a strong classifier by combining the outputs of the weak classifiers. It has many potential applications [2] [3] [4] and has been used successfully in many areas such as text classification, natural language processing, drug discovery and computational biology [5] vision and object recognition [6] medical diagnosis [7] and industrial chemical fault diagnosis [8]. The primary goal of AdaBoost as an ensemble classifier is to improve the accuracy of the base classifier by constructing ensemble decision rules [1] [2] [3] [9] that produce a strong classifier and perform better when they are combined than random guessing. The details of the algorithm have reported in the literature [2] [4] [9]. However, to address the limitation of the algorithm several variations of the algorithm have been proposed that uses different sigmoid functions that attempts to optimize performance of the algorithm during training.

AdaBoost can train its base classifiers using reweighting or by resampling method. However, only base classifiers that are designed to handle learning by reweighting during learning can use reweighing method. Therefore, many learning algorithms are defaults to boosting by resampling during training. There is no concrete evidence which of these two methods perform better. In boosting by reweighting, AdaBoost assigns weight to each training sample and adjusts the weights based on performance during iteration process. The base classifiers with poor predictions have their weights increased for further training in the next iteration and those with good predictions have their weighs reduced. This process is repeated iteratively until a stopping condition is met. On the other hand, during learning by resampling: weights are not passed to weak classifiers instead the training data is resampled with replacement to reflect the weights change. This method unlike reweighting method does not require very large data. It achieves boosting goal by concentrating on the subsequent classifiers on the samples that the previous weak classifier misclassified thereby adaptively increasing the probability of sampling with replacement of misclassified class for the next classifier.

The final output of either of the training methods is a linear combination of the generated base rules. The decision rules must be a general-purpose algorithm that has practical value in terms of efficiency, resources requirements and adaptable to a broad class of learning problems [4]. In this study we carried out several simulations using boosting by reweighting and boosting by resampling to find out which of the two implementation methods perform better. We used 15 different datasets obtained from the UCI repository, WEKA library and health sector. We applied decision tree, neural network, random forest and support vector machine algorithms as committee of classifiers trained by AdaBoost.

In this paper, we presented and tested several ensemble models by implementing the two methods that AdaBoost can use in training its base classifiers namely reweighting and resampling. The rest of this manuscript is organized as follows: section 1 is a brief description of AdaBoost and related work. Section 2 contains experimental settings. Sections 3 describes the experimental methodology. Section

4 is a brief discussion of results and in section 5 we highlight the conclusion and future work.

II. Overview and Related Work

A. Brief History of AdaBoost

AdaBoost was originally introduced by Freund [3] in 1995 as a classifier that uses induction method. It converts and combines series of weak learners into a better and stronger classifier than randomly selecting classifiers. It was later experimented and tested [2] [10] by the authors and other users. This has led to the introduction of many variants of the algorithm like AdaBoost.M2 [10], LogitBoost [11] SoftBoost [12] BrownBoost [4], etc. that uses different loss or objective functions. Many of these variants were introduced as enhancements [9] to resolve some of the limitations that AdaBoost faces. Some of these variants are currently available for regression and classification problems. In many cases the variants address binary and multi-class predictor problems. Studies show that AdaBoost is sensitive to noisy data and outliers but less susceptible to overfitting problem than other learning algorithms [4] [13]. The generalization error of the base learners can be controlled in terms of its training errors [2]. AdaBoost is greedy in the sense that it builds up a strong classifier incrementally by optimizing the weights and adding one weak classifier at a time during training. However, AdaBoost itself is not a learning algorithm but a meta-learning technique that tends to improve the performance of other algorithms known as the weak learners by weighting or resampling and combining them to form a stronger classifier. Boosting algorithms in general performs two main functions firstly, it chooses the training samples. Secondly it combines the trained multiple weak classifiers into a single and stronger classifier.

In general, AdaBoost algorithm can be visualized as a Neural Network algorithm in which the weak and the strong learners form the hidden and output parts of the model respectively as shown in Fig 1. The activation function that AdaBoost use depend the loss function of the variant; it can be used for classification or regression tasks. AdaBoost works by applying the weak learner sequentially to weighted versions of the data thereby more weight is given to samples that were misclassified in previous rounds. AdaBoost uses the subset of training data in training its base classifiers which are combined to form the final classifier. The weak learner can be any classification or regression algorithm, but it is common to use a CART (Classification and Regression Tree) model. One of the advantages of AdaBoost algorithm is that it does not require the knowledge of the base classifier and can therefore train base learners as all classifiers are designed to handle boosting by resampling. However, the actual performance of boosting is dependent on the data as well as the base classifier used in developing the model.

Like other algorithms, Robert Schapire [4] noted that AdaBoost can fail to perform well under insufficient data when the learning algorithms are set to achieve at least 50% accuracy which is in consistency with the algorithm's theory.

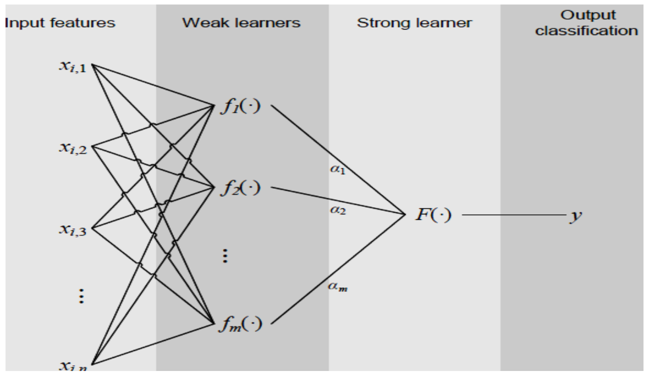


Figure 1 AdaBoost as a Neural Network algorithm with activation functions, hidden and output layers $f_i(.)$ are the base classifiers and F(.) is the summation function that combines the weak classifiers into a strong and more accurate classifier.

B. Previous and Related Work

There have been a few attempts in comparing boosting by reweighting and boosting by resampling methods. Despite this, there is still a disagreement among many authors which of the methods performs better.

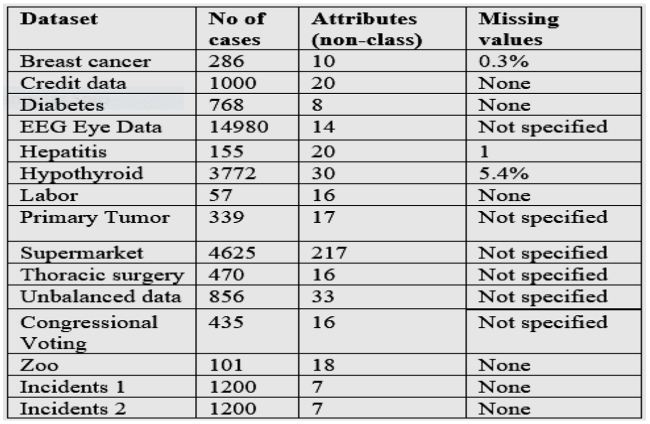
In their empirical study, Bauer and Kohavi [14] observed that Arc-x4 behaves differently than AdaBoost if reweighting is used instead of resampling method which shows a fundamental but not a conclusive difference. In a similar study, Quinlan [15] as cited by Bauer and Kohavi [14] reported better results when using boosting with reweighting method compared with boosting with resampling method. However, Quinlan's study was only based on implementations of algorithms that support samples by weighted instances. This, according to the author, is a direct implementation of the reweighting theory.

In another study, Souza and Matwin [16] used resampling with substitution instead of reweighting approach on many weak classifiers and datasets. The authors found a significant improvement with resampling compared with reweighting method. Contrarily, Botha [17] comparison of the two methods shows that boosting by reweighting can improve performance over resampling. In a similar study carried out by Seiffert *et al.* [18], which compare the two methods, the authors found out that boosting by resampling performs better than boosting by reweighing method.

III. Experimental Setting

A. Datasets Description

In this study, we used AdaBoost.M1 variant and 15 different datasets. The incident datasets were obtained from the London Ambulance Service (LAS). Other datasets used in the study were obtained from UCI [19] repository and WEKA library. The datasets were selected from different application domains based on data types, tasks and attributes to obtain a generalized comparison of the two boosting methods. We used four different types of classification models namely the Decision Stump, Neural Network, Random Forest and Support Vector Machine as base classifiers. We also used stratified 10-fold cross validation method to compare the performance and prediction accuracy of the ensemble models. The details of the datasets used are as presented in Figure 2.



Dataset	No of cases	Attributes (non-class)	Missing values
Breast cancer	286	10	0.3%
Credit data	1000	20	None
Diabetes	768	8	None
EEG Eye Data	14980	14	Not specified
Hepatitis	155	20	1
Hypothyroid	3772	30	5.4%
Labor	57	16	None
Primary Tumor	339	17	Not specified
Supermarket	4625	217	Not specified
Thoracic surgery	470	16	Not specified
Unbalanced data	856	33	Not specified
Congressional Voting	435	16	Not specified
Zoo	101	18	None
Incidents 1	1200	7	None
Incidents 2	1200	7	None

Figure 2 Descriptions of experimental datasets.

B. Base Modes: the Weak Classifiers

Decision Stumps - Boosting with decision stumps is quite popular in data mining and have been shown to achieve better performance compared to unbounded decision trees due to their simplicity. It has also been used in bagging technique. It consists of a one-level decision tree and one internal node which are connected to the terminal nodes. It makes prediction based on the value of a single input feature [20]. Decision stumps have been used as a base classifier in various machine learning models such as Viola-Jones' face detection framework that employs AdaBoost as a main classifier [6].

Artificial Neural Network (ANN) - The artificial neural networks are a family of artificial intelligence models like the biological brain used in solving prediction and classification tasks. ANN is capable of estimating complex and non-linear functions that depend on many inputs. It is generally presented as systems of interconnected neurons that exchange messages between each other. In this study, we used multilayer perception (MLP) with back-propagation which is a popular architecture of ANN [21]. During the simulation, we observed that ANN models took longer time to run compared with other models in all experimental datasets. This could due to the complexity of algorithm's architecture such as the number of neurons, number hidden layers and other parameter settings. Like AdaBoost, ANN connections have numeric weights that can be changed thereby making neural networks adaptive to inputs and capable of learning and solving complex problems [20].

Random Forest – Random forest is an ensemble of several decision trees. Like AdaBoost, Random forest is a meta-estimator algorithm that fits several decision tree classifiers on various sub-samples of the dataset. It operates by constructing many decision trees at training time. It then outputs the class that is the mode of the classes as mean prediction of the individual trees [20].

Support Vector Machine (SVM) - SVM has its roots in statistical learning theory and represents the decision boundary using a subset of the training examples known as support vector [22]. It is a discriminative classifier and is formally defined by separating hyperplanes. It has been successfully used for many classification and regression analysis problems. In SVM, new examples of data are mapped into a separating hyperplane and are predicted to belong to a category based on which side of the hyperplane the data falls on [20] [23]. Empirical studies show that SVM works well with high-dimensional data and avoids the dimensionality problem algorithm problem.

IV. Experimental methodology

During the study, all implementations were carried out using WEKA package (Waikato Environment for Knowledge Analysis) [20] software. WEKA is an open data mining suite implemented in JAVA for data classification, clustering, regression and visualization. We run the simulations on an Intel Core i5-3210M CPU 2.5GHz PC, equipped with 6.00 GB of RAM Windows 8.1 64-bit machine using various algorithmic settings of the boosting method and the base classifiers. The detail of the datasets used in the study is as presented in Fig. 2.

In the experimental setup, we used 100 independent iterations, 10-fold cross validation, 4 base learners, and 15 benchmark datasets. Each dataset is divided into 10 parts, 9 parts are used to train the model and 1 part is used to test the model. This method has the advantage of using all the datasets for both training and testing the model thereby try to avoid overfitting problem for each prediction model. Empirical study shows that stratified cross-validation tend to generate comparison results with lower bias and lower variance [24]. It involves splitting the original dataset into 10 folds of equal size. Nine parts are used for training and the tenth part using for testing. The learning dataset is used for acquiring rules and the validation dataset is used for validating the rules. The process is then carried out ten times.

a) For each fold train the classifier using all the folds except one ($K-1$).

b) Use the left-out fold to test the model by calculating the cross-validation metrics

c) Run the $k-fold$ cross validation run several times (in our case 10 times)

d) Average the cross-validation metrics across the subsets to get the final cross validation metrics.

The following learning algorithms were implemented as base classifiers during the simulation process: Decision Stumps, Neural Network, Random Forest and Support Vector Machine as described in the previous section. The base classifiers were selected based on their popularity [19], diverse applications [4], availability in experimental simulation [20], ability to handle training by reweighting and the classifier's cost matrix compatibility with the training data.

In the study, we used the original version of AdaBoost algorithm i.e. *AdaBoost.M1* as proposed by [2] as the boosting technique. There are two main reasons for using this version of the algorithm. Firstly, to have a generalized experimental set up that is independent of any AdaBoost variant. Secondly, to have an experimental conclusion that is based on the original version of the algorithm which serves as a primary algorithm for other variants.

Despite efforts made to minimize limitations that might affect the research there were a few constraints within which the research was carried out. Firstly, the research was based on secondary datasets, errors in datasets and missing data cannot be ruled out. Secondly, various parameters were configured to meet the needs of the experimental simulation. Changes to parameter settings can significantly affect the simulation results. Thirdly, the success of AdaBoost lies on the assumption that the errors of the individual models are uncorrelated because of diversity among the base classifiers. However, in practice the errors are highly correlated, therefore the reduction in overall error is generally small. Fourthly, all the successive models were weighted by their success during training.

In the study, five evaluation metrics were used namely: classification accuracy (CA), Kappa Statistics, RMSE, ROC and PRC. The evaluation of the RMSE and the accuracy are as represented in (1) and (2) respectively.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{1}$$

$$Accuraccy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

where, y_i is observed and $\hat{y}_i$ are modeled values respectively, True Positives (TP), True Negative (TN), False Positive (FP) and the False Negative (FN) are used in evaluating the performance of the models. To empirically compare the two methods of AdaBoost implementations described in this paper we conducted several experimental simulations as detailed in the previous sections.

Boosting by Reweighting

Criterion /Evaluation	C^+	KS	PRC-A	ROC-A	RMSE
Average Performance	73.375	0.306	0.719	0.721	0.326
Min Performance	15.000	-0.043	0.096	0.333	0.042
Max Performance	99.000	0.959	0.998	0.999	0.594

Boosting by Resampling

Criterion /Evaluation	C^+	KS	PRC-A	ROC-A	RMSE
Average / Performance	73.575	0.341	0.727	0.731	0.325
Min /Performance	15.000	-0.043	0.100	0.167	0.024
Max /Performance	99.000	0.959	0.998	0.998	0.767

Figure 3 Comparison results: Boosting by reweighting and boosting by resampling. Each column contains the evaluation metric used in the study.

V. Results and Discussions

The average metric performance of boosting by reweighting and boosting by resampling is as illustrated in Fig. 3. This is based on the on following criterions: Classification accuracy (C^+), Kappa Statistic (KS), Root Mean Error(σ_{RME}), ROC Area and PRC Area [9] [14].

A. Classification Accuracy (CA)

As shown in Figure 3 the classification accuracy of reweighting method is 73.375% and that of resampling method is 73.575%. This indicates a performance difference of 0.2%. The performance accuracy of boosting by reweighting and boosting by resampling of the models used in study are as illustrated in Fig. 4 and Fig. 5 respectively.

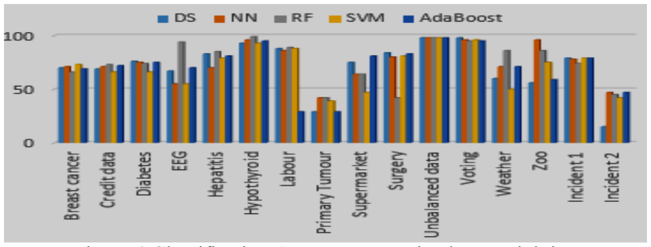


Figure 4 Classification Accuracy: Boosting by reweighting

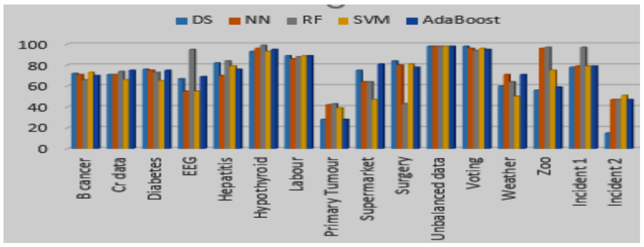


Figure 5 Classification Accuracy: Boosting by resampling

B. Kappa Statistics (KS)

In term of Kappa Statistics boosting by reweighting and boosting by resampling have an average value of 0.306 and 0.341 respectively. This shows a difference of 0.035 between the two methods. These are illustrated in Fig. 6 and Fig. 7 respectively.

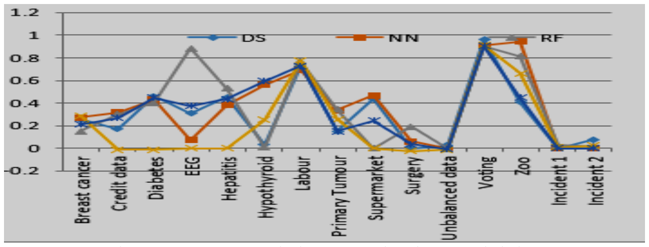


Figure 6 Kappa Statistics: Boosting by reweighting

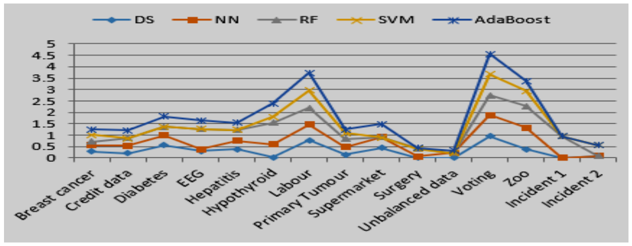


Figure 7 Figure 6 Kappa Statistics : Boosting by resampling

C. PRC-Area

When reweighting was used as a method of boosting the average PRC-Area is 0.719, on the other hand using resampling as method of boosting the value of PRC-Area is 0.727. This show a difference of 0.008 between the two methods. The PRC-Area of boosting by reweighting and boosting by resampling is illustrated graphically in Fig. 8 and Fig. 9 respectively.

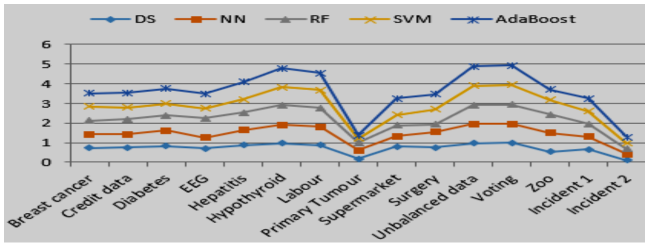


Figure 8 PRC-Area Boosting by reweighting

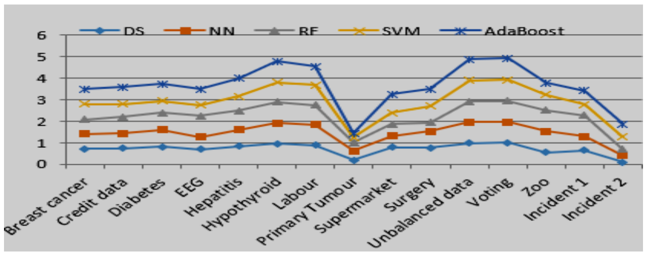


Figure 9 PRC-Area Boosting by resampling

D. RMSE

In term of RMSE metric, Table 3 shows that boosting by reweighting has value of 0.326 while boosting by resampling has an RMSE value of 0.325. Comparing the two methods it shows a difference of 0.001. These are shown graphically in Fig. 10 and Fig. 11 respectively.

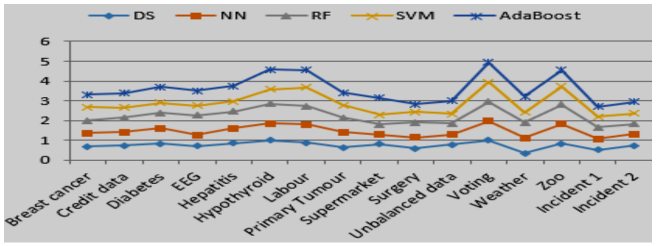


Figure 10 RMSE - Boosting by reweighting

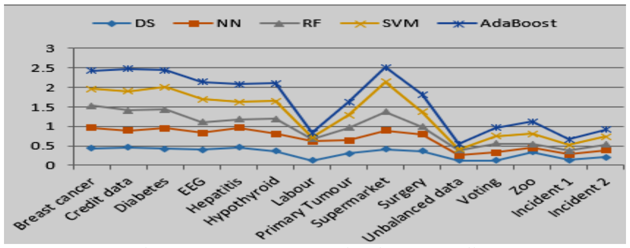


Figure 11 RMSE - Boosting by resampling

E. ROC-Area

The value of ROC-Area for boosting by reweighing and resampling methods are 0.721 and 0.731 respectively. This shows that performance difference between the two methods is 0.001. Graphically, these are illustrated in Fig. 12 and Fig. 13 respectively.

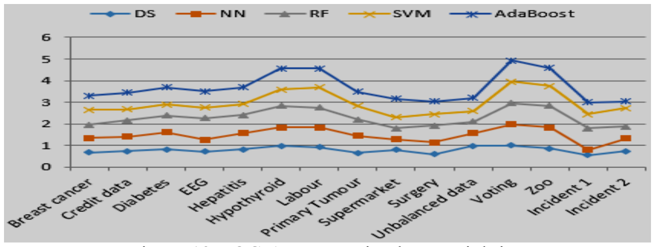


Figure 12 ROC-Area Boosting by reweighting

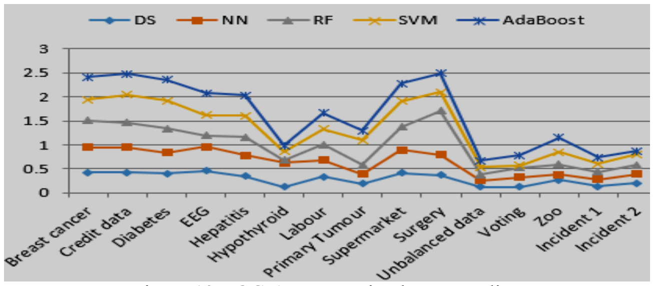


Figure 13 ROC-Area Boosting by resampling

VI. CONCLUSION REMARKS AND FUTURE WORK

In this paper, we tested and presented boosting implementation by reweighting and resampling methods. We applied Decision Stump, Neural Network, Random Forest and Support Vector Machine as base classifiers and AdaBoost as a boosting technique. We compare and evaluate performance of the two methods using healthcare and bioinformatics and other benchmark datasets obtained from the UCI repository.

The results of our study show that average performance of correctly classified by reweighting is 73% and that of boosting by resampling is 74%. Also, there are negligible differences in the KS, PRC-A, ROC-A and RMSE values. However, the results of the study show ANN ensemble models requires more time in training its base classifiers due to the complexity of the models. However, the study shows that the complexity of ANN ensemble models does not necessarily lead to better performance when compared to other ensemble models.

In the future, we intend to explore the influence of weak learner combination methods on performance and prediction accuracy of ensemble models. Investigation will also include the performance effect of algorithmic setting on boosting implementation methods.

REFERNCES

[1] E. Bauer and R. Kohavi, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," Machine Learning, vol. 36, no. 1, pp. 105-139, 1999.

[2] R. E. Schapire, "A Brief Introduction to Boosting," in IJCAI '99 Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999.

[3] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and application to boosting," Journal of computer and system sciences, vol. 55, pp. 119-139, 1997.

[4] Y. Freund, "Boosting a weak learning algorithm by majority," Information and Computation, vol. 121, no. 2, pp. 256-285, September 1995.

[5] R. Schapire and Y. Freund, Boosting: Foundations and algorithms, MIT Press, 2014.

[6] J. Lin and Y. Wang, "Using a Novel AdaBoost Algorithm and Chous Pseudo Amino Acid Composition for Predicting Protein Subcellular," Proten Pept Lett (US National Library of Medicine National Institutes of Health), vol. 18, no. 12, pp. 1219-25, December 2011.

[7] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," International Journal of Computer Vision, vol. 57, no. 2, pp. 137-154, 2004.

[8] K. A. Abuhasel and A. M. Iliyasu, "A combined AdaBoost and NEWFM Technique for Medical Data classification," Lecture Notes in Electrical Engineering, vol. 339, pp. 801-809, 18 February 2015.

[9] P. Karimi and H. Jazayeri-Rad, "Comparing the fault diagnosis performances of single Neural Networks and two Ensemble Neural Networks based on the boosting methods," Journal of Automation and Control, vol. 2, no. 1, pp. 21-32, 2014.

[10] R. Schapire, Computer Science 511 Theoretical Machine Learning, 2014.

[11] Y. Freund and R. E. Schapire, "Experiments with a New Boosting Algorithm," in Machine Learning: Proceedings of the Thirteenth International Conference, 1996.

[12] Y. Sun, J. Li and W. Hager, "Two new regularized AdaBoost algorithms," in International Conference on Machine Learning and Applications, 2004.

[13] M. K. Warmuth, K. Glocer and G. Ratsch, "Boosting Algorithms for Maximizing the Soft Margin," in Proceedings of the Twenty-first Annual conference on Neural Information Systems, 2007.

[14] W. S and N. H, "A new method for solving overfitting problem of gentle AdaBoost," in roc. SPIE 9069, Fifth International Conference on Graphic and Image Processing, Hong Kong, 2014.

[15] J. R. Quinlan, "Bagging, Boosting, and C4.5," in In Proceedings of the Thirteenth National Conference on Artificial Intelligence, 1996.

[16] E. Souza and S. Matwin, "Improvements to AdaBoost Dynamics," in Canadian AI'12 Proceedings of the 25th Canadian conference on Advances in Artificial Intelligence, 2012.

[17] M. Botta, "Resampling vs Reweighting in Boosting a Relational Weak Learner," in AI*IA 2001: Advances in Artificial Intelligence, 2001, pp. 70-80.

[18] C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse and A. Napolitano, "Resampling or Reweighting: A Comparison of Boosting Implementations," in 20th IEEE International Conference on Tools with Artificial Intelligence, 2008.

[19] UCI, "UCI Machine Learning Repository," [Online]. Available: http://archive.ics.uci.edu/ml/. [Accessed 21 December 2015].

[20] I. Witten, E. Frank and M. Hall, Data Mining Practical Machine Learning Tools and Techniques, 3rd ed., San Francisco, California: Morgan Kaufmann Publishers, 2011.

[21] S. Haykin, Neural Networks and Learning Machines: A Comprehensive Foundation, Third ed., Prentice Hall, 2008.

[22] T. Pang-Ning, S. Michael and K. Vikin, Data Mining, Pearson Addision Wesley, 2005.

[23] C. Corrinna and V. Vladimir, "Support-Vector Networks," Machine Learning, vol. 20, pp. 273-279, 1995.

[24] D. Delen, G. Walker and A. Kadam, "Predicting breast cancer survivability: A comparison of three data mining methods," Artif Intell Med., vol. 34, no. 2, pp. 113-27, 2005.