Bagging.LMS: A Bagging-based Linear Fusion with Least-Mean-Square Error Update for Regression

Yunfeng Wu[1], Cong Wang[1], and S.C. Ng[2]
[1]School of Information Engineering, Beijing University of Posts and Telecommunications,
Xi Tu Cheng Road 10, Haidian District, Beijing 100876, China
[2]School of Science and Technology, The Open University of Hong Kong,
30 Good Shepherd Street, Homantin, Kowloon, Hong Kong

***Abstract*—The merits of linear decision fusion in multiple learner systems have been widely accepted, and their practical applications are rich in literature. In this paper we present a new linear decision fusion strategy named Bagging.LMS, which takes advantage of the least-mean-square (LMS) algorithm to update the fusion parameters in the Bagging ensemble systems. In the regression experiments on four synthetic and two benchmark data sets, we compared this method with the Bagging-based Simple Average and Adaptive Mixture of Experts ensemble methods. The empirical results show that the Bagging.LMS method may significantly reduce the regression errors versus the other two types of Bagging ensembles, which indicates the superiority of the suggested Bagging.LMS method.**

I. INTRODUCTION

An ensemble is a multi-learner system in which a series of component learners are generated to solve the same task [1]. The component learners are combined with a certain fusion strategy to form the final prediction [2], [3]. In other words, the ensemble fuses the knowledge acquired by local learners to make a consensus decision which is supposed to be superior to the one attained by any individual learner working solely. The most popular ensemble algorithms are Boosting [4] and Bagging [5], which have already been widely used to improve accuracy for solving classification and regression problems [6].

Boosting works by repeatedly implementing a given weak learning machine on different distributed training data sets, and then fusing their outputs. The distribution of the training data in the current iteration depends on the performance of the prior learner in the previous iteration. The first version of the Boosting algorithm described by Schapire is Boosting by filtering [7], which involves filtering the training data by different versions of a weak learning algorithm. However, it requires a large size of training data, which is not feasible in many practical applications [8]. In order to overcome such a drawback, Freund and Schapire proposed AdaBoost [4] to find a typical mapping function or hypothesis with a low error rate relative to a given probability distribution of the training data. For regression problems, Freund and Schapire developed AdaBoost.R [9] which is effective by projecting the regression data into some classification sets. In spite of their effectiveness, the Boosting algorithms still contain some pitfalls. First, they have to expand each regression data set into many classification sets and the number of projected classification examples grows intensively large after a few boosted iterations. Second, the loss function changes from iteration to iteration and even differs between examples during the same iteration. In addition, the Boosting algorithms make use of the training errors to adjust the fusion weights despite the fact that the training errors are often highly biased [10].

On the other hand, Bagging introduces the bootstrap sampling technique [11] into the procedure of constructing component learners, and expects to generate enough independent variance among them [5]. The bias of the Bagging ensemble would converge by averaging while the variance gets much smaller than that of each component learner.

Recently, linear decision fusion strategies [3], [12], [13], [14], or linear combination rules are frequently used in AdaBoost, Bagging, and other ensemble learning methods, such as Random Forests [15]. In general, they can be categorized into two styles: Fixed and Trained rules [16]. The fixed rules, such as Simple Average (SA) [17], contain the fixed fusion parameters. On the contrary, the trained rules, e.g. Weighted Average (WA) [18] and Adaptive Mixture of Experts (AME) [19], require a learning process to initialize and update the fusion coefficients. In this paper, we introduce a new linear fusion strategy taking advantage of the Least-Mean-Square (LMS) algorithm for Bagging, called Bagging.LMS. And the motivation is to incorporate the merits of Bagging and the linear decision fusion strategy in order to improve accuracy in regression problems.

The rest of this paper is organized as follows. Section 2 presents the details of the LMS error update algorithm for the Bagging ensembles. Section 3 describes the numerical experiments of regression, and compares the empirical results among the Bagging-based SA, AME, and LMS strategies. In the end, the advantages of the Bagging.LMS and the future work are summarized in Section 4.

II. LINEAR FUSION WITH THE LMS ERROR UPDATE FOR BAGGING

The goal of regression is to get a statistical model of the functional relationship between one dependent variable y and the multivariate input $\mathbf{x} = [x_1, \ldots, x_N]^{\mathrm{T}}$. The regression model can be expressed by a function plus a residual ε, which is similar to a noise, i.e.

$$y = r(\mathbf{x}) + \varepsilon . \tag{1}$$

1-4244-0549-1/06/$20.00 ©2006 IEEE.

In other words, the task of regression can be regarded to find out the linear or nonlinear mapping throughout a set of input-response pairs $S = \{(\mathbf{x}, \mathbf{y})\}_{p=1}^{P}$ ($\mathbf{x} \in \Re^{N \times P}$, $\mathbf{y} \in \Re^{1 \times P}$).

The Bagging method contains a bootstrap sampling procedure which generates a series of "bagged" sets by resampling with replacement from the original data set. The "bagged" sets S^k, $k = 1, \cdots, K$, also of size P, are used for training the K component learners in the ensemble. These component learners are linearly fused to form the final prediction $f(\mathbf{x})$, i.e.

$$f(\mathbf{x}) = \boldsymbol{\alpha}^{\mathrm{T}} \mathbf{o}(\mathbf{x}), \tag{2}$$

where vector $\boldsymbol{\alpha} = [\alpha_0, \alpha_1, \cdots, \alpha_K]^{\mathrm{T}}$ represents the fusion coefficients correspondingly allocated to the matrix $\mathbf{o}(\mathbf{x}) = [\mathrm{bias}, \mathbf{o}_1(\mathbf{x}), \cdots, \mathbf{o}_K(\mathbf{x})]^{\mathrm{T}}$, ($\mathbf{o}_k(\mathbf{x}) \in \Re^{1 \times P}$,) which includes the fusion bias parameter and the outputs of component learners. Particularly, α_0 is fixed at +1 and allocated to the bias. According to the LMS error criterion, the instantaneous cost function in the i-th update iteration is

$$C(\mathbf{x}^i) = e(\mathbf{x}^i)/2. \tag{3}$$

where $e(\mathbf{x}^i)$ is the instantaneous estimate error, i.e.

$$e(\mathbf{x}^i) = y^i - f(\mathbf{x}^i) = y^i - \hat{\boldsymbol{\alpha}}^{i\mathrm{T}} \mathbf{o}(\mathbf{x}^i) \tag{4}$$

Differentiating (3) with respect to the fusion coefficients yields the gradient approximation of $C(\mathbf{x}^i)$, i.e.

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} C(\mathbf{x}^i) &\approx \frac{1}{2} \frac{\partial e(\mathbf{x}^i)^2}{\partial \boldsymbol{\alpha}}\bigg|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}^i} \\ &= \frac{1}{2} \frac{\partial}{\partial \boldsymbol{\alpha}} \left[(y^i)^2 - 2y^i \hat{\boldsymbol{\alpha}}^{i\mathrm{T}} \mathbf{o}(\mathbf{x}^i) + \hat{\boldsymbol{\alpha}}^{i\mathrm{T}} \left(\mathbf{o}(\mathbf{x}^i) \mathbf{o}(\mathbf{x}^i)^{\mathrm{T}} \right) \hat{\boldsymbol{\alpha}}^i \right]. \\ &= -y^i \mathbf{o}(\mathbf{x}^i) + \left(\hat{\boldsymbol{\alpha}}^{i\mathrm{T}} \mathbf{o}(\mathbf{x}^i) \right) \mathbf{o}(\mathbf{x}^i) \\ &= -\left(y^i - \hat{\boldsymbol{\alpha}}^{i\mathrm{T}} \mathbf{o}(\mathbf{x}^i) \right) \mathbf{o}(\mathbf{x}^i) = -e(\mathbf{x}^i) \mathbf{o}(\mathbf{x}^i) \end{aligned} \tag{5}$$

Hence the fusion coefficients update rule can be written by following the steepest descent gradient algorithm as

$$\boldsymbol{\alpha}^{i+1} = \boldsymbol{\alpha}^i + \mu[-\nabla_{\boldsymbol{\alpha}} C(\mathbf{x}^i)] = \boldsymbol{\alpha}^i + \mu e(\mathbf{x}^i) \mathbf{o}(\mathbf{x}^i), \tag{6}$$

where the positive parameter μ represents the magnitude of the update rate for the fusion coefficients in the negative gradient direction.

III. Experiments and Results

For the regression performance evaluation of the Bagging.LMS, four synthetic and two benchmark data sets are selected in our experiments. The details of the synthetic data sets are tabulated in Table 1. The 2-D Mexican Hat has been used by Stitson *et al.* [20] for investigating the classification performance of support vector machines. Two benchmark data sets are the Friedman #1 and Friedman #3, which have ever been used by Briedman [5] for testing the regression performance of Bagging.

For comparison purpose, two frequently used fusion strategies: Simple Average (SA) and Adaptive Mixture of Experts (AME) are also applied in the experiments for the Bagging ensembles (with the abbreviation of Bagging.SA and Bagging.AME, respectively). The component learners are three Multilayer Perceptrons (MLPs) independently trained by the scaled conjugate gradient [8], Levenberg-Marquardt [21], and resilient backpropagation [22] algorithms, respectively. Each MLP network contains one hidden layer which consists of five hidden nodes. We didn't optimize the MLP architecture because in this investigation we only focus on the relative regression performance between the Bagging ensembles instead of the absolute performance of a MLP. We repeated total 10 independent trails for each fusion strategy in order to obtain the statistical significant results.

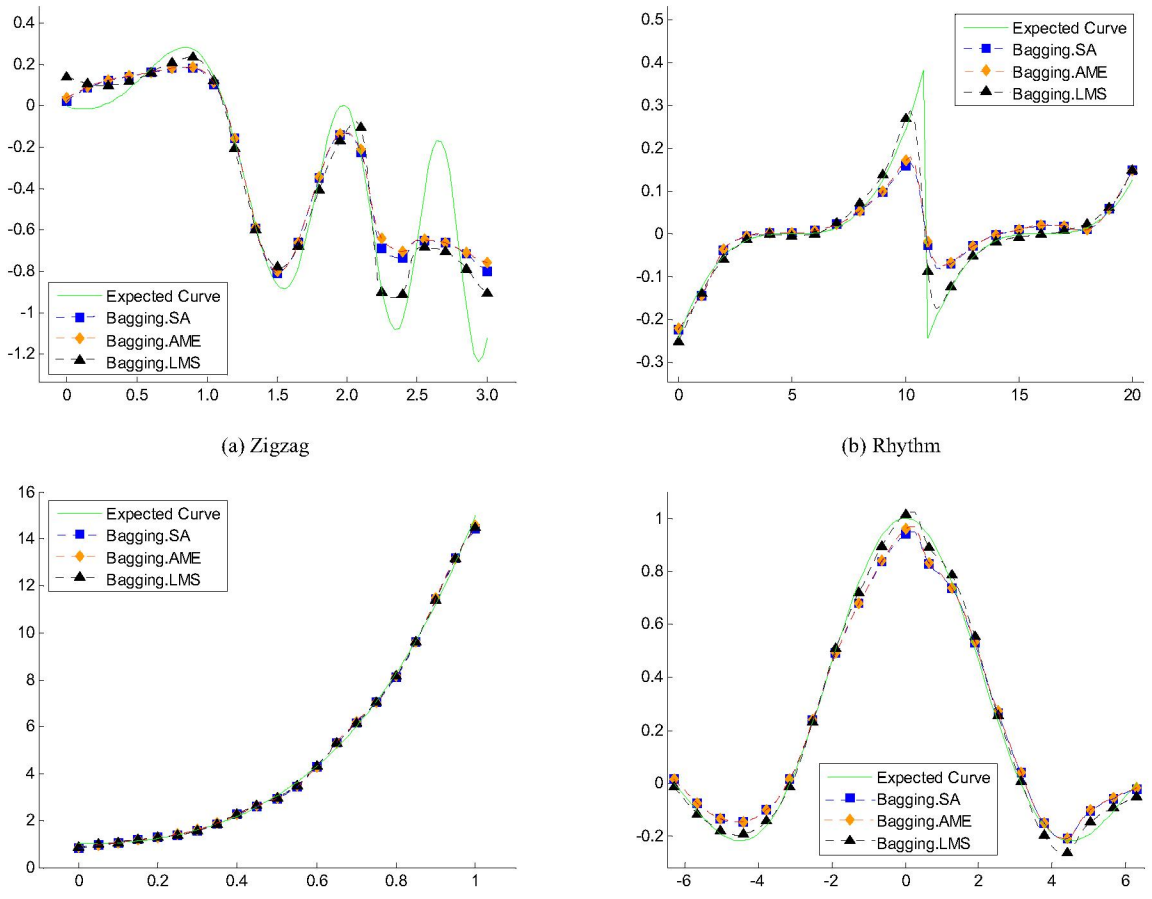
Fig. 1 plots the two-dimensional predictions (dotted lines) of the three types of Bagging ensembles compared with the four synthetic regression responses (solid line). It can be observed that the Bagging.LMS has a better performance of curve fitting than either the Bagging.SA or Bagging.AME. Table 2 gives the quantitative results in terms of mean squared error (MSE) averaged on the 10 independent trails. It can also be noted that the Bagging.SA outperforms the Bagging.AME for some data sets, in particular, Zigzag, Friedman #1, and Friedman #3, but the distinctions for

TABLE I Descriptions of the Synthetic Regression Data Sets

Data Sets	Function Expression	Variables	Size
Zigzag	$y = \sin x^2 \cos x^2 - 0.25x + \varepsilon$	$x \in \mathrm{U}[0,3]$ $\varepsilon \sim N(0, 0.05)$	100
Rhythm	$y = \left[\frac{(\mathrm{mod}(x,1) - 5)}{8} \right]^3 + \varepsilon$	$x \in \mathrm{U}[0,20]$ $\varepsilon \sim N(0, 0.01)$	100
Polynomial	$y = 1 + 2x + 3x^2 + 4x^3 + 5x^4 + \varepsilon$	$x \in \mathrm{U}[0,1]$ $\varepsilon \sim N(0, 0.1)$	100
2-D Mexican Hat	$y = \frac{\sin \lvert x \rvert}{\lvert x \rvert} + \varepsilon$	$x \in \mathrm{U}[-2\pi, 2\pi]$ $\varepsilon \sim N(0, 0.1)$	100

TABLE II The Averaged MSE of Bagging Ensembles with Three Different Types of Fusion Strategies

Data Sets	Mean Squared Error (MSE) of different Fusion Strategies for Bagging		
	SA	AME	LMS
Zigzag	0.0354	0.0372	0.0275
Rhythm	0.0035	0.0036	0.0022
Polynomial	0.0164	0.0142	0.0128
2-D Mexican Hat	0.0027	0.0026	0.0008
Friedman#1	0.2520	0.2545	0.2394
Friedman#3	0.0132	0.0411	0.0065



(a) Zigzag (b) Rhythm

(c) Polynomial (d) 2-D Mexican Hat

Fig. 1. Predictions of the Bagging ensembles with different decision fusion strategies for four synthetic regression data sets.

Rhythm, 2-D Mexican Hat are not much. On the other side, the Bagging.LMS achieves significantly higher prediction accuracy, especially for Zigzag, 2-D Mexican Hat, and Friedman #3, versus the Bagging.SA (0.0079, 0.0019, and 0.0067 higher, respectively) or Bagging.AME (0.0097, 0.0018, and 0.0346 higher, respectively). Fig. 2 gives box-and-whisker regression plots of three Bagging ensembles in terms of averaged MSE values with the interquartile range. It is clear that the Bagging.LMS consistently outperforms the other two fusion strategies on the total six regression data sets.

IV. Conclusion

The empirical results show that the Bagging ensemble with the LMS decision fusion strategy does consistently improve the prediction accuracy versus the previous fusion strategies for regression, which indicates that this ensemble method may also be promising for the design of multiple classifier systems. The further analysis of classification performance of the Bagging.LMS and more experimental comparisons with other trained fusion rules would be the next step of our work.

Acknowledgment

This work was supported in part by National Science Foundation of China under the Grant No. 60575034, and by the 2005 Innovation Research Funds from Graduate School, Beijing University of Posts and Telecommunications.

References

[1] L.K. Hansen, P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[2] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, Mar 1998.

[3] Y.F. Wu, J. He, Y. Man, and J.I. Arribas, "Neural network fusion strategies for identifying breast masses," in *Proc. 2004 Int'l Joint Conf. Neural Networks (IJCNN'04)*, Budapest, Hungary, 2004, vol. 3, pp. 2437–2442.

[4] Y. Freund and R.E. Schapire, "Experiment with a new boosting algorithm," *Proc. 13th Int'l Conf. on Machine Learning (ICML'96)*, Bari, Italy, 1996, pp. 148–156.

[5] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[6] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[7] R.E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.

[8] S. Haykin, *Neural Networks: A Comprehensive Foundation, 2nd ed.*, Englewood Cliffs, NJ: Prentice Hall PTR, 1998.

[9] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, 1997.

[10] G. Ridgeway, "The state of boosting," *Computing Science and Statistics*, vol. 31, pp. 172–181, 1999.

[11] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*, New York, NY: Chapman and Hall, 1993.

[12] K. Tumer and J. Ghosh, "Analysis of decision boundaries in linearly combined neural classifiers," *Pattern Recognition*, vol. 29, no. 2, pp. 341–348, 1996.
[13] L.I. Kuncheva, "A theoretical study on six classifer fusion strategies," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281–286, 2002.
[14] Y.F. Wu, J.M. Zhang, C. Wang, and S.C. Ng, "Linear decision fusions in multilayer perceptrons for breast cancer diagnosis," in *Proc. 17th IEEE Int'l Conf. Tools with Artificial Intelligence (ICTAI'05)*, Hong Kong, 2005, pp. 699–700.
[15] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
[16] F. Roli, G. Fumera, and J. Kittler, "Fixed and trained combiners for fusion of unbalanced pattern classifiers," in *Proc. 5th Int'l Conf. Information Fusion (IF'02)*, Annapolis, MD, USA, 2002, vol. 1, pp. 278–284.
[17] Y.F. Wu and J.I. Arribas, "Fusing output information in neural networks: Ensemble performs better," in *Proc. 25th IEEE EMBS Annu. Int'l Conf. (EMBC'03)*, Cancun, Mexico, 2003, vol. 3, pp. 2265–2268.
[18] G. Fumera and F. Roli, "A theoretical and experimental analysis of linear combiners for multiple classifier systems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 942–956, 2005.
[19] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 11, pp. 79–87, 1991.
[20] M.O. Stitson, J.A.E. Weston, A. Gammerman, V. Vovk, and V. Vapnik, "Experiments with support vector machines," Royal Holloway College, University of London, London, UK, Technical Report: CSD-TR-96-19, 1996.
[21] M.T. Hagan and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.
[22] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. 1993 IEEE Int'l Conf. Neural Networks (ICNN'93)*, San Francisco, CA, USA, 1993, vol. 1, pp. 586–591.

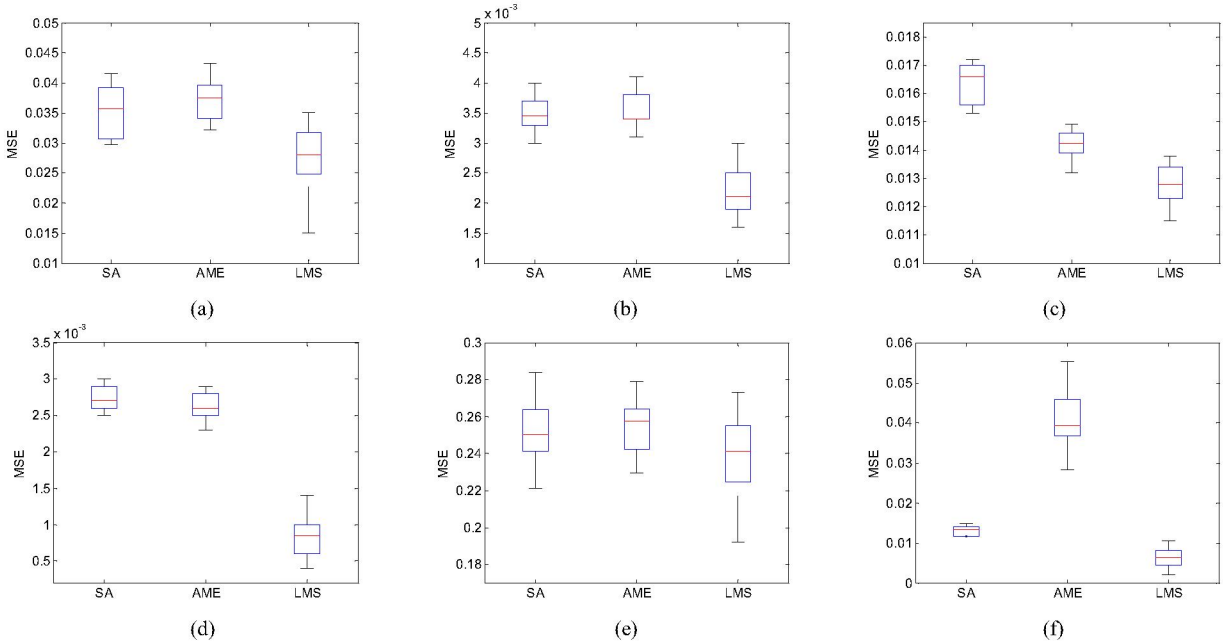


Fig. 2. Performance comparison on regression sets (a) Zigzag, (b) Rhythm, (c) Polynomial, (d) 2-D Mexican Hat, (e) Friedman #1, and (f) Friedman #3.