Voting based Extreme Learning Machine with Accuracy based ensemble Pruning

Sanyam Shukla
CSE department, MANIT, Bhopal, India

R. N. Yadav
ECE Department, MANIT,
Bhopal, India

ABSTRACT

Extreme Learning Machine is a fast single layer feed forward neural network for real valued classification. It suffers from the problem of instability and over fitting. Voting based Extreme Learning Machine, VELM reduces this performance variation in Extreme Learning Machine by employing majority voting based ensembling technique. VELM improves the performance of ELM at the cost of increased redundancy. This problem can be reduced using ensemble pruning techniques. This work proposes and evaluates Voting based Extreme Learning Machine with Accuracy based ensemble Pruning, VELM_AP. VELM_AP generates component classifier in the same way as VELM.

General Terms

Real valued classification

Keywords

Ensemble Pruning; Extreme learning Machine;

1. INTRODUCTION

Classification is a supervised learning technique used for developing a model, which can be used to predict the class label of test instance. Intrusion detection, spam filtering, biometric recognition etc. are some examples of classification problems. Classification problems in which all the attributes are real valued are called real valued classification problems. So many classifiers like SVM, C4.5, Naive Bayes etc. are available for real valued classification. Extreme learning machine, ELM [1] is a state of art classifier for real valued classification problems. ELM is a single layer feed forward neural network in which the weights between input and hidden layer are initialized randomly. The weights between hidden and output layer are computed analytically, which makes extreme learning machine fast compared to other gradient based classifiers. Due to random initialization of input layer weights the performance of extreme learning machine fluctuates. The performance fluctuation due to any change in the parameters of the classification algorithm or training dataset composition is known as error due to variance. Ensembling approaches like bagging[2], adboost.M1 [3], adaboost.M2[3] can be used to reduce this variance in performance and also, increase the performance of the classifier. As ELM suffers from problem of stability and over fitting, many variants of ELM [4]–[10] based on ensembling techniques have been proposed. VELM [4], uses majority voting to get combined outcome of independent component classifiers of the ensemble. It improves the performance of ELM at the cost of increases redundancy. Ensemble pruning techniques [11]–[18] can be employed to a get a sub-ensemble containing accurate and diverse classifiers. VELM_AP uses accuracy measure for pruning VELM. In the next section this paper discusses related work i.e. ELM, VELM and ensemble pruning techniques. After this section, this paper describes the proposed work. After that, this paper describes the experimental setup and results obtained. The last section consists of conclusion and future work.

2. RELATED WORK

This section contains the brief review of the fundamental topics which were proposed earlier and are important from the perspective of the proposed work.

2.1 Extreme Learning Machine

Let the input to ELM be N training samples with their targets [(x_1, t_1), (x_2, t_2),…, (x_j, t_j),…, (x_N, t_N)] Here j=1, 2, **…,** N, x_j = [x_{j1}, x_{j2} , …, x_{jF}]^T Є R^m and t_i ϵ 1, 2, ..., C. Here ,the number of features and classes are represented by F and C respectively. Fig. 1 shows the architecture of ELM.

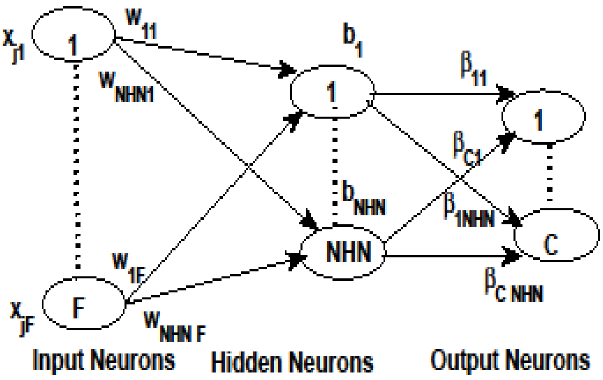

Fig 1: **Architecture of ELM**

In ELM, the weights between input and hidden neurons are assigned randomly. The weights between output and hidden neurons are computed analytically. This reduces the overhead of tuning the learning parameters, which makes it fast and more accurate compared to other gradient based techniques. In ELM, the neurons in the hidden layer use non-linear activation function whereas the neurons in the output layer use linear activation function. Vector, w_i = [w_{1i}, w_{2i},…, w_{Fi}]^T represents the weight vector connecting Fth input neurons to the ith hidden neurons , where i=1, 2,**….** , NHN, b_i the bias of ith hidden neuron. The output of the ith hidden neuron is represented by g($w_{i.}$ x + b_i). The hidden layer output of all training samples is represented as follows.

$$H_{train}(N\,X\,NHN)=\begin{bmatrix} g(w_1.x_1+b_1) & \Lambda & g(w_{NHN}.x_1+b_{NHN}) \\ \mathrm{M} & \mathrm{K} & \mathrm{M} \\ g(w_1.x_N+b_1) & \Lambda & g(w_{NHN}.x_N+b_{NHN}) \end{bmatrix}$$

Here, g is the activation function. Any infinitely differentiable like Sigmoid function, Radial Basis function etc. can be used as the activation function of hidden layer neurons. Vector, β_k = [β_{1k}, β_{2k}, ..., $\beta_{NHN\ k}$]^T represents the output weight vector which connects the kth output neurons to the hidden neurons, here k = 1, 2, ..., C. The target of x_j i.e. t_j is represented as target vector T_j **=** [T_{j1}, T_{j2}, …, T_{jC}]^{T,} where t_{jk} = +1 if t_j = k else t_{jk} = -1. Vector, β_k is determined analytically using the equation given below.

$$\beta = \left(H_{train}\right)^{+} T_{train}$$

$$Here, \quad \beta\left(NHN\,X\,C\right) = \left[\beta_1^{\;T}, \beta_2^{\;T}, ..., \beta_C^{\;T}\right]$$

$$T_{train}\left(NXC\right) = \begin{bmatrix} t_1^T \\ \mathrm{M} \\ t_N^T \end{bmatrix}$$

H_{test} (n x NHN), is the hidden layer output matrix for the n testing instances. The predicted output for testing instances i.e. Y_{test} (n x C) is determined using the following equation

$$Y_{test} = H_{test}\,\beta$$

L_{test}(n x 1), output label of n testing instances is determined using this equation

$$L_{test} = \arg^{row}_{\max}\left(Y_{test}\right)$$

The arg function returns the index of the maximum value for each row of Y_{test}.

2.2 Voting Based Extreme Learning Machine

ELM suffers from the problem of instability and over fitting. The instability problem arises due to random initialization of weights between the input and hidden layer. V-ELM [4] solves this problem, by generating a number of classifiers, succeeded by majority voting for finding the prediction of the ensemble. Independent component classifiers of the ensemble are generated by randomly assigned different weights between input and hidden layer. For n testing instances, the output label (L_{test}) corresponding to all the component classifiers is obtained. The final predicted output (FP) is given by:

$$FP = \mathrm{mode}\left(L_{test}\right)$$

Here, $L_{test} = [L^1_{test}, L^2_{test}, \ldots, L^{NCE}_{test}]$

The mode operation calculates the class to which the maximum numbers of classifiers are voting. Taking an example of binary classification where, an instance belongs to either positive class or negative class. Number of classifier, NCE =100. Let for any test instance 70 classifiers vote for positive class whereas, 30 vote for negative output. Then the final output of V-ELM is positive class.

2.3 Ensemble Pruning

It is stated in [19] that many is better than all. Instead of using all component classifiers, a subset of accurate and diverse classifiers may give equal or better performance. A number of ensemble pruning methods have been proposed. Ensemble pruning techniques are mainly classified in three categories [20] : Order based, Clustering based and Optimization based. Ordering based ensemble pruning technique orders the component classifiers of the ensemble as per their importance which is quantified by suitable metric. The final sub ensemble is constructed by choosing first few classifiers as per their ordering. The number of classifiers in the pruned ensemble is determined by setting threshold. Some of the ordering based pruning techniques are Reduce Error Pruning, Kappa pruning, Complementariness Pruning[18], Margin distance pruning[18] etc. The optimization based pruning techniques give better solution compared to other techniques but they are computationally intensive[15]. The author in [21] uses Accuracy and reduce error pruning technique to get an optimally pruned ensemble. Backtracking in reduce error pruning increases computational overhead and guarantees that the pruned ensemble will have greater or equal accuracy than the full ensemble.

3. PROPOSED WORK

In order to reduce the redundant classifiers in VELM this work proposes and evaluates ordering based ensemble pruning algorithm. This work uses G-mean metric to quantify the importance of component classifiers of the ensemble. Compared to overall accuracy, G-mean is a better accuracy metric when data is not balanced. The proposed work assumes that all the ELM based classifiers are diverse as the weights between the input and hidden neurons are assigned randomly. The pseudo code of proposed algorithm is as follows:

Algorithm for VELM_AP

Training Phase

I. Generate NCE ELM based classifiers.

II. Find the output of all component classifiers of the pruned ensemble for training dataset. Compute accuracy on training data.

III. Arrange the classifiers in decreasing order of training G-mean.

IV. To avoid tie condition during voting select odd number of top classifiers from the ordered list of classifiers to make the pruned ensemble

Testing Phase

I. Find the output of component classifiers of the pruned ensemble for test dataset.

II. Perform majority voting of outcomes of the classifiers in the pruned ensemble to get the final outcome,

III. Do performance evaluation using the final outcome.

4. EXPERIMENTAL SETUP & RESULT ANALYSIS

4.1 Data Specification

The proposed work is evaluated using 12 binary and 3 multiclass datasets, downloaded from the Keel-data set Repository [22]. The data sets in Keel Repository are available in 5 fold cross validation format i.e. for each dataset has 5 training and 5 testing sets. The specification of datasets for each testing and training dataset is shown in the Table I.

Table 1. Specifications of datasets used for evaluation.

DATA SET	Number of classes	Number of Attributes	Number of Training instances	Number of Testing instances
APPENDICITIS	2	7	84	22
BANANA	2	2	4240	1060

BUPA	2	6	276	69
CHESS	2	36	2556	640
HABERMAN	2	3	244	62
HAYES-ROTH	3	4	128	32
HEART	2	13	216	54
IONOSPHERE	2	33	280	71
NEWTHYROID	3	5	172	43
PHONEME	2	5	4323	1081
PIMA	2	8	614	154
SA_HEART	2	9	369	93
SONAR	2	60	166	42
SPECTFHEART	2	44	213	54
VEHICLE	4	18	676	170

4.2 Performance Metrics

The results of binary classification can be categorized as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). True positive and True Negative are the correctly classified instances belonging to positive class and negative class respectively. False Negative is the number of instances belonging to positive class that are misclassified as negative class. FP is the number of instances belonging to negative class and classified as positive class. The overall accuracy is calculated by the following formula.

$$Overall\ Accuracy = \frac{\#\,TP + \#TN}{\#\,Samples}$$

$$Gmean = \sqrt{\frac{\#TP}{\#TP + \#\,FN} * \frac{\#TN}{\#TN +\ \#\,FP}}$$

Here, # represents number of.

4.3 Parameter Setting

V-ELM is treated as the special case of V-ELM_AP, when all the classifiers participate in majority voting in order to have a fair comparison between V-ELM and VELM_AP. Results presented in this section are averaged over 10 trials. In each trail 50 ELM classifiers are generated and the final outcome of VELM is the majority voting of all these 50 classifiers. Optimal number of NHN has been found by varying NHN from [10, 20… 100]. To obtain the optimal results for V-ELM_AP a grid search is done on NHN from [10 20 …100] and on P_NCE from [1, 2…50]. P_NCE is the number of classifiers in the pruned ensemble. Pruned Ensemble, PE is made by selecting top P_NCE classifiers from the ordered list of classifiers. Overall accuracy of the PE is calculated by conducting majority voting of selected P_NCE classifiers. Use of any pruning technique will give different overall accuracy corresponding to the choice of P_NCE and NHN. The impact of varying these parameters for Appendicitis dataset is shown in Fig. 2. In Fig. 2(a) surface plot obtained by varying these parameters is given. It can be seen from the figure that performance obtained is low for higher values of NHN possibly because of over fitting. Along with the dependency on NHN, the output of VELM_AP is also dependent on number of classifiers in the pruned ensemble i.e. P_NCE. Fig. 2(b) displays variation in test performance when P_NCE is varied.

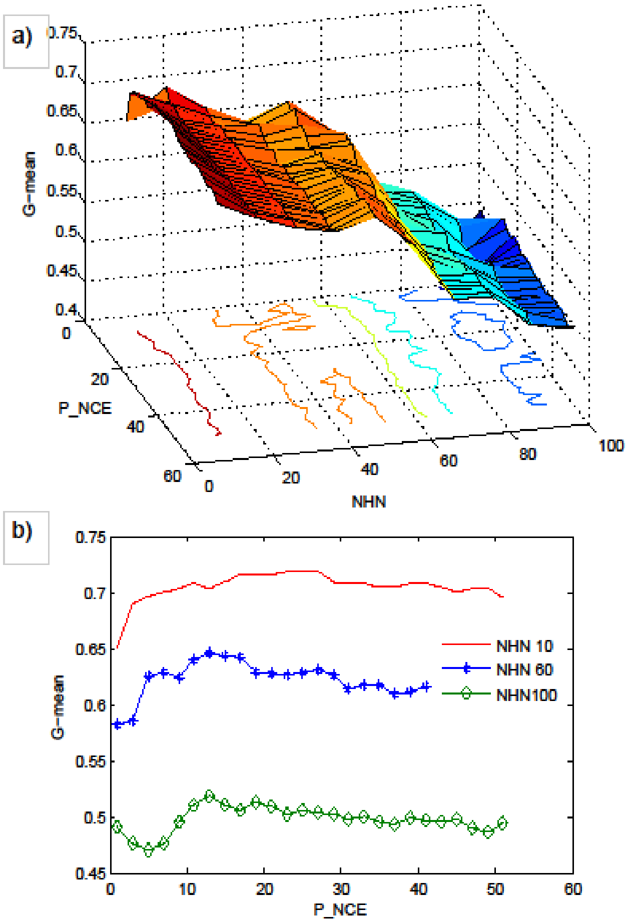

Fig 2. Display of variation in testing performance of appendicitis dataset a) by varying NHN and P_NCE. b) Varying P_NCE with constant NHN.

Fig. 3 shows the variation in performance of various datasets when P_NCE is varied with NHN kept constant. It can also be observed that pruning decreases the computational requirement with slight increase in performance.

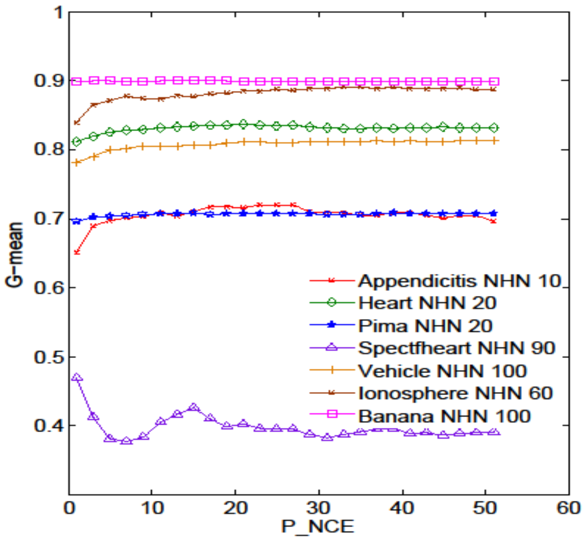

Fig. 3. Display of variation in test G-mean by varying P_NCE with constant NHN

Table 2. G-Mean of Test Dataset

DATA SET	V-ELM		VELM_AP		
	NHN	*G-mean %*	*P_NCE*	*NHN*	*G-mean %*
APPENDICITIS	10	69.57	25	10	**71.99**
BANANA	100	89.89	13	100	**89.96**
BUPA	20	69.74	41	20	**69.76**
CHESS	100	**95.37**	51	100	**95.37**
HABERMAN	20	48.57	39	20	**48.86**
HAYES-ROTH	40	**72.07**	51	40	**72.07**
HEART	20	83.24	20	20	**83.65**
IONOSPHERE	80	89.08	39	60	**89.17**
NEWTHYROID	30	**87.10**	51	30	**87.10**
PHONEME	100	**80.38**	51	100	**80.38**
PIMA	30	70.71	15	20	**70.80**
SA_HEART	20	64.40	7	20	**64.91**
SONAR	80	84.30	49	80	**84.65**
SPECTFHEART	90	39.11	10	90	**46.89**
VEHICLE	100	81.31	41	100	**81.32**

4.4 Result and Analysis

Testing G-mean and Overall accuracy of proposed classifier for various datasets is shown in Table 2 and Table 3 respectively. It can be observed from Table 2 that VELM_AP is better than VELM. VELM_AP outperforms VELM for all evaluated datasets. For further comparison of proposed classifier with V-ELM wilcoxon is conducted. The threshold value of alpha is taken as .05. The p value obtained by the test is equal to 9.7656e-04. The smaller the p value the improvement is more significant.

. **Table 3: Testing Average Overall Accuracy (AOA) .**

DATA SET	V-ELM		V-ELM_AP		
	NHN	*AOA%*	*P_NCE*	*NHN*	*AOA%*
APPENDICITIS	10	87.80	17	10	**89.04**
BANANA	90	90.27	13	90	**90.32**
BUPA	20	72.93	41	20	**72.95**
CHESS	100	**95.42**	51	100	**95.42**
HABERMAN	20	72.98	3	10	**74.42**
HAYES-ROTH	40	**75**	51	40	**75**
HEART	20	84.07	19	20	**84.52**
IONOSPHERE	80	92.12	39	60	**92.26**
NEWTHYROID	30	**94.79**	51	30	**94.79**
PHONEME	100	84.26	49	100	**84.27**
PIMA	20	77.64	19	20	**77.76**
SA_HEART	20	73.41	7	20	**73.63**
SONAR	100	84.93	47	80	**85.21**
SPECTFHEART	10	79.41	3	20	**79.47**
VEHICLE	100	82.89	41	100	**82.92**

5. CONCLUSION AND FUTURE WORK

This paper proposes a new classifier, VELM_AP which is an extension of VELM. VELM gives better performance than ELM with increased computational and memory requirement. VELM_AP first creates NCE classifiers using ELM. VELM_AP then applies accuracy based ensemble pruning to reduce the redundant classifiers. VELM_AP uses G-mean measure to quantify the importance of classifier to get an optimal subset of pruned ensemble. All the classifiers are arranged as per decreasing order of their training G-mean. Then a few top classifiers from the ordered list are chosen to make the pruned ensemble. The proposed classifier is evaluated using various datasets available at Keel repository. VELM_AP outperforms V-ELM for all evaluated datasets taken from KEEL repository. This is further illustrated from the result of wilcoxon signed rank test. The future work includes finding an approach to determine the optimal number of classifiers to be selected in the pruned ensemble. The future work also includes exploring other ensemble pruning techniques to enhance the performance of voting based extreme learning machine.

6. REFERENCES

[1] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70. pp. 489–501, 2006.

[2] L. Breiman, "Bagging predictors: Technical Report No. 421," 1994.

[3] Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," *Mach. Learn. Work. ...*, 1996.

[4] J. Cao, Z. Lin, G. Bin Huang, and N. Liu, "Voting based extreme learning machine," *Inf. Sci. (Ny).*, vol. 185, pp. 66–77, 2012.

[5] J. Cao, S. Kwong, R. Wang, X. Li, K. Li, and X. Kong, "Class-specific soft voting based multiple extreme learning machines ensemble," *Neurocomputing*, vol. 149, Part , no. 0, pp. 275–284, Feb. 2015.

[6] N. Liu and H. Wang, "Ensemble based extreme learning machine," *IEEE Signal Process. Lett.*, vol. 17, pp. 754–757, 2010.

[7] J. Zhai, H. Xu, and X. Wang, "Dynamic ensemble extreme learning machine based on sample entropy," *Soft Computing*, vol. 16. pp. 1493–1502, 2012.

[8] Y. Lan, Y. C. Soh, and G. Bin Huang, "Ensemble of online sequential extreme learning machine," *Neurocomputing*, vol. 72. pp. 3391–3395, 2009.

[9] G. Wang and P. Li, "Dynamic Adaboost ensemble extreme learning machine," in *ICACTE 2010 - 2010 3rd International Conference on Advanced Computer Theory and Engineering, Proceedings*, 2010, vol. 3.

[10] L. Yu, X. Xiujuan, and W. Chunyu, "Simple ensemble of extreme learning machine," in *Proceedings of the 2009 2nd International Congress on Image and Signal Processing, CISP'09*, 2009.

[11] Z. Lu, X. Wu, X. Zhu, and J. Bongard, “Ensemble Pruning via Individual Contribution Ordering,” in *The 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining Washington DC*, 2010, no. 1, pp. 871–880.

[12] T. Windeatt and C. Zor, “Ensemble Pruning Using Spectral Coefficients,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 24, no. 4, pp. 673–678.

[13] L. Guo and S. Boukir, “Margin-based ordered aggregation for ensemble pruning,” *Pattern Recognit. Lett.*, vol. 34, no. 6, pp. 603–609, 2013.

[14] I. Partalas, G. Tsoumakas, and I. Vlahavas, “An ensemble uncertainty aware measure for directed hill climbing ensemble pruning,” *Mach. Learn.*, vol. 81, no. 3, pp. 257–282, 2010.

[15] G. Martinez-Muñoz, D. Hernández-Lobato, and A. Suarez, “An analysis of ensemble pruning techniques based on ordered aggregation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 245–259, 2009.

[16] Y. Zhang, S. Burer, and W. N. Street, “Ensemble pruning via semi-definite programming,” *J. Mach. Learn. Res.*, vol. 7, pp. 1315–1338, 2006.

[17] L. Rokach, “Collective-agreement-based pruning of ensembles,” *Comput. Stat. Data Anal.*, vol. 53, no. 4, pp. 1015–1026, 2009.

[18] G. Martínez-Muñoz and A. Suárez, “Aggregation Ordering in Bagging,” in *Proceedings of the {IASTED} International Conference on Artificial Intelligence and Applications*, 2004, pp. 258–263.

[19] Z. H. Zhou, J. Wu, and W. Tang, “Ensembling neural networks: Many could be better than all,” *Artif. Intell.*, vol. 137, no. 1–2, pp. 239–263, 2002.

[20] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. 2012.

[21] M. Bhardwaj and V. Bhatnagar, “Towards an optimally pruned classifier ensemble,” *Int. J. Mach. Learn. Cybern.*, pp. 1–20, 2014.

[22] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, “KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework,” *J. Mult. Log. Soft Comput.*, vol. 17, pp. 255–287, 2011.