

Improving Predictions Using Linear Combination of Multiple Extreme Learning Machines

Pedro J. García-Laencina

*Centro Universitario de la Defensa de San Javier,
(University Centre of Defence at the Spanish Air Force Academy)
Ministerio de Defensa-Universidad Politécnica de Cartagena*

crossref <http://dx.doi.org/10.5755/j01.itc.42.1.1667>

Abstract. This work presents several effective approaches for linear combination of multiple artificial neural networks based on the extreme learning machine (ELM) algorithm. Given a learning task, a large set of neural networks are firstly trained by ELM. Then, these trained machines are efficiently ranked and the useless models are effectively discarded in order to provide an ensemble system with better generalization performance. The ensemble system is constructed using an automatic and fast forward model selection by minimizing the leave-one-out error, without user intervention. Experiments on an artificial regression dataset and three real-world engineering problems are discussed. According to the obtained results, the weighted linear combination of ELMs improves predictions by exploiting model diversity in the ensemble system with fast learning speed.

Keywords: Artificial neural network (ANN); Extreme learning machine (ELM); Linear combination; Ensembles; Regression.

1. Introduction

Artificial neural networks (ANN) have been successfully applied to many engineering and science fields [1]. Thus, they have been used to solve problems in industrial applications as control processes, fault detection and quality evaluation; in security applications based on biometric recognition; in medical applications as diagnosis, detection and evaluation of medical phenomena; in financial applications as stock market prediction, credit worthiness, credit rating, bankrupt prediction, and prices forecasts; and finally, in many others fields of the science as biological systems analysis, botanical recognition, recipes and chemical formulation optimization, and ecosystem evaluation.

ANN modelling often entails the construction of multiple networks with different architectures, learning procedures and training parameters in order to obtain an accurate model. Typically, one of the trained ANNs is usually chosen as the ‘best’ model and the remaining networks are discarded. Instead of using an individual network, it is widely known that combining the estimates of multiple models can improve the performance of the best network in the ensemble [2]–[5].

In general, a neural network ensemble is constructed in two main steps: training a number of

component neural networks and, then, combining the component predictions. One of the major drawbacks in the first step is that traditional training methods do not provide an efficient implementation. It is due to the many parameters to be properly tuned by slow (often gradient-based) algorithms for obtaining a good enough model. Besides, a large number of models with different complexity have to be trained for exploiting model diversity in the ensemble system. Due to this, this work makes use of the extreme learning machine (ELM) algorithm, which has been introduced to solve the speed limitation in the learning process of single layer feedforward neural networks (SLFNs) [6]–[9]. Different from the common understanding of learning, the hidden layer of SLFNs does not need to be tuned using the ELM algorithm: it applies random computational nodes in the hidden layer and, then, analytically determines the output weights using the Moore-Penrose generalized inverse. The ELM algorithm tends to provide good generalization performance at extremely fast learning speed. However, the optimal network size is usually unknown and tedious experimentation becomes necessary to find a good model. Several automatic growing and pruning approaches have been proposed for ELM architectures, being the Optimally Pruned ELM (OP-ELM) methodology one of the most

appropriate techniques for robust and fast design of ELM networks [10].

With respect to the second stage of the ensemble construction, the output of the ensemble system is computed by combining the outputs of each model, being simple averaging or weighted averaging the most prevailing approaches for regression tasks. At this point, it is important to remark that, in practice, most ensemble approaches use all of those trained networks to design the ensemble system and assign weight values according to its contribution to the ensemble. Some recent works have proposed adaptive ensembling approaches for neural networks [4], but their major drawbacks are that they are sensitive to random subsampling of validation sets and, also, they require to tune a threshold parameter for performing model selection. In this work, the most useful/accurate models for the ensemble are fast automatically chosen and it is done without user intervention. Particularly, after the different networks have been trained, their outputs are ranked according to the multiresponse sparse regression (MRSR) method, which determines the contribution of each individual model in the ensemble to the global prediction of the target variable. Once the candidate networks are ranked, the better models are chosen following an incremental forward selection methodology by fast minimization of the leave-one-out (LOO) error, which is exactly measured with the predicted residual sums of squares (PRESS) statistic. As it is described in next sections, the model selection is not sensitive to random subsampling of validations sets (all samples are used for training and validation) and there is not any tuneable parameter to perform model selection.

Finally, it is widely known that the success of an ensemble system relies upon the diversity among the individual models, i.e. hypotheses disagree with each other in many of the individual model predictions. Diversity is a key aspect since it may determine whether the ensemble will perform better than its individual components. In this work, model diversity is exploited by varying the input layer of the individual models. Three different scenarios are evaluated: the original feature input space, the resulting reduced feature subset by forward selection and random input subspaces.

The rest of this paper is organized as follows: Section 2 gives an overview of the ELM algorithm for training SLFNs and some of its extensions for automatic design of the neural architecture. The ensemble approaches for ELM networks are described in Section 3 and Section 4 gives experimental results on four well-known regression problems. Finally, the main conclusions and future works are reported in Section 5.

2. ELM algorithm for training SLFNs

Consider a dataset defined by N samples (x_j, t_j) , where $x_j = [x_{j1}, x_{j2}, \dots, x_{jn}]^T \in \mathbb{R}^n$ is the j -th input vector and $t_j \in \mathbb{R}$ is its corresponding target value. This dataset is learned using a SLFN with M hidden neurons and activation function $f(\cdot)$, which is mathematically modeled as

$$o_j = \sum_{i=1}^M v_i f(\mathbf{w}_i^T \mathbf{x}_j + b_i) = \sum_{i=1}^M v_i h_{ij}, \quad j = 1, \dots, N; \quad (1)$$

where $w_i = [w_{i1}, w_{i2}, \dots, w_{in}]^T$ is the weight vector connecting the i -th hidden neuron and the input units, v_i is the weight connecting the i -th hidden neuron and the output unit, and b_i is the threshold of the i -th hidden neuron. Besides, h_{ij} is the i -th hidden output for x_j . Note that the network output unit is linear. Figure 1 shows a scheme of a SLFN architecture with single hidden layer of neurons. One of the most popular activation function is the sigmoid and, in this case, the SLFN is widely known as multi-layer perceptron (MLP). The learning objective is that $o_j \approx t_j$ with a good generalization capability.

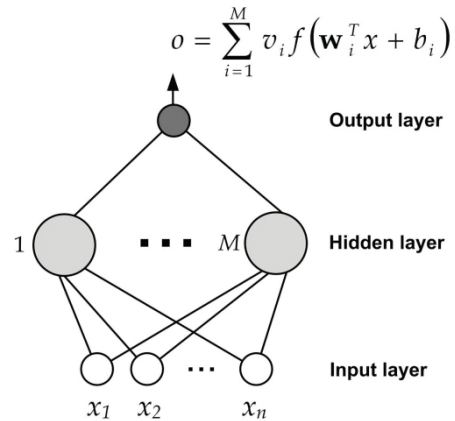


Figure 1. Standard neural architecture of a SLFN with single hidden layer of M neurons. The input layer is composed of n and the output layer has one unit

The standard ELM method is based on the concept that if input weights w_i and biases b_i are randomly assigned, then a SLFN can be considered as a linear system and the output weight vector, \mathbf{v} , can be analytically determined through simple generalized inverse operation of the hidden layer output matrices [6]–[8]. For fixed w_i and b_i , the training of a SLFN is simply equivalent to find a leastsquares solution, $\hat{\mathbf{v}}$, for the linear system $\mathbf{H}\mathbf{v} = \mathbf{t}$, where

$$\mathbf{H} = \begin{bmatrix} f(\mathbf{w}_1^T \mathbf{x}_1 + b_1) & \dots & f(\mathbf{w}_M^T \mathbf{x}_1 + b_M) \\ \vdots & \dots & \vdots \\ f(\mathbf{w}_1^T \mathbf{x}_N + b_1) & \dots & f(\mathbf{w}_M^T \mathbf{x}_N + b_M) \end{bmatrix}_{N \times M} \quad (2)$$

From [7], [8], the solution is analytically given by $\hat{\mathbf{v}} = \mathbf{H}^\dagger \mathbf{t}$, where \mathbf{H}^\dagger is the Moore-Penrose generalized inverse of \mathbf{H} , [11]. In particular, the implementation of ELM uses singular value decomposition (SVD) to

calculate \mathbf{H}^\dagger for faster computations and numerical stability. ELM is a batch learning approach much simpler and faster than traditional learning algorithms for SLFN. Without iteratively tuning parameters as in standard training procedures, the learning speed of ELM can be thousands of times faster than the gradient based learning algorithms. Given enough hidden neurons, the performance of this simple learning algorithm is comparable to traditional gradient based algorithms in terms of predictive accuracy results.

A. Automatic design of neural networks based on ELM

As it have been already mentioned, the ELM algorithm provides an analytical training procedure for a SLFN with a fixed architecture (i.e., a predefined hidden layer size, M) and randomly assigned hidden parameters. However, the optimal network size is usually unknown and tedious experimentation becomes necessary to find a good network. It means that an exhaustive search of M is done by means of a cross-validation procedure [1], which may imply an excessive computational cost in large datasets. Although the advantages of the ELM algorithm have been proved in many scientific areas, it has been found that the obtained networks tend to require more hidden nodes than traditional training methods because the random assignment of parameters may introduce inappropriate values for them. Several growing [12]–[14] and pruning [10], [15], [16] improvements for the ELM algorithm have been proposed in order to obtain a more compact network and to avoid an extensive search of the optimal hidden layer size. It has been shown that, in general, the growing techniques are more sensitive to the random weight initialization than pruning procedures and, then, they can be trapped in a sub-optimal solution. From the different pruning approaches, the optimally pruned ELM (OP-ELM) methodology stands out as a robust and fast technique for automatic design of ELM networks [10].

The OP-ELM method [10] improves the standard ELM algorithm by pruning inappropriate hidden nodes using an exact and efficient ranking criterion, which is based on the multiresponse sparse regression (MRSR) algorithm and the prediction sum of squares (PRESS) statistic. As it is shown in [10], OP-ELM provides extremely fast accurate models and achieves roughly the same level of accuracy as that of other well known machine learning methods, such as support vector machines (SVM) or gaussian processes (GP). In what follows, the three main stages of the OPELM method are outlined.

1. Random initialization of a large SLFN. This first step is performed using the standard ELM algorithm for a large enough number of neurons, M . From a practical point of view, it is advised to set M clearly above the number of input features: $M \gg n$. For sigmoid functions, the weights are drawn randomly

from an uniform distribution in an interval that covers the input data range (previously normalized to zero mean and unit variance) [10].

2. *Ranking of hidden units.* In a second stage, the MRSR algorithm is applied in order to rank the hidden neurons according to their accuracy. This method is based on a regularization approach inspired by the LASSO methodology [17], which minimizes the residual sum of squares while bounding the L_1 -norm of the weight vector by a specified value. MRSR is in essence an extension of the wellknown least angle regression (LARS) algorithm and, hence, it is a variable ranking technique [18], rather than a selection one. It must be outlined that the obtained ranking by MRSR is exact for linear problems [17]. Since the output of an ELM network is linear with respect to the randomly initialized hidden units, the obtained ranking of neurons by MRSR is exact [10]. More details of the MRSR method can be found in [17].
3. Selection of hidden units. Once the ranking of the hidden neurons has been obtained and \mathbf{H} is consequently ordered, the best M^* hidden units for the ELM model are chosen using the LOO error, which can be exactly calculated for linear models by using the PRESS statistic [19]. Consider that $\hat{\mathbf{v}}_{(j)}$ is the least-squares error solution when the j -th sample (j -th row of \mathbf{H}) is omitted, i.e.,

$$\hat{\mathbf{v}}_{(j)} = \mathbf{H}_{(j)}^\dagger \mathbf{t}_{(j)}, \quad (3)$$

where $\mathbf{H}_{(j)}$ and $\mathbf{t}_{(j)}$ are, respectively, \mathbf{H} without its j -th row and \mathbf{t} without its j -th component vector.

Then, the LOO error is given by

$$\epsilon_{PRESS} = \frac{1}{N} \sum_{j=1}^N \|\mathbf{h}_j \hat{\mathbf{v}}_{(j)} - \mathbf{t}_{(j)}\|^2, \quad (4)$$

where \mathbf{h}_j is the j -th row of \mathbf{H} . To select a model by using the LOO error as criterion, the ϵ_{PRESS} for each model is computed and the model with a minimal ϵ_{PRESS} is chosen. Directly using the previous formula the computation of ϵ_{PRESS} would be extremely inefficient and computationally prohibitive since the model has been computed as many times as the number of samples. However, it is well-known that there is a non-iterative and exact formula giving the PRESS statistic as:

$$\epsilon_{PRESS} = \frac{1}{N} \|\Gamma(\mathbf{I} - \mathbf{P})^{-1}(\mathbf{P} - \mathbf{I})\|_{\mathcal{F}}^2, \quad (5)$$

with $\mathbf{P} = \mathbf{H}\mathbf{H}^\dagger$, \mathbf{I} is the identity matrix and where $\Gamma(\mathbf{A})$ keeps only the diagonal entries of the square matrix \mathbf{A} . In this work, $\|\mathbf{A}\|_{\mathcal{F}}$

denotes the matrix Frobenius norm of \mathbf{A} . A proof of formula (5) is given in [20].

For OP-ELM networks, the PRESS statistic is iteratively computed by adding a hidden node (which are previously ranked in \mathbf{H}) to the model. The model with M^* hidden neurons obtains the lowest PRESS statistic and this model is considered optimal, i.e.,

$$\epsilon_{PRESS}^{M^*} < \epsilon_{PRESS}^k, \forall k \in (1, 2, \dots, M). \quad (6)$$

On the contrary to the standard ELM algorithm, OPELM does not need to divide learning set into training and validation subsets because it directly determines the optimal hidden layer size by computing the LOO error with the PRESS statistic. Therefore, OP-ELM is faster than the standard ELM algorithm and it uses more training samples. Note that OP-ELM provides a SLFN with different hidden layer size for each input weight initialization.

3. Linear combination of ELMs

It is widely known that the performance of a SLFN can be improved by combining several solutions to the same task, i.e., by combining the outputs of different individual ELMs. Rather than generating a population of L models and using a single 'best' model in isolation, a combination of these L networks would exploit, rather than ignore, the information contained in the redundant models. Such combination of models is sometimes called *committees* or *ensembles*. Two main issues arise when considering the ensemble approach: first, the creation of L models to be combined in an ensemble; and second, the method by which the outputs of the members are combined [2]. In the first step, note that when the trained networks which all generalised identically are combined in an ensemble system, there will be no improvement in performance between the outputs from the individual models and the ensemble of them. In order to the models generalize differently, they must each have their initial conditions varied prior to creation (for example, by varying initial weights or topology) or different training data set (for example, by varying training sets or input features). With respect to the second issue, the simplest way to construct a committee is to average the predictions of a set of L individual ELM models (with the same number of hidden neurons and different weight initializations) [21], but other combination schemes are also possible.

Liu and Wang presented an ensemble ELM method which uses a cross-validation scheme in order to increase the diversity between the individual models [22]. This procedure trains K ELM networks on different R pairs of data sets which are generated by cross-validation. Whereas, [22] uses the same hidden layer size for all $R \times K$ ELMs and it does not perform a selection of appropriate candidate networks for the ensemble. In order to discard inaccurate individual models, Chen et al. [23] measure the

diversity between individual models (with a predefined hidden layer size) and the ensemble output using the Pearson correlation coefficient and then computes its product with the mean square error (MSE) of each model. In particular, it discards a network from the ensemble if its corresponding product is greater than the mean value of all product results [23], which means that discarded models have large MSE and small diversity. Heeswijk et al. [24] introduce an adaptive ensemble of random ELM networks for time series prediction where its ensemble weights (with positivity constraints) are iteratively updated according to a predefined learning rate. This procedure can be very expensive in computational terms. For solving this inconvenient, Heeswijk et al. [25] implement the ELM algorithm using GP (Graphics Processing) computing (instead of standard CPU implementation) in order to reduce the training time. Nevertheless, this technique is quite sensitive on the correct choice of the validation set and its results vary if the analysis is repeated with different random splits.

This section presents robust and fast approaches for ensembling ELMs by exploiting model diversity. In particular, consider a linear combination scheme of L ELM networks,

$$y_j = \sum_{l=1}^L \lambda_l o_l, \quad (7)$$

where y is the ensemble output and λ_l is l -th ensemble weight. Thus, given the N input vectors, the following problem must be solved:

$$y_j = \sum_{l=1}^L \lambda_l o_l \approx t_j, \quad j = 1, \dots, N; \quad (8)$$

where o_l is the prediction of the l -th model given x_j and λ_l is the ensemble weight vector connecting the l -th model and the output units. The ensemble weights can be obtained by the least-squares solution of (8): $\lambda = \mathbf{O}^\dagger \mathbf{T}$, where \mathbf{O}^\dagger denotes the Moore-Penrose generalized inverse of the network output matrix \mathbf{O} , whose l -th column is the l -th network's output vector for the N input vectors. Here, this ensembling procedure is known as least-squares combination of ELMs (LSC-ELM).

Instead of using all the L models, the L^* networks (with $L^* \leq L$) whose linear combination provides better generalization capability are chosen. From the basis of the OP-ELM methodology, a direct and exact linear combination of the L^* chosen networks is obtained and, then, the useless networks are effectively discarded. Besides, in order to exploit diversity, this work analyzes three different alternatives to construct the L initial models: (1) to train all networks with the same initial input features; (2), to train all networks with a subset of input variables which is obtained by forward feature selection; and (3), to train each ELM network with

different random subsets of features. Each candidate network for the ensemble system is trained using the OP-ELM algorithm, as in [24]. On the contrary to other ensembling methods, this procedure has more diversity and independence in the individual ELM networks because they are trained with different hidden layer sizes and subsets of input features. As far as it is concerned, there is not any previous work that analyzes the influence of the input feature space on combining extreme learning machines. The selection of ELMs in the ensemble is fast and exact thanks to the advantages of the MRSR algorithm and the PRESS statistic, without influence to the random subsampling for obtaining a validation set.

A. Initial construction of ELM networks

The first stage is to construct L different SLFNs using the OP-ELM algorithm and three different alternatives are analyzed in order to exploit diversity:

1. The first approach is to consider that all SLFNs are trained using the n features that define the input dataset. In this case, the diversity is given by the different SLFNs (with different hidden layer sizes) obtained with OP-ELM for each random input weight initialization. It is known as OC-ELM (Optimal Combination of ELMs) and it is a direct enhancement of LSC-ELMs and OP-ELM.
2. Feature selection is a useful approach where the irrelevant inputs, which can be harmful for modeling the target data, are discarded. Following this, a second alternative is to obtain a subset of input variables through forward feature selection (without a previous ranking), in which features are sequentially added to an empty candidate set until the addition of further features does not decrease the LOO error in each individual model. Note that several repetitions (L) of the OP-ELM algorithm have to be done for averaging the LOO error in each possible feature subset. Once the convergence is reached, L different SLFNs are obtained (with different hidden

layer sizes), which have been trained using OP-ELM in the best subset of inputs. It is known as OC-ELM-FFS (Optimal Combination of ELMs based on Forward Feature Selection). This procedure gives the best subset of inputs but it entails a significant computational cost due to the incremental forward search.

3. In order to an ensemble achieves better accuracy than individual networks, it is critical that there should be enough independence or dissimilarity (i.e., diversity) among the L models. Thirdly, in order to increase the diversity, a good alternative is to consider different random subsets of features during the OP-ELM training of each network. Thus, it gives L different SLFNs which are composed of different number of hidden units and input features. It is known as OC-ELM-RFS (Optimal Combination of ELMs based on Random Feature Subsets). In this case, it may be required an enough large set of network candidates with random input subsets because inaccurate models can be obtained from certain feature subsets. Due to this, high-dimensional datasets require a larger number of random networks. It should be outlined that these inappropriate models (i.e., inappropriate random feature subsets) will be automatically omitted in the next pruning stage.

Figure 2 shows the linear combination scheme of L neural networks trained by OP-ELM. Note that the L networks have different hidden layer sizes and, in this approach, the feature input space is the same for all networks. The most appropriate models (with the same input feature space) are effectively chosen using OC-ELM. Whereas, when the OC-ELM-FFS approach is used, the input feature space is composed of the best d attributes, with $d < n$. Finally, in the case of the OC-ELM-RFS approach, the L models have different random input subsets.

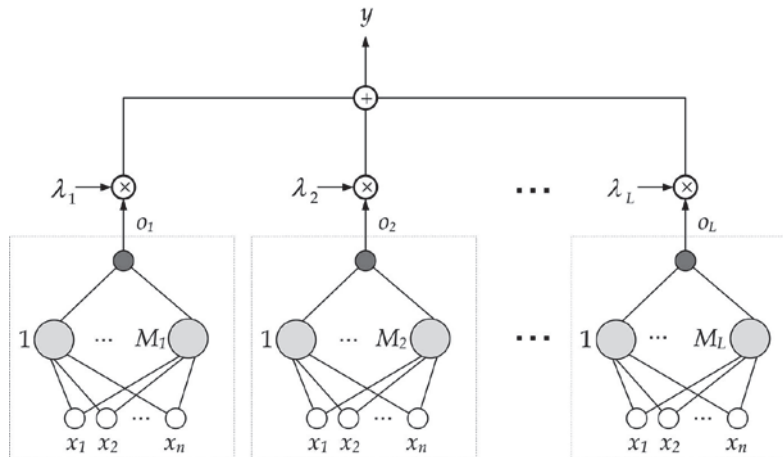


Figure 2. Linear combination of multiple extreme learning machines

B. Ensemble design

Once the L different ELMs have been trained, their network outputs (o_i) are previously ranked using the MRSR method [17] and the network output matrix \mathbf{O} is consequently ordered. Then, the better L^* models are chosen by minimizing the LOO error, which can be exactly measured with the PRESS statistic [20]. The inappropriate and useless models are discarded from the ensemble. This ensemble pruning reduces the storage needs, speeds up the operation stage and has the potential of improving the performance of the single neural architecture. It should be noted that the obtained ensemble system is fast and exact. Note that this stage is the same for the above three alternatives.

Finally, it should be noted that the model selection is exact and automatic. It is insensitive to random subsampling of validation sets (all samples are used for training and validation) and there is not any tuneable parameter.

4. Experiments

A. Datasets

In order to evaluate the performance of the ELM methods discussed in this work, four well-known regression problems are studied. Table 1 summarizes these datasets.

Table 1. Regression datasets used in the experiments

Dataset	Training Size	Testing Size	n
<i>Friedman</i>	27179	13589	10
<i>Stocks</i>	634	316	9
<i>Parkinsons</i>	3525	2350	16
<i>Tecator</i>	172	44	100

The first two datasets (*Friedman* and *Stocks*) are from L. Torgo's website [26], the *Parkinsons* dataset is obtained from UCI Machine Learning Repository [27] and the *Tecator* dataset is from the StatLib archive [28]. Except for *Tecator* (which is originally divided), each dataset is randomly splitted up into training (70%) and test sets (30%). These regression problems have been selected trying to cover different complexities, such as large datasets or high dimensional data. The first dataset, *Friedman*, is an artificial problem and the three remaining benchmark prediction problems belong to different realworld scientific areas (economics, biomedical engineering and chemometrics).

- *Friedman* is a synthetic benchmark dataset. The input vectors are composed of ten attributes which have been independently generated following uniform distribution between 0 and 1. The target variable is computed by

$$10\sin(x_1x_2) + 20\left(x_3 - \frac{1}{2}\right)^2 + 10x_4 + 5x_5 + \sigma \quad ,$$

where σ is a Gaussian white random noise with zero mean and unit variance. Then, the last five attributes are not relevant for predictions.

- *Stocks* data contain daily stock prices from January 1988 through October 1991 for ten aerospace companies. The task is to predict the price of the 10th company given the prices of the rest.
- *Parkinsons* dataset is composed of a range of biomedical voice measurements from 42 people with early-stage Parkinson's disease recruited to a sixmonth trial of a telemonitoring device for remote symptom progression monitoring. There are 5785 voice recordings and each input vector has sixteen biomedical voice measures. From these variables, the main aim is to predict the UPDRS score, which is a rating scale used to follow the longitudinal course of Parkinson's disease.
- *Tecator* database consists of 215 near-infrared absorbance spectra of meat samples. Each observation consists of a 100-channel absorbance spectrum in the wave length range 850-1050 nm. The task is to predict the fat content of a meat sample on the basis of its near infrared absorbance spectrum.

B. Experimental results

Initially, the standard ELM algorithm and the OP-ELM method have been used for training individual SLFNs by considering that the hidden layer size (M) is between 1 and 150. The learning and testing phases are repeated L times (L different weight initializations) in which the sigmoid function is selected as the activation function. In particular, $L = 50$ different repetitions have been done in all problems, except for *Tecator* because this highdimensional dataset requires more trials ($L = 100$) in order to OC-ELM-FFS obtains different random networks with enough input feature diversity, as it has been already mentioned in the previous section. After that, the ensemble based approaches are validated. In particular, the L trained OP-ELM networks are firstly combined by the least-squares solution of (8), i.e., the LSE-ELM method. Next, the three ensemble procedures (OC-ELMs, OC-ELM-FFS and OC-ELM-RFS) are evaluated. Note that the OC-ELM method makes use of the same previously L trained models with OP-ELM. In the experiments, all approaches use the LOO cross-validation technique, except for the standard ELM algorithm that uses 10-fold CV for selecting the hidden layer size. All simulations have been carried out in MATLAB 7.11(R2010b) environment running in the same machine with 4 GB of memory and 2.67 GHz processor.

Table 2 shows the obtained experimental results using the five aforementioned ELM-based approaches. In particular, for each method and dataset, this table shows the total computational training time, the test RMSE results from L trials and the model size. Note

that the RMSE of the ELM and OP-ELM methods are the best results of L trials and, meanwhile, the results of the remaining approaches are given by the different ensemble schemes of the L networks. In the last column of Table 2, the hidden layer sizes for the ELM and OP-ELM methods are shown and, meanwhile, the number of chosen networks is shown for the remaining ensemble-based approaches.

Table 2. Experimental results on four regression datasets

Data set	Learning Method	RMSE results	Training Time (s)	Model size
<i>Friedman</i>	ELM	2.67	7.87e+4	139
	OP-ELM	2.64	2.18e+3	99
	LSE-ELM	2.38	2.26e+3	50
	OC-ELM	2.38	2.20e+3	30
	OC-ELM-FFS	1.44	5.47e+4	16
<i>Stocks</i>	OC-ELM-RFS	1.28	2.02e+3	33
	ELM	1.11	2.46e+3	141
	OP-ELM	1.11	1.18e+2	101
	LSE-ELM	0.81	1.19e+2	50
	OC-ELM	0.78	1.19e+2	24
<i>Parkinsons</i>	OC-ELM-FFS	0.74	2.30e+3	16
	OC-ELM-RFS	0.72	1.03e+2	18
	ELM	9.02	1.07e+4	139
	OP-ELM	9.59	1.22e+3	85
	LSE-ELM	8.97	1.23e+3	50
<i>Tecator</i>	OC-ELM	8.95	1.23e+3	20
	OC-ELM-FFS	9.02	6.40e+4	9
	OC-ELM-RFS	8.87	1.16e+3	39
	ELM	4.88	8.25e+3	76
	OP-ELM	3.57	5.34e+2	67
	LSE-ELM	2.45	5.40e+2	100
	OC-ELM	2.28	5.42e+2	43
	OC-ELM-FFS	1.28	1.07e+4	15
	OC-ELM-RFS	2.04	5.28e+2	29

With respect to relation between the dataset size and the required training time, two datasets can be compared: *Tecator* is a high dimensional dataset with a reduced number of samples and, in contrast, *Friedman* is defined by a large sample set with fewer input features. As it can be seen from Table 2, the required training time with *Friedman* (a large sample dataset) is longer than with *Tecator* (a small sample dataset). Therefore, according to these results, it can be said that the number of training samples is more determined on training time than the number of input features.

As it can be observed from Table 2, OP-ELM outperforms the ELM method by using more compact networks and less training time, except for *Parkinsons* dataset. With respect to the ensemble-based approaches, all of them provide better generalization capabilities than ELM and OP-ELM in all problems.

It is worthy of remark that these advantages are obtained with a negligible increment of the total training time with OP-ELM, except for the OC-ELM-FFS procedure which is based on incremental forward feature selection and, thus, it needs more

computational efforts. In all simulations, the OC-ELM-FFS and OC-ELM-RFS provide better results than the ensemble procedures based on the original input feature space. From these results, the use of random feature subsets is beneficial to the ensemble construction. This approach may be better than performing a previous forward feature selection, except for *Tecator* dataset. For example, in *Tecator*, the input data space is successfully reduced from one hundred to only five features with OC-ELM-FFS. Whereas, in *Friedman* dataset, although OC-ELM-FFS chooses the relevant features (which are previously known by the definition of this artificial dataset), this method achieves worse performance results than OC-ELM-RFS, which uses random feature subsets (including different combinations of relevant and irrelevant features). For the *Parkinsons* data, OC-ELM-FFS entails a high computational increment but the RMSE results are not improved. In this case, OC-ELM-RFS provides the best solution in terms of prediction accuracy. Note that OC-ELM-RFS requires, as expected, more networks to construct the ensemble than OC-ELM-FFS in all problems.

Therefore, according to these experimental results, it is recommended to exploit diversity in the ELM ensemble system by varying the input data space with random subsets and, also, a feature selection is clearly useful for high-dimensional datasets, prior to ensemble construction.

5. Conclusions

This work presents effective procedures for linear combination of multiple neural networks based on the ELM algorithm. These approaches exploit the diversity among the different neural networks in the ensemble system in order to improve predictions. Useless models are discarded from the ensemble system using the basis of the OP-ELM methodology and, thus, an optimal combination of networks is achieved. ELM networks have been automatically constructed with different input feature spaces: the original input feature space, the input subset obtained by forward feature selection and the random feature subsets. The experimental results show that an ensemble approach of ELM models outperforms an individual ELM model in terms of generalization capability. The ensemble diversity is increased by varying the input feature space and it produces a performance improvement. It should be emphasized that the model selection is performed exactly and fastly without any tuneable parameter. As a future work, it is intended to explore the data editing techniques in order to generate different learning subsets for training the individual ELM networks. Other ongoing works are to check different ranking measures for model selection, such as the Akaike information criterion [29] and to use parallel and GPU programming for achieving training speed enhancement.

References

- [1] **S. Haykin.** *Neural Networks: A Comprehensive Foundation (2nd Edition)*. Prentice Hall, July 1998.
- [2] **L. K. Hansen, P. Salamon.** Neural network ensembles. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990, Vol. 12, No. 10, pp. 993-1001.
- [3] **M. F. Moller.** "Diversity in neural network ensembles," Ph.D. dissertation, University of Birmingham, 2003.
- [4] **A. Lipnickas.** Adaptive committees of neural classifiers. In: *Information Technology and Control*, 2008, Vol. 37, No. 3, pp. 205-210.
- [5] **S. Nabavi-Kerizi, M. Abadi, E. Kabir.** A PSO-based weighting method for linear combination of neural networks. In: *Computers & Electrical Engineering*, 2010, Vol. 36, No. 5, pp. 886-894.
- [6] **G. Huang, Q. Zhu, C. Siew.** Extreme learning machine: a new learning scheme of feedforward neural networks. In: *IEEE International Joint Conference on Neural Networks*, 2004, Vol. 2, pp. 985-990.
- [7] **G. Huang, Q. Zhu, C. Siew.** Extreme learning machine: Theory and applications. In: *Neurocomputing*, 2006, Vol. 70, No. 1-3, pp. 489-501.
- [8] **G.-B. Huang, D. H. Wang, Y. Lan.** Extreme learning machines: A survey. In: *International Journal of Machine Learning and Cybernetics*, 2011, Vol. 2, No. 2, pp. 107-122.
- [9] **P. J. García-Laencina, A. Bueno-Crespo, J.-L. Sancho-Gómez.** *Neurocomputing: Learning, Architectures and Modelings*. Nova Science, 2012, ch. 6. Design and Training of Neural Architectures using Extreme Learning Machines, pp. 119-145.
- [10] **Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, A. Lendasse.** OP-ELM: Optimally Pruned Extreme Learning Machine. In: *IEEE Transactions on Neural Networks*, 2010, Vol. 21, No. 1, pp. 158-162.
- [11] **C. R. Rao, S. K. Mitra.** *Generalized Inverse of Matrices and its Applications*. New York: Wiley, 1971.
- [12] **G.-B. Huang, L. Chen, C. K. Siew.** Universal approximation using incremental constructive feedforward networks with random hidden nodes. In: *IEEE Transactions on Neural Networks*, 2006, Vol. 17, No. 4, pp. 879-892.
- [13] **G.-B. Huang, M.-B. Li, L. Chen, C.-K. Siew.** Incremental extreme learning machine with fully complex hidden nodes. In: *Neurocomputing*, 2008, Vol. 71, pp. 576-583.
- [14] **G.-B. Huang L. Chen.** Enhanced random search based incremental extreme learning machine. In: *Neurocomputing*, 2008, Vol. 71, pp. 3460-3468.
- [15] **H.-J. Rong, Y.-S. Ong, A.-H. Tan, Z. Zhu.** A fast prunedextreme learning machine for classification problem. In: *Neurocomputing*, 2008, Vol. 72, No. 1-3, pp. 359-366.
- [16] **Y. Miche, P. Bas, C. Jutten, O. Simula, A. Lendasse.** A methodology for building regression models using extreme learning machine: OP-ELM. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, 2008, pp. 247-252.
- [17] **T. Similä, J. Tikka.** Multiresponse sparse regression with application to multidimensional scaling. In *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005*, ser. LNCS 3697, 2005, pp. 97-102.
- [18] **B. Efron, T. Hastie, L. Johnstone, R. Tibshirani.** Least angle regression. In: *Annals of Statistics*, 2004, Vol. 32, pp. 407-499.
- [19] **A. Bartoli.** On computing the prediction sum of squares statistic in linear least squares problems with multiple parameter or measurement sets. In: *International Journal on Computer Vision*, November 2009, Vol. 85, pp. 133-142.
- [20] **R. Myers.** *Classical and Modern Regression with Applications*, 2nd ed. Duxbury Press, 1990.
- [21] **Y. Lan, Y. Soh, G.-B. Huang.** Ensemble of online sequential extreme learning machine. In: *Neurocomputing*, 2009, Vol. 72, No. 13-15, pp. 3391-3395.
- [22] **N. Liu, H. Wang.** Ensemble based extreme learning machine. In: *IEEE Signal Processing Letters*, 2010, Vol. 17, No. 8, pp. 754-757.
- [23] **H. Chen, H. Chen, X. Nian, P. Liu.** Ensembling extreme learning machines. In: *4th International Symposium on Neural Networks, ISNN*, 2007, pp. 1069-1076.
- [24] **M. Van Heeswijk, Y. Miche, T. Lindh-Knuutila, P. Hilbers, T. Honkela, E. Oja, A. Lendasse.** Adaptive ensemble models of extreme learning machines for time series prediction. In: *International Conference on Artificial Neural Networks*, ser. LNCS, Vol. 5769, No. July, 2009, pp. 305-314.
- [25] **M. van Heeswijk, Y. Miche, E. Oja, A. Lendasse.** Solving large regression problems using an ensemble of GPU-accelerated ELMs. In: *ESANN2010: 18th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2010, pp. 309-314.
- [26] **L. Torgo.** Regression datasets. [Online]. Available at: <http://www.liaad.up.pt/ltorgo/Regression/DataSets.html>.
- [27] **A. Asuncion, D. Newman.** UCI machine learning repository. [Online]. Available at: <http://www.ics.uci.edu/~mllearn/MLRepository.html>.
- [28] **P. Vlachos.** Statlib archive. [Online]. Available at: <http://lib.stat.cmu.edu>.
- [29] **Z. Zhao, Y. Zhang, H. Liao.** Design of ensemble neural network using the akaike information criterion. In: *Engineering Applications of Artificial Intelligence*, Vol. 21, No. 8, 2008, pp. 1182-1188.

Received May 2012.