Ensemble of Minimal Learning Machines for Pattern Classification

Conference Paper · June 2015

DOI: 10.1007/978-3-319-19222-2_12

Diego Parente Paiva Mesquita
Aalto University
27 PUBLICATIONS **119** CITATIONS

Joao Paulo Pordeus Gomes
Universidade Federal do Ceará
102 PUBLICATIONS **423** CITATIONS

Ensemble of Minimal Learning Machines for Pattern Classification

Diego Parente Paiva Mesquita[1], João Paulo Pordeus Gomes[1], and Amauri Holanda Souza Junior[2](✉)

[1] Department of Computer Science, Federal University of Ceará,
Fortaleza, Ceará, Brazil
diego@diegoparente.com, jpaulo@lia.ufc.br

[2] Department of Computer Science, Federal Institute of Ceará,
Maracanaú, Ceará, Brazil
amauriholanda@ifce.edu.br

Abstract. The use of ensemble methods for pattern classification have gained attention in recent years mainly due to its improvements on classification rates. This paper evaluates ensemble learning methods using the Minimal Learning Machines (MLM), a recently proposed supervised learning algorithm. Additionally, we introduce an alternative output estimation procedure to reduce the complexity of the standard MLM. The proposed methods are evaluated on real datasets and compared to several state-of-the-art classification algorithms.

1 Introduction

Recently there has been a lot of interest in ensemble methods for classification. This is mainly due to their ability to achieve high performance in a variety of classification tasks, such as face recognition [1], recognition of spontaneous face expressions [2], hyperspectral remote sensing [3] and character recognition [4]. According to [6] and [7], a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual components is to use classifiers that are accurate and diverse. In this context, a classifier is said to be accurate if it achieves an error classification rate smaller than what is achieved using random guesses. Two classifiers are called diverse if they produce different classification errors. In order to achieve a suitable balance between accuracy and diversity, various strategies have been developed to extend standard classification algorithms to the ensemble framework. Manipulations of training examples [8], selection of subsets of features [9], and injection of randomness in the initialization step [5] are some of the most popular strategies. Ensemble strategies based on combinations of classifiers' results using voting, weighted voting, summation, mean- and median-based averaging schemes are also commonly used [10].

Among the recently proposed supervised learning algorithms, the Minimal Learning Machine (MLM, [11]) has gained attention for its simple and easy implementation, additionally requiring the adjustment of only a single hyperparameter (K, the number of reference points). Learning in MLM consists in

I. Rojas et al. (Eds.): IWANN 2015, Part II, LNCS 9095, pp. 142–152, 2015.
DOI: 10.1007/978-3-319-19222-2_12

building a linear mapping between input and output distance matrices. In the generalization phase, the learned distance map is used to provide an estimate of the distance from K output reference points to the target output value. Then, the output point estimation is formulated as multilateration problem based on the predicted output distance and the locations of the reference points.

This work aims to evaluate the MLM on a ensemble framework using well known methods for ensemble generation and classifier combination. In addition, we propose a simplification of the output estimation step in MLM that speeds up its computational time complexity. Ensembles of MLMs are benchmarked against a number of to state-of-the-art classifiers using UCI classification datasets. The results show that the proposed methods are competitive, achieving higher classification rates than the reference methods in most of the selected problems.

The remainder of the paper is organized as follows. Section 2 introduces the Minimal Learning Machine and discusses the novel proposal for the output estimation step. Section 3 presents methods for ensemble learning using MLMs. The experiments are reported in Section 4. Conclusions are given in Section 5.

2 Minimal Learning Machine

We are given a set of N input points $X = \{\mathbf{x}_i\}_{i=1}^{N}$, with $\mathbf{x}_i \in \mathbb{R}^D$, and the set of corresponding outputs $Y = \{\mathbf{y}_i\}_{i=1}^{N}$, with $\mathbf{y}_i \in \mathbb{R}^S$. Assuming the existence of a continuous mapping $f : \mathcal{X} \to \mathcal{Y}$ between the input and the output space, we want to estimate f from data with the multiresponse model

$$\mathbf{Y} = f(\mathbf{X}) + \mathbf{R}.$$

The columns of the matrices $\mathbf{X}$ and $\mathbf{Y}$ correspond to the D inputs and S outputs respectively, and the rows to the N observations. The columns of the $N \times S$ matrix $\mathbf{R}$ correspond to the residuals.

The MLM is a two-step method designed to

1. reconstruct the mapping existing between input and output distances;
2. estimating the response from the configuration of the output points.

In the following, the two steps are discussed.

2.1 Distance Regression

For a selection of reference input points $R = \{\mathbf{m}_k\}_{k=1}^{K}$ with $R \subseteq X$ and corresponding outputs $T = \{\mathbf{t}_k\}_{k=1}^{K}$ with $T \subseteq Y$, define $\mathbf{D}_x \in \mathbb{R}^{N \times K}$ in such a way that its kth column $\mathbf{d}(X, \mathbf{m}_k)$ contains the distances $d(\mathbf{x}_i, \mathbf{m}_k)$ between the N input points $\mathbf{x}_i$ and the kth reference point $\mathbf{m}_k$. Analogously, define $\boldsymbol{\Delta}_y \in \mathbb{R}^{N \times K}$ in such a way that its kth column $\boldsymbol{\delta}(Y, \mathbf{t}_k)$ contains the distances $\delta(\mathbf{y}_i, \mathbf{t}_k)$ between the N output points $\mathbf{y}_i$ and the output $\mathbf{t}_k$ of the kth reference point.

We assume that there exists a mapping g between the input distance matrix $\mathbf{D}_x$ and the corresponding output distance matrix $\boldsymbol{\Delta}_y$ that can be reconstructed using the multiresponse regression model

$$\boldsymbol{\Delta}_y = g(\mathbf{D}_x) + \mathbf{E}.$$

The columns of the matrix $\mathbf{D}_x$ correspond to the K input vectors and columns of the matrix $\boldsymbol{\Delta}_y$ correspond to the K response vectors, the N rows correspond to the observations. The columns of matrix $\mathbf{E} \in \mathbb{R}^{N \times K}$ correspond to the K residuals.

Assuming that mapping g between input and output distance matrices has a linear structure for each response, the regression model has the form

$$\boldsymbol{\Delta}_y = \mathbf{D}_x\mathbf{B} + \mathbf{E}. \tag{1}$$

The columns of the $K \times K$ regression matrix $\mathbf{B}$ correspond to the coefficients for the K responses. Under the normal conditions where the number of selected reference points is smaller than the number of available points available (i.e., $K < N$), the matrix $\mathbf{B}$ can be approximated by the usual least squares estimate:

$$\hat{\mathbf{B}} = (\mathbf{D}_x'\mathbf{D}_x)^{-1}\mathbf{D}_x'\boldsymbol{\Delta}_y. \tag{2}$$

For an input test point $\mathbf{x} \in \mathbb{R}^D$ whose distances from the K reference input points $\{\mathbf{m}_k\}_{k=1}^K$ are collected in the vector $\mathbf{d}(\mathbf{x}, R) = [d(\mathbf{x}, \mathbf{m}_1) \dots d(\mathbf{x}, \mathbf{m}_K)]$, the corresponding estimated distances between its unknown output $\mathbf{y}$ and the known outputs $\{\mathbf{t}_k\}_{k=1}^K$ of the reference points are

$$\hat{\boldsymbol{\delta}}(\mathbf{y}, T) = \mathbf{d}(\mathbf{x}, R)\hat{\mathbf{B}}. \tag{3}$$

The vector $\hat{\boldsymbol{\delta}}(\mathbf{y}, T) = [\hat{\delta}(\mathbf{y}, \mathbf{t}_1) \dots \hat{\delta}(\mathbf{y}, \mathbf{t}_K)]$ provides an estimate of the geometrical configuration of $\mathbf{y}$ and the reference set T, in the $\mathcal{Y}$-space.

2.2 Output Estimation

The problem of estimating the output $\mathbf{y}$, given the outputs $\{\mathbf{t}_k\}_{k=1}^K$ of all the reference points and estimates $\hat{\boldsymbol{\delta}}(\mathbf{y}, T)$ of their mutual distances, can be understood as a multilateration problem [12] to estimate its location in $\mathcal{Y}$.

Numerous strategies can be used to solve a multilateration problem [13]. From a geometric point of view, locating $\mathbf{y} \in \mathbb{R}^S$ is equivalent to solve the overdetermined set of K nonlinear equations corresponding to S-dimensional hyper-spheres centered in $\mathbf{t}_k$ and passing through $\mathbf{y}$. Figure 1 graphically depicts the problem for $S = 2$.

Given the set of $k = 1, \dots, K$ spheres each with radius equal to $\hat{\delta}(\mathbf{y}, \mathbf{t}_k)$

$$(\mathbf{y} - \mathbf{t}_k)'(\mathbf{y} - \mathbf{t}_k) = \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k), \tag{4}$$

the location of $\mathbf{y}$ is estimated from the minimization of the objective function

$$J(\mathbf{y}) = \sum_{k=1}^{K} \Big((\mathbf{y} - \mathbf{t}_k)'(\mathbf{y} - \mathbf{t}_k) - \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k) \Big)^2. \tag{5}$$

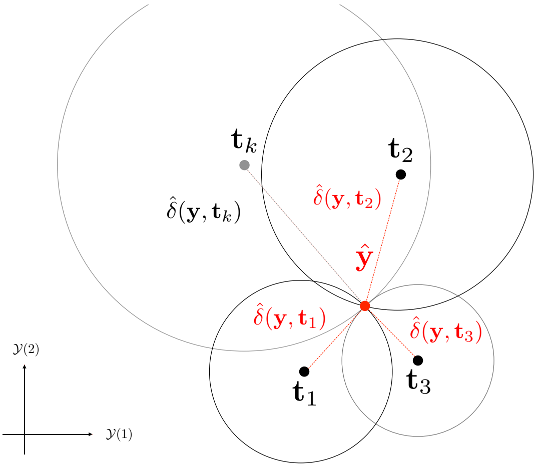
$\mathbf{t}_k$
$\mathbf{t}_2$
$\hat{\delta}(\mathbf{y}, \mathbf{t}_k)$
$\hat{\delta}(\mathbf{y}, \mathbf{t}_2)$
$\hat{\mathbf{y}}$
$\hat{\delta}(\mathbf{y}, \mathbf{t}_1)$
$\hat{\delta}(\mathbf{y}, \mathbf{t}_3)$
$\mathcal{Y}(2)$
$\mathbf{t}_1$
$\mathbf{t}_3$
$\mathcal{Y}(1)$

Fig. 1. Output estimation

The cost function has a minimum equal to 0 that can be achieved if and only if $\mathbf{y}$ is the solution of (4). If it exists, such a solution is thus global and unique. Due to the uncertainty introduced by the estimates $\hat{\delta}(\mathbf{y}, \mathbf{t}_k)$, an optimal solution to (5) can be achieved by any minimizer $\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmin}}\ J(\mathbf{y})$ like the nonlinear least square estimates from standard gradient descent methods. The original MLM proposal applies the Levenberg-Marquardt (LM) method [14] to solve the output estimation step.

2.3 Extension to Classification

An important class of problems is classification, where we are concerned with the prediction of categorical variables or class labels. For the task, we are still given N input points $X = \{\mathbf{x}_i\}_{i=1}^N$, with $\mathbf{x}_i \in \mathbb{R}^D$, and corresponding class labels $L = \{l_i\}_{i=1}^N$, with $l_i \in \{C_1, \ldots, C_S\}$, where C_j denotes the j-the class. For $S = 2$, we have binary classification, whereas for $S > 2$ we have multi-class classification.

The MLM can be extended to classification in a straightforward manner by representing the S class labels in a vectorial fashion through an 1-of-S encoding scheme [15]. In such approach, a S-level qualitative variable is represented by a vector of S binary variables or bits, only one of which is *on* at a time. In the classification of a test observation $\mathbf{x}$ of unknown class label $l \in \{C_1, \ldots, C_S\}$, the estimated class $\hat{l}$ associated to the output estimate $\hat{\mathbf{y}}$ is $\hat{l} = C_{s^*}$, where

$$s^* = \underset{s=1,\ldots,S}{\text{argmax}}\ \{\hat{y}^{(s)}\}. \tag{6}$$

Given this formulation, the Minimal Learning Machine provides a general framework that can be used for regression, binary and multi-class classification.

Complexity Analysis. The training procedure of the Minimal Learning Machine can be roughly decomposed into two parts: *i*) calculation of the pairwise distance matrices in the output and input space; *ii*) calculation of the least-square solution for the multiresponse linear regression problem on distance matrices. The first part takes $\Theta(KN)$ time. The computational cost of the second part is driven by the calculation of the Moore-Penrose pseudoinverse matrix, which runs in $\Theta(K^2N)$ time if we consider the SVD algorithm.

In order to establish a comparison, the MLM training computational cost is similar to what is presented by an Extreme Learning Machine when the number of hidden neurons is equal to the number of reference points. It is worthy to notice that the ELM is considered one of the fastest methods for nonlinear regression and classification tasks.

Concerning the computational analysis of the generalization (output estimation) step in MLM, we consider the Levenberg-Marquardt method due to its fast and stable convergence, even though any gradient descent method can be used on the minimization step in Eq. 5. For each iteration, the LM method involves the computation of the Jacobian $\mathbf{J} \in \mathbb{R}^{K \times S}$ and the inverse of $\mathbf{J}^T\mathbf{J}$. The computational complexity of the LM algorithm is approximately $\Theta(I(KS^2 + S^3))$, where S is the dimensionality of $\mathbf{y}$ and I denotes the number of iterations.

Speeding up the Output Estimation. The MLM was proposed as a general supervised learning method, capable of dealing with classification and regression tasks. This resulted in a general formulation that does not take advantage of the particularities of each task. Considering the classification case, one can notice that the finite and discrete set of possible outputs may help finding a solution to the output estimation without solving an optimization problem. In fact, the output estimation step is the computationally critical part in MLM in comparison to standard classification methods. Thus, reducing the computational complexity of the output estimation step is particularly useful in the context of ensemble methods, since such an step is extensively computed for each ensemble member.

The output estimation part aims to find the best estimation for the label (numerical output vector) $\hat{\mathbf{y}}$ of a given input test vector $\mathbf{x}$ from estimated distances to the reference points $\hat{\boldsymbol{\delta}}(\mathbf{y}, T)$ in the output space. In the classification task, since the set of feasible values for $\hat{\mathbf{y}}$ is limited to the number of classes, a few trials should be computed to select an estimate to the output. Moreover, an alternative to trying all possible values is to take the output (label) associated with the nearest reference point in the output space, whose distance is likely to be about 0 — given that all the classes are represented in the set of reference points.

Consider a classification problem with S classes. In this situation, the reference output points $\mathbf{t}_k$ assume S possible values and the objective function (5) can be rewritten as:

$$J(\mathbf{y}) = \sum_{s=1}^{S} N_s \Big((\mathbf{y} - \mathbf{t}^s)'(\mathbf{y} - \mathbf{t}^s) - \hat{\delta}^2(\mathbf{y}, \mathbf{t}^s) \Big)^2 \tag{7}$$

where N_s is the number of reference points belonging to class C_s, and $\mathbf{t}^s$ denotes the vectorial representation (1-of-S encoding) of the class C_s.

For the sake of simplicity, let us assume that all classes are equally represented in the set of reference points. Therefore, the factor N_s can be neglected from the objective function. The argument $(\mathbf{y} - \mathbf{t}^s)'(\mathbf{y} - \mathbf{t}^s)$ may assume only two possible values: zero, when $\mathbf{y} = \mathbf{t}^s$, or a positive number, otherwise. Similarly, the estimated distance $\hat{\delta}^2(\mathbf{y}, \mathbf{t}^s)$ return S distinct values: the estimate for the distance to the correct class (supposed to be 0 in a perfect reconstruction of the distance), and estimates for the distances to the other classes (positive numbers). We are interested in minimizing Eq. 7. It is accomplished when we set $\mathbf{y} = \mathbf{t}^{s^*}$ to the class s^* with smallest distance estimate, i.e., $s^* = \underset{s=1,\ldots,S}{\operatorname{argmin}}\ \hat{\delta}^2(\mathbf{y}, \mathbf{t}^s)$. It means that the classification step can be carried out based on the label of the reference point associated to the smallest output distance estimate. In practice, we use the estimated distances $\hat{\delta}(\mathbf{y}, \mathbf{t}_k)$ given from Eq. 3 in such a way that the estimated class $\hat{l}$ for an input pattern $\mathbf{x}$ is given by $\hat{l} = l_{k^*}$, where

$$k^* = \underset{k=1,\ldots,K}{\operatorname{argmin}}\ \hat{\delta}^2(\mathbf{y}, \mathbf{t}_k), \tag{8}$$

and l_{k^*} represents the label of the k^*th reference point.

This strategy corresponds to carrying out a nearest neighbors classifier based on the estimated distances in the output space. It turns out the computational complexity of the output estimation step to $O(K)$ with small constant factor. One may notice that only S distinct values need to be evaluated in order to find the minimum estimated distance.

3 Ensemble Strategies

The proposed ensemble methods consists of two strategies for manipulating the training examples, and two output combination schemes. Manipulating the training examples is one of the simplest strategies to generate different classifiers. For example, it can be achieved by resampling the training set or by using different samples in the model training. These strategies aim to increase the classifiers variability and consequently the overall ensemble performance. On the other hand, a combination strategy consists of combining the output of each ensemble member to generate a final consensus output. Many strategies have been proposed and the most commonly used one is based on a majority voting scheme.

The strategies used in this work are detailed in the following.

3.1 Selecting Training Examples and Reference Points

The standard procedure for selecting the number of reference points K in MLM is based on resampling methods, e.g., cross-validation. Then, K randomly selected points from the learning points comprise the set of reference points. The first ensemble strategy consists of using M classifiers with the standard procedure to select K. By doing so, we expect the classifiers to differ with respect to the randomly chosen reference points. Henceforth this procedure will be referred as *sampling procedure 1.*

The second strategy consists of inserting uncertainty by using a randomly selected fraction P of the total available training data as learning points to each of the M classifiers. Then, all the learning points are used as reference points ($K = P$). This procedure will be referred to as *sampling procedure 2.* One may notice that, in the procedure 2, the fraction P (and consequently K) is defined beforehand whereas that in the first procedure it is selected through cross-validation. Also, the number of samples used for learning is different in the two strategies.

3.2 Combination Strategies

The combination of the classifiers' outputs is based on two approaches. The first method uses the majority voting scheme; in the case, an out-of-sample point is assigned to the most voted class among all the classifiers. It corresponds to the standard voting method.

The second approach consists of a weighted majority voting; in this case, each classifier vote is associated with a weighting factor. Such a factor is related to the classifier prediction confidence. Assuming the 1-of-S encoding scheme for the classifiers, the weight associated with each classifier is given by

$$w = \frac{\max_{s=1,\ldots,S} \hat{y}^{(s)}}{\sum_{s=1}^{S} \hat{y}^{(s)}}, \tag{9}$$

where $\hat{y}^{(s)}$ is the sth component of the output estimate vector $\hat{\mathbf{y}}$. It is straightforward noticing that the larger is the maximum component value of $\hat{\mathbf{y}}$ in comparison to the other components, the higher is the prediction confidence and, consequently, the associated weight w. After calculating the weights for each classifier, the class is chosen according to the weighted majority voting.

4 Performance Evaluation

Using the described procedures for ensemble generation, it is possible to create 4 different MLM ensembles. Combining *procedure 1* with the voting and the weighted voting scheme generates, respectively, the voting based MLM (V-MLM) and the weighted voting based MLM (WV-MLM). The combination of *procedure 2* with the voting and the weighted voting scheme generates, respectively, the random sampling voting based MLM (RSV-MLM) and the random sampling weighted voting based MLM (RSWV-MLM).

The performances of the proposed MLM ensemble strategies are compared to standard MLM and some state of the art methods under real-world datasets. Simulations using V-ELM, SVM, OP-ELM, BP and and KNN are conducted on 10 UCI datasets. Datasets used are described in Table 1.

Table 1. Datasets description

Datasets	Attributes \#	Classes \#	Training data \#	Testing data \#
Balance	4	3	400	225
Breast	30	2	300	269
Diabetes	8	2	576	192
Glass	10	2	100	114
Heart	13	2	100	170
Sonar	60	2	100	108
Wine	13	3	100	78
Monk 1	6	2	124	432
Monk 2	6	2	169	432
Monk 3	6	2	122	432

For the first 7 datasets in Table 1, at each trial we deal the data at random between the training and testing set. For the remaining ones, training and testing data are fixed for all trials. Each value in Table 2 and Table 3 reflects the outcome of 50 similar trials.

4.1 Comparisons with MLM

Table 2 shows the performance of the proposed methods compared to the MLM. For WV-MLM, V-MLM and MLM the number of reference point was selected using 10-fold cross validation and the reference point were chosen randomly from the training set. For RSV-MLM and RSWV-MLM the value adopted for P was 0.8. All MLM ensembles in the experiments are comprised of 7 MLMs.

As expected, ensemble MLM strategies achieved higher accuracies when compared to the standard MLM for most datasets. Particularly, WV-MLM achieved the best results in 4 of the datasets. It is also important to notice that for most datasets, the ensemble methods achieved a low standard deviation.

4.2 Comparisons with SVM, OP-ELM, BP, KNN and V-ELM

Table 3 compares the performance of MLM ensembles against SVM [17], OP-ELM [18], BP [19], KNN [20] and V-ELM [5]. MAX-MLM denotes the results from the MLM variant with higher average accuracy on Table 2. For SVM, the Gaussian RBF is used as the kernel function, the cost parameter C and the kernel parameter γ are searched in a grid formed by $C = [2^{12}, 2^{11}, \ldots, 2^{-2}]$ and $\gamma = [2^4, 2^3, \ldots, 2^{-10}]$. For OP-ELM, the three possible kernels, linear, sigmoid and Gaussian are used as a combination. For BP, Levenberg-Marquardt algorithm is used to train the neural network. For KNN, 7 nearest neighbors are used and the Euclidean norm is adopted to calculate the distance. For V-ELM, 7 independent ELMs are adopted for training and majority voting, the number of hidden node is gradually increased up to 50 and chosen using cross validation.

Table 2. Performance comparison with MLM

Datasets	MLM		V-MLM		RSV-MLM		WV-MLM		RSWV-MLM	
	Acc.	Dev.	Acc.	Dev.	Acc.	Dev.	Acc.	Dev.	Acc.	Dev.
Balance	89.85	1.43	89.97	1.54	89.31	1.45	**90.20**	1.48	89.75	1.45
Breast	97.19	0.65	97.33	0.68	**97.40**	0.68	97.33	0.68	97.33	0.67
Diabetes	75.49	2.50	**76.03**	2.46	74.07	2.57	75.92	2.57	74.36	2.46
Glass	95.71	3.16	96.04	2.96	96.00	2.80	**96.05**	2.92	95.81	2.96
Heart	82.01	2.28	82.52	2.19	81.25	245	**82.56**	2.33	81.49	2.54
Sonar	**82.03**	4.11	81.92	4.20	82.00	3.79	81.79	4.10	81.46	3.92
Wine	98.41	1.23	98.41	1.23	98.43	1.27	98.41	1.23	**98.48**	1.28
Monk 1	82.04	2.17	83.98	0.78	**84.28**	0.56	83.25	0.66	83.46	0.78
Monk 2	**82.17**	0.00	**82.17**	0.00	82.02	0.61	**82.17**	0.00	81.86	0.68
Monk 3	93.32	0.06	93.51	0.00	93.37	0.36	93.54	0.07	**93.73**	0.28

Table 3. Performance comparison between MLM variants, SVM, OP-ELM, BP, KNN and V-ELM

	MAX-MLM		SVM		OP-ELM		BP		KNN		V-ELM	
	Acc.	Dev	Acc.	Dev	Acc.	Dev	Acc.	Dev	Acc.	Dev	Acc.	Dev
Balance	90.2	1.48	**95.88**	1.31	92.31	1.83	90.92	2.14	87.00	1.8	91.24	1.49
Breast	**97.40**	0.68	95.55	0.82	95.33	1.29	95.01	1.66	96.32	1.03	96.75	0.94
Diabetes	76.03	2.46	77.31	2.73	77.34	3.17	77.23	2.81	74.09	2.73	**78.56**	2.46
Glass	**96.05**	2.92	91.84	2.78	91.65	3.23	91.30	3.09	90.18	2.60	93.33	2.21
Heart	**82.56**	2.33	76.10	3.46	81.05	2.96	71.25	8.54	80.79	2.57	82.53	2.27
Sonar	82.03	4.11	**83.48**	3.88	71.70	4.79	70.31	5.40	66.30	4.93	79.11	3.51
Wine	**98.48**	1.28	97.48	1.57	98.18	1.72	94.10	3.12	96.23	2.01	98.31	1.61
Monk1	84.28	0.56	**94.44**	0.01	74.79	3.91	69.99	13.82	80.56	0.01	85.75	1.41
Monk2	82.17	0.00	84.72	0.01	70.35	3.58	72.84	2.92	71.53	0.01	**85.84**	1.27
Monk3	**93.73**	0.28	90.04	0.01	88.77	2.31	80.41	6.07	80.79	0.01	90.44	1.00

Detailed procedure for all methods, except for the MLM approaches, can be found in [5].

From the table, one can see that, regarding accuracy, the MLM variants outperform the other methods in 5 out of 10 cases, while SVM wins in 3 cases and V-ELM wins in 2. As for the standard deviation, the MLM variants are usually comparable to SVM and V-ELM, while the other methods present significantly higher deviation. It is worth highlighting the performance of the MLM ensembles on the datasets with the highest number of features. The proposed methods achieved better classification rates in 4 out of 5 datasets of highest dimensionality. Another important result can observed when comparing the proposed methods with V-ELM. The MLM ensembles outperformed V-ELM in 6 out of 10 datasets.

5 Conclusion

This work evaluates Minimal Learning Machine ensembles for classification tasks. Four MLM ensembles were proposed based on sampling and voting strategies. Additionally, we introduced a fast alternative for computing the output estimation step in the MLM, thus allowing its usage for ensemble learning.

Despite the simple ensemble generation strategies used in this work, MLM ensembles showed promising results, outperforming some state-of-the-art algorithms in many of the evaluated datasets. Future works may investigate different strategies for ensemble generation using boosting approaches.

Acknowledgments. The authors acknowledge the support of CNPq (Grant 456837/2014-0).

References

1. Lu, J., Plataniotis, K.N., Venetsanopoulos, A.N., Li, S.Z.: Ensemble-based discriminant learning with boosting for face recognition. IEEE Transactions on Neural Networks **17**, 166–178 (2006)
2. El Abd Meguid, M.K., Levine, M.D.: Fully automated recognition of spontaneous facial expressions in videos using random forest classifiers. IEEE Transactions on Affective Computing **5**, 141–154 (2014)
3. Chen, Y., Zhao, X., Lin, Z.: Optimizing Subspace SVM Ensemble for Hyperspectral Imagery Classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **7**, 1295–1305 (2014)
4. Asadi, N., Mirzaei, A., Haghshenas, E.: Multiple Observations HMM Learning by Aggregating Ensemble Models. IEEE Transactions on Signal Processing **61**, 5767–5776 (2013)
5. Cao, J., Lin, Z., Huang, G.-B., Liu, N.: Voting based extreme learning machine. Information Sciences **185**, 66–77 (2012)
6. Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence **12**, 993–1001 (1990)
7. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, p. 1. Springer, Heidelberg (2000)
8. Yoav, F., Schapire, R.E.: Experiments with a New Boosting Algorithm. Proceedings of the International Conference on Machine Learning **1**, 148–156 (1996)
9. Tsymbal, A., Pechenizkiy, M., Cunningham, P.: Diversity in search strategies for ensemble feature selection. Information Fusion **6**, 83–98 (2005)
10. Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**, 226–239 (1998)
11. de Souza Junior, A.H., Corona, F., Miche, Y., Lendasse, A., Barreto, G.A., Simula, O.: Minimal learning machine: a new distance-based method for supervised learning. In: Rojas, I., Joya, G., Gabestany, J. (eds.) IWANN 2013, Part I. LNCS, vol. 7902, pp. 408–416. Springer, Heidelberg (2013)
12. Niewiadomska-Szynkiewicz, E., Marks, M.: Optimization Schemes For Wireless Sensor Network Localization. International Journal of Applied Mathematics and Computer Science **19**, 291–302 (2009)
13. Navidi, W., Murphy Jr., W.S., Hereman, W.: Statistical methods in surveying by trilateration. Computational Statistics & Data Analysis **27**, 209–227 (1998)

14. Marquardt, D.W.: An Algorithm for Least-Squares Estimation of Nonlinear Parameters. Journal of the Society for Industrial and Applied Mathematics. **11**, 431–441 (1963)
15. Souza Junior, A.H., Corona F., Miché Y., Lendasse, A., Barreto, G.: Extending the minimal learning machine for pattern classification. In: Proceedings of the 1st BRICS countries conference on computational intelligence, vol. 1, pp. 1–8 (2013)
16. Frank, A., Asuncion, A.: UCI Machine Learning Repository University of California. Irvine, School of Information and Computer Sciences (2010)
17. Hsu, C.W., Lin, C.J.: A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks **13**, 415–425 (2002)
18. Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: OP-ELM: Optimally Pruned Extreme Learning Machine. IEEE Transactions on Neural Networks **21**, 158–162 (2010)
19. Haykin, S.: Neural Networks, A Comprehensive Foundation 2nd ed., Pearson education Press (2001)
20. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory **13**, 21–27 (1967)