Prediction of Melting Points of Organic Compounds Using Extreme Learning Machines

Akshay U. Bhat,* Shamel S. Merchant, and Sunil S. Bhagwat

Chemical Engineering Department, Institute of Chemical Technology, University of Mumbai, Matunga, Mumbai 400019, India

Nonlinear regression methods such as artificial neural networks have been extensively used in prediction of properties of compounds from their molecular structure. Recently a new fast algorithm for training artificial neural networks known as the extreme learning machine was developed. In this paper we apply a simple ensemble of extreme learning machines to a large data set of melting points of organic molecules. The results obtained by extreme learning machines (cross-validated test set root-mean-square error = 45.4 K) are slightly better than those obtained using *k* nearest neighbor regression with genetic parameter optimization (cross-validated test set error = 46.2 K) and significantly better than those obtained by artificial neural networks trained using gradient descent (test set error = 49.3 K). The training of the extreme learning machine involves only linear regression resulting in faster training. Ensembling the extreme learning machines removes the dependence of results on initial random weights and improves the prediction. We also discuss the similarity between an ensemble of extreme learning machines and the random forest algorithm.

1. Introduction

Artificial neural networks have been extensively used in chemical engineering for control systems, prediction of properties, correlation of different parameters, etc.[1−4] However, traditional neural network training algorithms based on gradient descent optimization suffer from various drawbacks such as long training time, multiple local minima, high dependence on random initial weights, and the need for tuning of parameters such as learning rate and momentum. Recently Huang et al. invented a new fast algorithm for training artificial neural networks known as the extreme learning machine.[5,6] The extreme learning machine has been applied to the prediction of various data sets such as terrain reconstruction, classification of mental tasks, etc.[7,8] The benefits of this approach are that it has only one tunable parameter, namely, the number of neurons, and its training algorithm consists of only a single step.[5−8] Still, the extreme learning machine is found to be dependent on the random initial weights. To overcome this drawback, this article applies an ensemble of extreme learning machines. The ensembling procedure is to average results from multiple extreme learning machines trained on the same data set but each having different random initial weights.

The data set used is a quantitative structure−property relationship (QSPR) data set of melting points of 4173 molecules.[9,10] It has been earlier studied by Karthikeyan et al. using artificial neural networks and by Nigsch et al. using *k* nearest neighbor regression and genetic parameter optimization.[9,10]

Prediction of melting points has been studied extensively using both group contributions and quantitative structure−property relationship type methods. These studies have been motivated by the use of melting points to predict various physiochemical properties, some of them being in the areas of ionic liquids and drugs.[9−21] The melting point is difficult to model as it depends on several features such as crystal structure, packing in crystal, and inter- and intramolecular forces, which are not efficiently captured by molecular descriptors.[9,10]

The quantitative structure−property relationship methodology uses molecular structure in the form of graphs to calculate different parameters that describe the molecule. These parameters are known as molecular descriptors. The properties to be predicted are calculated as a function of molecular descriptors. The relationship between descriptors and molecular property may be nonlinear; hence nonlinear regression methods such as neural networks are employed to find the appropriate function.[9,10]

2. Extreme Learning Machine

The extreme learning machine was invented by Huang et al.[5,6] Its architecture is similar to that of a single-layer feed-forward neural network; the only difference is that there is no bias to output neuron. Every neuron in the input layer is connected to all neurons in the hidden layer. All hidden layer neurons are also provided with a bias. The activation function for the output neuron layer is linear, while that of the hidden neuron layer can be any piecewise continuous function. The extreme learning machine employs a completely different algorithm for calculating weights and biases, unlike the back-propagation or conjugate gradient descent training algorithm.

In the case of extreme learning machines, the weights and biases between the hidden layer and input layer neurons are randomly assigned. For an extreme learning machine having "*j*" hidden layer neurons trained on a data set having "*k*" training cases and "*i*" input neurons, the activation of all hidden layer neurons is calculated for every training case using the following formula.

$$H_{jk} = g(\sum(W_{ji}X_{ik}) + B_j)$$

where $g(\cdot)$ is any nonlinear piecewise continuous activation function, W_{ji} is the weight between the *i*th input neuron and *j*th hidden layer neuron, B_j represents the bias for the *j*th hidden layer neuron, X_{ik} is the input at the *i*th input neuron of the *k*th

* To whom correspondence should be addressed at 5, Sangam Road No. 1. Pandurang Wadi, Aarey Road, Goregoan East, Mumbai, India 400063. Tel.: 91 22 28740641. Fax: 91-22-2414, 5614. E-mail: akshayubhat@gmail.com.

10.1021/ie0704647 CCC: $40.75
Published on Web 01/10/2008

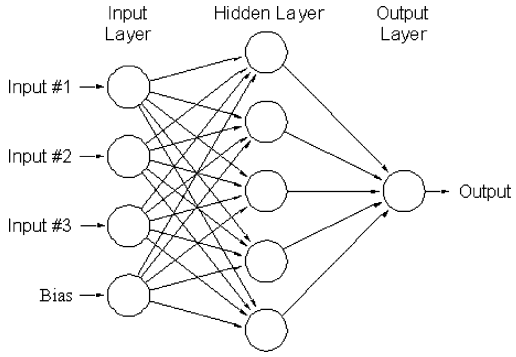

Figure 1. An extreme learning machine with five hidden layer neurons.

training case, and H_{jk} is the matrix containing activation of the jth hidden layer neuron for the kth training case.

The activation of all hidden layer neurons for all training samples is represented by a matrix **H**, which has k rows and j columns. The **H** matrix is called the hidden layer output matrix of the neural network.

The weights between hidden layer neurons and the output neuron are determined by performing a least-squares fit to the target values in the training set against the output of the hidden layer neurons for each training case. In mathematical notation, this is equivalent to solving the following linear system.

$$H_{k\times j}\beta_{j\times 1} = T_{k\times 1}$$

$$\boldsymbol{\beta} = (\beta_1 ... \beta_j)_{j\times 1}$$

where $\boldsymbol{\beta}$ is a vector containing weights between hidden layer neurons and the output layer neuron.

$$\mathbf{T} = (T_1 ... T_k)_{k\times 1}$$

where **T** is the vector containing targets for all training cases.

To obtain the weights, the above system is solved by multiplying the Moore−Penrose pseudoinverse of the **H** matrix with **T**, which is the same as performing least-squares multilinear regression.

$$\boldsymbol{\beta} = \mathbf{H}'\mathbf{T}$$

$\boldsymbol{\beta}$ = vector containing weights between hidden layer and output layer neurons, $\mathbf{H}'$ = Moore−Penrose pseudoinverse of matrix **H**, and **T** = vector containing targets for the training case. This completes the training of the network.

Thus training of an extreme learning machine involves only two steps: (1) random assignment of weights and biases to hidden layer neurons and calculation of the hidden layer output matrix **H**; (2) calculation of output weights using the Moore−Penrose pseudoinverse of matrix **H** using the values of targets for training cases.

The training process is fast as it involves finding the Moore−Penrose inverse of the hidden layer matrix, which is done much faster than normal epoch based training algorithms such as Levenberg−Marquardt; also the training only relies on a closed-form solution and does not involve any kind of nonlinear optimization routines. Consequently, the training time is significantly reduced and the only parameter left to be tuned is the number of hidden layer neurons.[5,6]

The extreme learning machine works by making use of a large number of random nonlinear projections of input space. Each neuron corresponds to a single projection. In the case of a conventional artificial neural network each of these projections is tuned. In the case of the extreme learning machine linear regression is performed on these projections; projections that match the curve get higher weights.[5,6] This way a line is fitted to the training points in the space of hidden layer neurons. It has been proved that, similar to single-layer feed-forward neural networks, extreme learning machines are also universal function approximators.[5,6]

3. Ensemble of Extreme Learning Machines

The extreme learning machine training algorithm described in section 2 indicates the importance of the randomly initialized hidden layer weights for model accuracy. Hence, to make results independent of random weights, we train multiple extreme learning machines on the same training set, with each having the same number of hidden layer neurons but different randomly assigned weights. Once trained separately, the final output for each case is calculated by averaging the outputs of each individual machine. This procedure is known as "ensembling", and the network is called as an "ensemble of extreme learning machines".

The only published reference on ensembles of extreme learning machines is by Chen et al.[22] It was published while this paper was being revised. They found that ensembling the extreme learning machine improved its performance on synthetic regression data sets.

The ensemble of extreme learning machines is depicted in Figure 2.

4. Data Set

The data set is from www.cheminformatics.org.[23] It contains melting points of 4173 nondrug molecules and 145 two-dimensional (2D) descriptors for each molecule. The data were originally collected from the Molecular Diversity Preservation International database,[24] and descriptors were calculated using Molecular Operating Environment software.[25]

The range of melting points is from 287.16 to 665.66 K. The molecular weights of the compounds varied from 84.1 to 815.63 g/mol. The range of experimental error in the measurement of these melting points is less than 5 K.[9,10] The descriptors used are 2D descriptors which are same as those used by Karthikeyan et al. and by Nigsch et al.[9,10] The 145 2D descriptors used encoded information about two-dimensional structures of the molecules.[9,10] By using the same descriptors as those used in earlier papers, we can be sure that the difference obtained in the results is only due to the difference in modeling technique.

5. Experimental Section

The 145 2D descriptors were standardized by setting the mean of each descriptor to 0 and variance to 1. The standardized descriptors were converted into their first 27 principal components that retained 99.45% of the variance observed. All the principal components were scaled to be between −1 and 1. Also, the values of melting point were scaled to be between −1 and 1. The activation function for hidden layer neurons was sigmoid, and that of the output neuron was linear without bias. Thus,

$$g(x) = 1/(1 + \exp(-x))$$

where $g(x)$ is the sigmoid activation function

The hidden layer was fully connected to the input layer. The weights of the hidden neurons were randomly generated between −1 and 1 from a uniform probability distribution, and biases were selected randomly in the range 0−1. The output weights

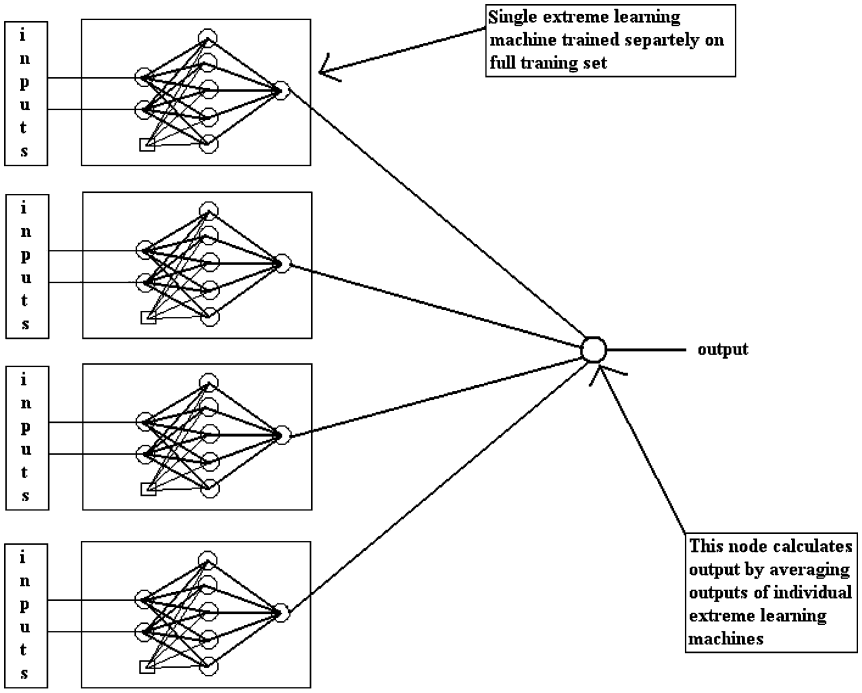

Figure 2. Ensemble of extreme learning machines containing four machines.

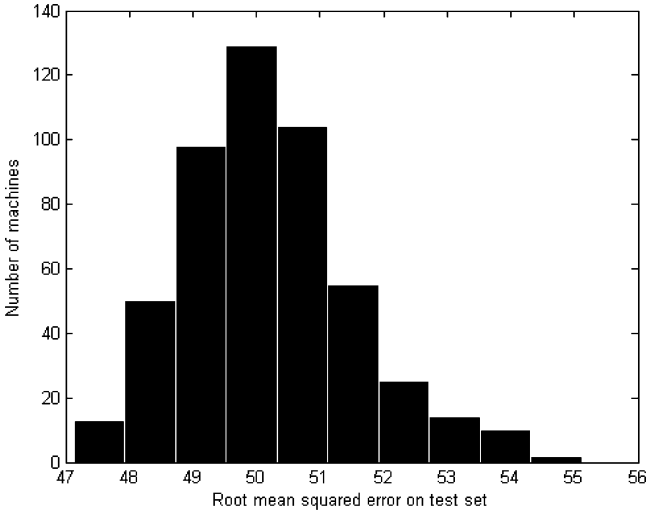

Figure 3. Histogram showing distribution of individual test set root-mean-square errors for 500 extreme learning machines.

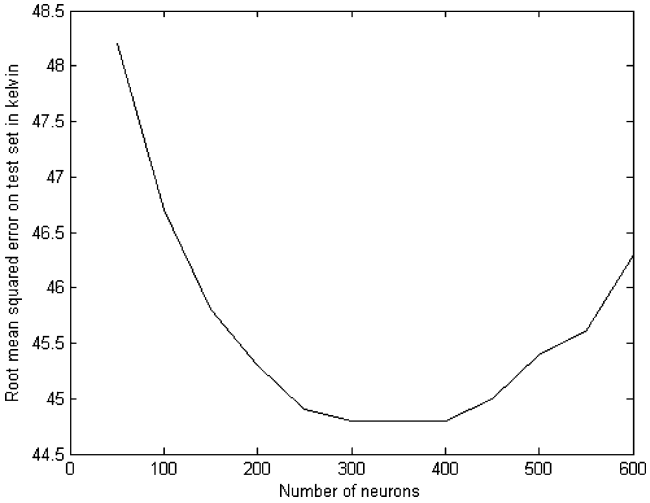

Figure 4. Variation in error on test set against number of neurons in hidden layer.

were determined using the Moore−Penrose generalized pseudo-inverse of the outputs of hidden layer neurons for the training set and respective values of scaled melting points.

An ensemble of 50 extreme learning machines was used, and outputs of all machines were averaged. All machines in the ensemble had the same number of hidden layer neurons but different random hidden layer weights and output layer weights. All computations were performed in the Matlab computing environment. The Matlab code for the extreme learning machine was taken from the Web site www.ntu.edu.sg/home/ebhuang.[26]

To measure the performance of the extreme learning machine, the root-mean-square error and average absolute error on a test set of molecules not used in developing the model were calculated. The test set was randomly chosen from the data set of 4173 molecules, and this procedure was performed multiple times to check the generalization capability of the model. Nigsch et al. applied the same procedure for the testing generalization capability of the model.[9]

5.1. Figure of Merit. We have used the following figure of merit. Let

$$N = \text{number of molecules in test set}$$

$$\text{root-mean-square error on test set} = \left(\sum(\text{predicted melting point} - \text{actual melting point})^2/N\right)^{1/2}$$

$$\text{cross-validated root-mean-square error} = \left(\sum(\text{root-mean-square error on test set for single trial})^2/\text{trials}\right)^{1/2}$$

N was 1000 and 10 trials were performed for selecting the optimum number of neurons.

6. Results

To compare between results obtained by a single extreme learning machine and an ensemble, 500 machines each having 300 hidden layer neurons were trained on a random test train

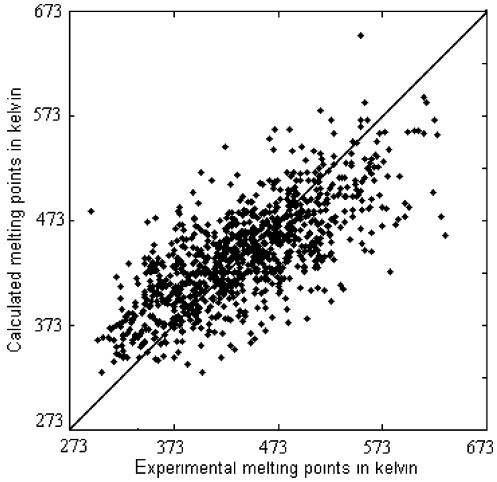

Figure 5. Parity plot showing experimental versus predicted melting points for 1000 molecules in test set.

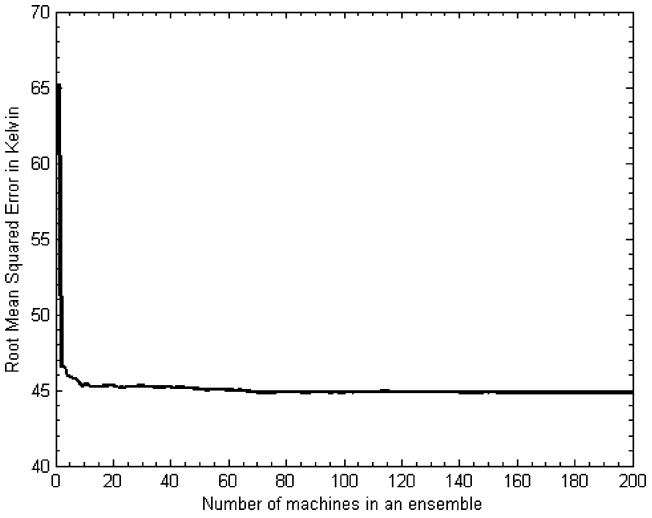

Figure 6. Variation of error of an ensemble with number of machines.

split. It was found that the error on the test set by a single extreme learning machine varied between 47 and 55 K. Figure 3 shows the histogram of the distribution of the root-mean-square error.

However, the root-mean-square error on the test set for the ensemble formed by the averaging machines was 44.8 K, which is significantly lower compared to the errors that can be achieved using a single machine. Also, the error obtained using the ensemble was independent of initial hidden layer weights.

6.1. Determining Number of Neurons. To determine the number of neurons, the data set was randomly split into 1000 (test):3173 (training) sets for 10 times and the error on the test set was used in determining the optimum number of neurons. An ensemble of 50 extreme learning machines was used. The number of neurons was varied from 50 to 600 in steps of 50 neurons. The test set error decreased until 300 neurons and remained nearly constant from 300 to 400 neurons; above 400 neurons the error on test set began to increase again, indicating ovefitting. Figure 4 shows the variation of test set error with number of neurons in hidden layer.

6.2. Final Model. The final model chosen was an ensemble of 50 extreme learning machines, each having 300 neurons in the hidden layer. To test the accuracy of the model, the data set was randomly split 25 times with 1000 molecules in the test set and the remaining in the training set. The root-mean-square error, correlation coefficient, and average absolute error were calculated for each trial on the training and test sets. The results are summarized in Table 1.

Figure 5 shows the parity plot between experimental and predicted values of the melting points of 1000 molecules in the test set.

The time required to train 50 extreme learning machines was 425 s on a Pentium 4, 3 GHz desktop computer with 512 MB of RAM.

7. Discussion

It was found that ensembling had a significant advantage over a single extreme learning machine. The error obtained was significantly lower compared to that for the single extreme learning machine (see Figure 6); also, the predictions were reproducible as they did not depend upon the random hidden layer weights.

The extreme learning machine required a larger number of neurons: 300 compared to 12 hidden layer neurons in the neural network model by Karthikeyan et al.[10] This can be explained on the basis of the number of free parameters tuned by the training algorithm. In the case of the extreme learning machine the weights between the hidden layer and the input layer are initialized randomly while those between the hidden layer and the output neuron are tuned using pseudoinversion; hence, the actual number of free parameters tuned is 300 in the case of the extreme learning machine. In the neural network model by Karthikeyan et al.,[10] all weights and biases were tuned using the conjugate gradient descent training algorithm; hence the total number of free parameters was 337, which is nearly the same as those tuned by the ensemble extreme learning machine.[10] Even though 300 neurons were used in the final model, a smaller number of neurons (around 250−300) can be used with only a negligible loss in generalization capability.

The generalization error is defined by statistical learning theory as probabilistic bounds on the sum of the training error and a term dependent on the VC (Vapnik−Chervonenkis) dimension of the machine.[27] The VC dimension of a machine is a function of the number of free parameters.[27] The VC dimension of the extreme learning machine has not been determined up to now. Hence we have used the number of free parameters in comparing the generalization performance of extreme learning machines and artificial neural networks.

When a new neuron is added, the number of free parameters increases by 1 in the case of extreme learning machines; however, in the case of artificial neural networks they increase by a number equal to the number of input neurons + number of output neurons + 1. Consequently, one can have better control on the number of free parameters tuned.

The Moore−Penrose pseudoinverse has only one solution for a given linear system; hence, there is no possibility of multiple local minima, which occur in the training of an artificial neural network by the gradient descent algorithm.[5,6]

Table 1. Averaged Results Obtained on 25 Random Test Train Splits of Data Set

	root-mean-square error (K)			absolute error (K)			squared correlation coefficient		
set	mean	min	max	mean	min	max	mean	min	max
training	39.7	39	40.1	30.7	30.2	31.2	0.62	0.61	0.63
test	45.4	43.9	47.6	35.1	33.6	37	0.5	0.45	0.55

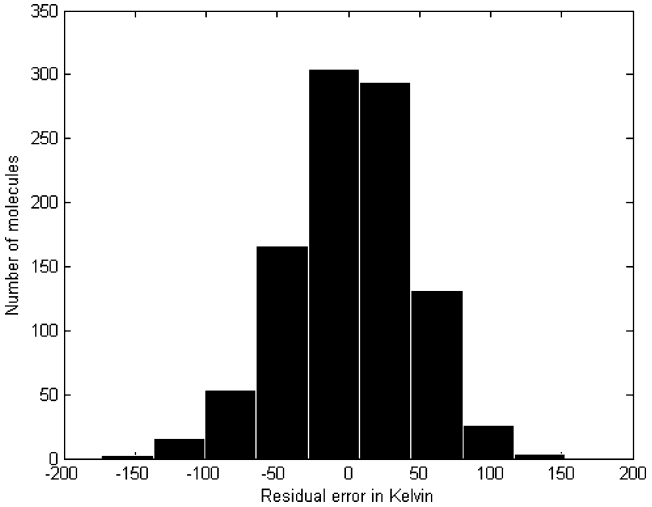

Figure 7. Histogram of residual for 1000 molecules in test set.

Due to a single step learning algorithm the training of the extreme learning machine is very fast compared to that of artificial neural networks trained using gradient descent algorithms. The time required to train a single extreme learning machine having 300 neurons was only 9 s using a Matlab implementation of the algorithm. Even higher speed can be achieved by using an algorithm coded in C++.[5,6]

The proposed ensembling procedure used was found to reduce the error further compared to a single extreme learning machine. This is mainly due to negative correlation between predictions of the individual extreme learning machines. For example, if the melting point of a molecule is predicted to be higher than actual by one machine and other machine predicts lower than actual, then by averaging the prediction of two machines we get close to the actual value. Hence, by using multiple machines, we are able to get to a better value than that given by a single machine. Also, it was found that the average value obtained was independent of hidden layer weights of individual machines, and even when a new set of machines were trained and predictions were averaged, the value turned out to be nearly the same.

7.1. Comparison with Earlier Studies. Comparing with an earlier study on the same data set by Nigsch et al., the error obtained by our model on the test set was slightly lower. Nigsch et al. predicted melting points using *k* nearest neighbor regression with genetic parameter optimization, and their model had a test set error of 46.2 K.[9] Karthikeyan et al. developed a single-layer feed-forward neural network model for the prediction of melting points on the same data set. Their model had a test set error of 49.3 K.[10] The model developed using extreme learning machines has a test set error of 45.4 K. Comparing the test set errors, it is found that the model developed using extreme learning machines has better predictability compared to earlier models while requiring manual tuning of the number of hidden layer neurons.

Most of the error in the prediction originates due to the inability of molecular descriptors to capture information about crystal structure and due to different possible polymorphs of molecules. The parity plot shows that molecules having lower melting points (323.16−373.16 K) are overpredicted while those having higher melting points (523.16−623.16 K) are underpredicted. Such a trend was also visible in earlier models developed by Karthikeyan al and Nigsch et al.[9,10] This trend is attributed to the data set, as it was found that all low melting point molecules were surrounded by higher melting point ones and higher melting point molecules were surrounded by the lower melting point molecules in descriptor space.[9,10] We believe that, to decrease the error further, additional molecular descriptors that encode information about intermolecular interactions and crystal structure must be developed. (See Figure 7.)

7.2. Similarity between Ensemble of Extreme Learning Machines and Random Forest. Random forest is a state-of-the-art method for nonlinear regression which uses randomly grown regression trees.[28] it has been used recently for QSPR modeling.[29−31] There exist many similarities between an ensemble of extreme learning machines and random forest. Similar to an ensemble of extreme learning machines, which uses the average of independent extreme learning machines for prediction, random forest uses a simple average of multiple independent regression trees for prediction. Each extreme learning machine is trained separately on the training set; similarly each tree is independently grown on a training set or a bootstrap sample from the training set. While growing a tree at each node, a randomly predetermined number of features are selected for splitting and growing the tree. Each tree is grown fully above the minimum number of nodes. This can be considered to be similar to the random assignment of hidden layer weights. The ensemble of extreme learning machines converges as the number of machines in an ensemble becomes large. Similarly, as the number of trees becomes large, the random forest converges.[28] It has been shown empirically that random forest performs better than other tree creation methods such as boosting, which is gradient descent over loss functions.[28] Similarly, the extreme learning machine has been shown to have better performance than artificial neural networks trained using gradient descent.[5,6] Hence, as one can see, there exist many similarities between the two algorithms and it would be very interesting in the future to compare both algorithms.[5,6,28]

8. Conclusion

A simple ensemble of extreme learning machines has been applied to the prediction of melting points of organic molecules. The ensemble of extreme learning machines has an advantage over earlier approaches by having a lower generalization error, only one tunable parameter, a small computation time, and a nondependence on random initial weights.

Acknowledgment

We thank Prof. Bhavik Bakshi for reviewing the manuscript and improving the quality of the paper. We thank Dr. Guangbin Huang for providing the code of the extreme learning machine and replying to our mail. We also thank Dr. Andreas Bender for replying to our e-mail, and we thank Dr. M. Karthikeyan, Florian Nigsch, and Dr. John B. O. Mitchell for providing the data set used in this paper. A.U.B. thanks the Dhirubhai Ambani Foundation and S.S.M. thanks the Ratan Tata Foundation for awarding a scholarship.

Literature Cited

(1) Jha, S. K.; Madras, G. Neural Network Modeling of Adsorption Equilibria of Mixtures in Supercritical Fluids. *Ind. Eng. Chem. Res.* **2005**, *44* (17), 7038−7041.

(2) Omata, K.; Yamada, M. Prediction of Effective Additives to a Ni/Active Carbon Catalyst for Vapor-Phase Carbonylation of Methanol by an Artificial Neural Network. *Ind. Eng. Chem. Res.* **2004**, *43* (20), 6622−6625.

(3) Pollock, G. S.; Eldridge, R. B. Neural Network Modeling of Structured Packing Height Equivalent to a Theoretical Plate. *Ind. Eng. Chem. Res.* **2000**, *39* (5), 1520−1525.

(4) Yeo, Y. K.; Kwon, T. I. A Neural PID Controller for the pH Neutralization Process. *Ind. Eng. Chem. Res.* **1999**, *38* (3), 978−987.

(5) Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'2004)*, Budapest, Hungary, July 25−29, 2004; Vol. 2, pp 985−990.

(6) Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme Learning Machine: Theory and Applications. *Neurocomputing* **2006**, *70*, 489−501.

(7) Yeu, C.-W. T.; Lim, M.-H.; Huang, G.-B.; Agarwal, A.; Ong, Y. S. A New Machine Learning Paradigm for Terrain Reconstruction. *IEEE Geosci. Remote Sens. Lett.* **2006**, *3*, 382−386.

(8) Wang, D.; Huang, G.-B. Protein Sequence Classification Using Extreme Learning Machine. *Proceedings of the International Joint Conference on Neural Networks (IJCNN'2005)*, Montreal, Canada, July 31−Aug 4, 2005.

(9) Nigsch, F.; Bender, A.; van Buuren, B.; Tissen, J.; Nigsch, E.; Mitchell, J. B. O. Melting Point Prediction Employing k-Nearest Neighbor Algorithms and Genetic Parameter Optimization. *J. Chem. Inf. Model.* **2006**, *46*, 2412−2422.

(10) Karthikeyan, M.; Glen, R. C.; Bender, A. General Melting Point Prediction Based on a Diverse compound Data Set and Artificial Neural Networks. *J. Chem. Inf. Model.* **2005**, *45*, 581−590.

(11) Abramowitz, R.; Yalkowsky, S. H. Melting Point, Boiling Point, and Symmetry. *Pharm. Res.* **1990**, *7*, 942−947.

(12) Dearden, J. C. Quantitative Structure-Property Relationships for Prediction of Boiling Point, Vapor Pressure, and Melting Point. *Environ. Toxicol. Chem.* **2003**, *22*, 1696−1709.

(13) Trohalaki, S.; Pachter, R. Prediction of Melting Points for Ionic Liquids. *QSAR Comb. Sci.* **2005**, *24*, 485−490.

(14) Trohalaki, S.; Pachter, R.; Drake, G. W.; Hawkins, T. Quantitative Structure-Property Relationships for Melting Points and Densities of Ionic Liquids. *Energy Fuels* **2005**, *19*, 279−284.

(15) Jain, A.; Yang, G.; Yalkowsky, S. H. Estimation of Melting Points of Organic Compounds. *Ind. Eng. Chem. Res.* **2004**, *43*, 7618−7621.

(16) Bergström, C. A. S.; Norinder, U.; Luthman, K.; Artursson, P. Molecular Descriptors Influencing Melting Point and Their Role in Classification of Solid Drugs. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1177−1185.

(17) Katritzky, A. R.; Jain, R.; Lomaka, A.; Petrukhin, R.; Maran, U.; Karelson, M. Perspective on the Relationship between Melting Points and Chemical Structure. *Cryst. Growth Des.* **2001**, *1*, 261−265.

(18) Zhao, L. W.; Yalkowsky, S. H. A Combined Group Contribution and Molecular Geometry Approach for Predicting Melting Points of Aliphatic Compounds. *Ind. Eng. Chem. Res.* **1999**, *38*, 3581−3584.

(19) Johnson, J. L. H.; Yalkowsky, S. H. Two New Parameters for Predicting the Entropy of Melting: Eccentricity (epsilon) and Spirality (mu). *Ind. Eng. Chem. Res.* **2005**, *44*, 7559−7566.

(20) Dannenfelser, R. M.; Yalkowsky, S. H. Predicting the Total Entropy of Melting: Application to Pharmaceuticals and Environmentally Relevant Compounds. *J. Pharm. Sci.* **1999**, *88*, 722−724.

(21) Modarressi, H.; Dearden, J. C.; Modarress, I. QSPR Correlation of Melting Point for Drug Compounds Based on Different Sources of Molecular Descriptors. *J. Chem. Inf. Model.* **2006**, *46*, 930−936.

(22) Chen, H.; Chen, H.; Nian, X.; Liu, P. Ensembling Extreme Learning Machines. *Advances in Neural Networks−ISNN*; Lecture Notes in Computer Science 4491; Springer: Berlin, 2007; pp 1069−1076.

(23) http://www.cheminformatics.org/ (accessed March 20, 2006).

(24) Molecular Diversity Preservation International (MDPI). http://www.mdpi.org/ (accessed March 20, 2006).

(25) MOE (Molecular Operating Environment); Chemical Computing Group, Inc.: Montreal, Quebec, Canada.

(26) www.ntu.edu.sg/home/ebhuang (accessed March 20, 2006).

(27) Vapnik, V. N. *The Nature of Statistical Learning Theory*; Springer-Verlag: Berlin, 1995.

(28) Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5−32.

(29) Palmer, D. S.; O'Boyle, N. M.; Glen, R. C.; Mitchell, J. B. O. Random Forest Models To Predict aqueous Solubility. *J. Chem. Inf. Model.* **2007**, *47* (1), 150−158.

(30) Ehrman, T. M.; Barlow, D. J.; Hylands, P. J. Virtual Screening of Chinese Herbs with Random Forest. *J. Chem. Inf. Model.* **2007**, *47* (2), 264−278.

(31) Zhang, Q. Y.; Aires-de-Sousa, J. Random Forest Prediction of Mutagenicity from Empirical Physicochemical Descriptors. *J. Chem. Inf. Model.* **2007**, *47* (1), 1−8.

Received for review March 31, 2007
Revised manuscript received October 11, 2007
Accepted October 30, 2007

IE0704647