Extreme Learning Machines for Multiclass Classification: Refining Predictions with Gaussian Mixture Models

Emil Eirola[1], Andrey Gritsenko[2], Anton Akusok[1,2], Kaj-Mikael Björk[1], Yoan Miche[3,4], Dušan Sovilj[3], Rui Nian[5], Bo He[5], and Amaury Lendasse[1,2]

[1] Arcada University of Applied Sciences, Helsinki, Finland
emil.eirola@arcada.fi
amaury-lendasse@uiowa.edu
[2] Department of Mechanical and Industrial Engineering and the Iowa Informatics Initiative, The University of Iowa, Iowa City, USA
[3] Department of Information and Computer Science, Aalto University School of Science, FI–00076, Finland
[4] Nokia Solutions and Networks Group, Espoo, Finland
[5] College of Information Science and Engineering, Ocean University of China, 266003 Qingdao, China

Abstract. This paper presents an extension of the well-known Extreme Learning Machines (ELMs). The main goal is to provide probabilities as outputs for Multiclass Classification problems. Such information is more useful in practice than traditional crisp classification outputs. In summary, Gaussian Mixture Models are used as post-processing of ELMs. In that context, the proposed global methodology is keeping the advantages of ELMs (low computational time and state of the art performances) and the ability of Gaussian Mixture Models to deal with probabilities. The methodology is tested on 3 toy examples and 3 real datasets. As a result, the global performances of ELMs are slightly improved and the probability outputs are seen to be accurate and useful in practice.

Keywords: Classification, Machine Learning, Neural Network, Extreme Learning Machines, Gaussian Mixture Models, Multiclass Classification, Leave-one-out Cross-Validation, PRESS Statistics, Parental Control, Internet Security.

1 Introduction

The Extreme Learning Machines and other neural networks have a successful history of being used to solve classification problems. The standard procedure is to convert the class labels into numerical 0/1 binary variables (or equivalently, $+1/-1$), effectively transforming the situation into a regression task. When a new sample is fed through the network to produce a result, the class is assigned based on which numerical value it is closest to. While this leads to good performance in terms of classification accuracy and precision, the network outputs as such are

not very meaningful. This paper presents a method which converts the outputs into more interpretable probabilities by using Gaussian Mixture Models (GMM).

Most classifiers based on neural networks provide results which can not directly be interpreted as probabilities. Probabilities are useful for understanding the confidence in classification, and evaluating the possibility of misclassification. In a multiclass problem, for instance, certain misclassification results may be considerably more harmful or expensive than others.

One example is in website filtering based on user-defined categories, where neural networks are used to classify previously uncategorized sites [1,2]. More reliable estimates of the risks involved are necessary for cloud security service provider to make informed (but automated) filtering decisions. Other such cases where the penalty for choosing the wrong class may vary greatly, include detecting malicious software activity [3,4,5], bankruptcy prediction [6] and nuclear accident prediction [7].

It is true that the optimal least-squares estimator is equivalent to the conditional probability:

$$\hat{y}(x) = \mathrm{E}[Y \mid x] = p(Y = 1 \mid x).$$

In practice, however, the results can be outside the range 0–1, and this interpretation is not very easy or useful.

Gaussian Mixture Models can be used to transform the values in the output layer to more interpretable probabilities. Specifically, this is accomplished by fitting the model to the training data and using it to calculate the probability of a sample belonging to a class, conditional on the output of the ELM. This procedure of refining the classification result of the ELM also leads to better classification accuracy and precision in some cases, as illustrated in the Experiments (section 3.2).

In related work, the Sparse Bayesian Extreme Learning Machine [8] presents another approach to use an ELM and obtain estimates of the posterior probability for each class. In the SBELM, the parameters of the ELM and the Bayesian inference mechanism are linked, and must be learned together through an iterative optimization scheme. This contrasts the currently proposed method, where the ELM and GMM layers are entirely decoupled, and can be trained separately.

The remainder of this paper is structured as follows: Section 2 reviews the Extreme Learning Machines and Gaussian Mixture Models before introducing two variants of the proposed refinement procedure. An experimental comparison on a variety of datasets is provided in Section 3. Section 4 presents conclusions and further works.

2 Global Methodologies

2.1 Extreme Learning Machines

Extreme Learning Machines (ELMs) [9] are single hidden-layer feed-forward neural networks where *only* the output weights are optimised, and all the weights between the input and hidden layer are assigned randomly (see Figure 1). Due to

its fast computational speed and theoretical guarantees [10], the method recently received an active development both theoretically [11,12,13], including optimally pruned modification of ELM [14,15], and in applications [16], in particular: finding mislabeled samples using ELM [17], ELM for time series prediction [18,19], identification of evolving fuzzy systems using OP-ELM [20], accelerating ELM using GPU [21], ELM for regression with missing data [22], solving feature selection problem using ELM [23], ELM for nominal data classification [24], etc.

Training this model is simple, as the optimal output weights $\boldsymbol{\beta}$ can be calculated by ordinary least squares or various regularised alternatives.

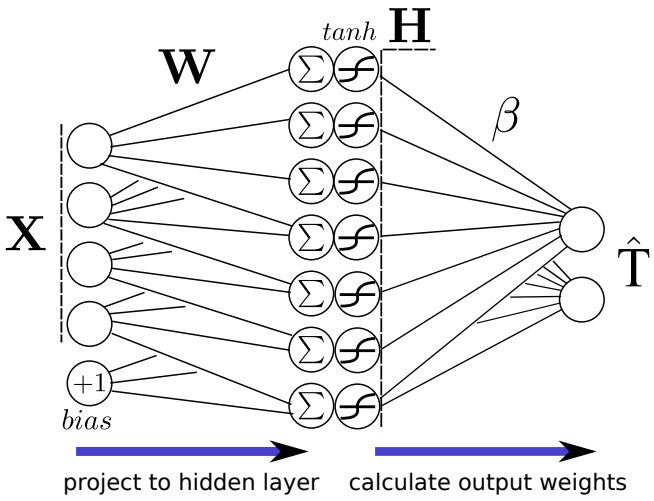

Fig. 1. Extreme learning machine with multiple outputs. Bias is conveniently included as an additional constant +1 input. Hidden layer weights $\mathbf{W}$ are fixed, only output layer weights $\boldsymbol{\beta}$ are calculated.

In the following, a multi-class classification task is assumed. The data is a set of N distinct samples $\{\boldsymbol{x}_i, y_i\}$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, \ldots, c\}$ where c is the number of distinct classes. Encode classification targets as one binary variable for each class (one-hot encoding). $\mathbf{T}$ is the matrix of targets such that $T_{ij} = 1$ if and only if $y_i = j$, i.e., sample i belongs to class j. Otherwise, $T_{ij} = 0$. In the case of two classes, a single output variable is sufficient.

A single (hidden) layer feedforward neural network (SLFN) with d input nodes, c output nodes, and M neurons in the hidden layer can be written as

$$f(\boldsymbol{x}) = \sum_{k=1}^{M} \boldsymbol{\beta}_k h\left(\boldsymbol{w}_k \cdot \boldsymbol{x}\right), \tag{1}$$

where $\boldsymbol{w}_k$ are randomly assigned d-dimensional weight vectors, the output layer weights $\boldsymbol{\beta}_k$ are c-dimensional vectors, and $h(\cdot)$ an appropriate nonlinear acti-

vation function, e.g., the sigmoid function. The output of f is a c-dimensional vector, and class assignment is determined by which component is the largest.

In terms of matrices, the training of the network can be re-written as finding the least-squares solution to the matrix equation.

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}, \quad \text{where} \quad H_{ik} = h\big(\boldsymbol{w}_k \cdot \boldsymbol{x}_i\big). \tag{2}$$

Constant bias terms are commonly included by appending a 1 to each $\boldsymbol{x}_i$ and concatenating a column of 1s to $\mathbf{H}$.

2.2 PRESS Statistics for Selecting the Optimal Number of Neurons

The number of hidden neurons is the only tunable hyperparameter in an ELM model. It is selected using a Leave-One-Out (LOO) Cross-Validation error. The LOO method is usually a costly approach to optimize a parameter since it requires to train the model on the whole dataset but one sample, and evaluate on this sample repeatedly for all the samples of the dataset. However, the output layer is linear for the ELM model, and the LOO error has a closed form given by Allen's Prediction Sum of Squares (PRESS) [25]. This closed form allows for fast computation of the LOO Mean Square Error, which gives an estimate of the generalization error of ELM. The optimal number of hidden neurons is found as the minimum of that Meas Squared Error.

The Allen's PRESS formula written with the multi-output notations of the paper is

$$\mathrm{MSE}_{\mathrm{LOO}}^{\mathrm{PRESS}} = \frac{1}{Nc} \sum_{n=1}^{N} \sum_{k=1}^{c} \left(\frac{\mathbf{T} - \mathbf{H}\mathbf{H}^{\dagger}\mathbf{T}}{\left[\mathbf{1}_N - \mathrm{diag}\left(\mathbf{H}\mathbf{H}^{\dagger}\right)\right] \mathbf{1}_c^T} \right)_{ik}^2, \tag{3}$$

where $\mathbf{H}^{\dagger}$ denotes the Moore-Penrose pseudo-inverse [26] of $\mathbf{H}$, and the division and square operations are applied element-wise.

2.3 Gaussian Mixture Models

Mixtures of Gaussians can be used for a variety of applications by estimating the density of data samples [27,28]. A Gaussian Mixture Model can approximate any distribution by fitting a number of components, each representing a multivariate normal distribution. See Figure 2 as an example.

The model is defined by its parameters, which consist of the mixing coefficients π_k, the means $\boldsymbol{\mu}_k$, and covariance matrices $\boldsymbol{\Sigma}_k$ for each component k ($1 \leq k \leq K$) in a mixture of K components. The combination of parameters is represented as $\boldsymbol{\theta} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

The model specifies a distribution in $\mathbb{R}^d$, given by the probability density function

$$p(\boldsymbol{x} \,|\, \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{4}$$

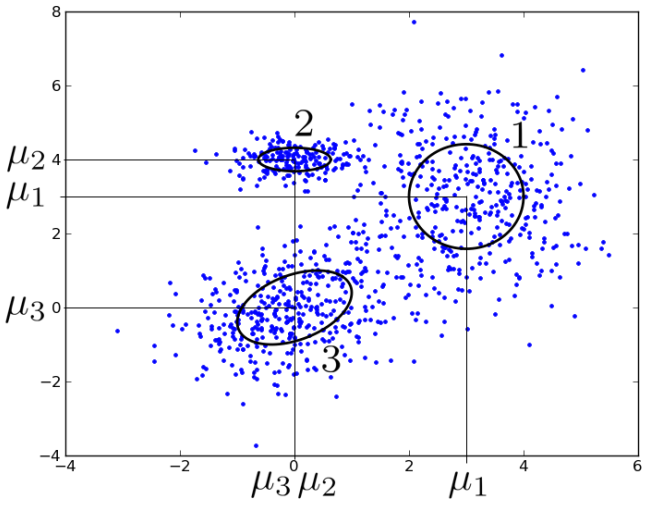

Fig. 2. An example of 2D data with 3 Gaussian components after the convergence of GMM.

where $\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the probability density function of the multivariate normal distribution

$$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right). \tag{5}$$

The standard procedure for fitting a Gaussian Mixture Model to a dataset is maximum likelihood estimation by the Expectation-Maximisation (EM) algorithm [29,30,28]. The E-step and M-step are alternated until convergence is observed in the log-likelihood. Initialisation before the first E-step is arbitrary, but a common choice is to use the clustering algorithm K-means to find a reasonable initialisation [27].

The only parameter to tune select is the number of components K. This can be done by separately fitting several models with different values for K, and using the BIC criterion [31] to select the best model. In the proposed methodology, we are using the BIC criterion to select the value of K. Several further criteria are discussed in [32, Ch. 6].

2.4 ELM-GMM

The main idea of the proposed method is to first train a standard ELM for classification, and then use a GMM to refine the results into more interpretable probabilities. This is accomplished by building a separate GMM for each class, on the ELM outputs of the samples from that class. If Y is the output of the

ELM, the GMM is a model for the conditional distribution $p(Y \mid C)$ for each class. This leads to c separate GMMs.

Given a new sample, prediction is conducted as follows: calculate the ELM output Y, and apply Bayes' theorem to find the posterior probability of each class C:

$$p(C \mid Y) = p(Y \mid C)\frac{p(C)}{p(Y)}.$$

Specifically:

$$p(C \mid Y) \propto p(Y \mid C)p(C),$$

where the proportionality constant is determined by the condition of adding up to 1. The class priors $p(C)$ are given by the proportions in the training set (i.e., the maximum likelihood estimate).

The end result is now interpretable as a probability. A summary of the training and testing algorithms is presented in Algorithm 1.

Algorithm 1 Training the model and finding the conditional class probabilities for unseen data.

▷ **Training step**

Require: Input data $\mathbf{X}$, targets $\mathbf{T}$

1: Randomly assign input vectors $\boldsymbol{w}_k$ and form $\mathbf{H}$
2: Calculate $\boldsymbol{\beta}$ as the least squares solution to eq. (2)
3: Calculate outputs on training data: $\mathbf{Y} = \mathbf{H}\boldsymbol{\beta}$
4: **For each** class C **do**
5: Fit a GMM_C to the rows of $\mathbf{Y}$ corresponding to the class C
6: **End for**
7: Calculate $p(C)$ based on proportions of each class
8: **Return** $\boldsymbol{w}_k$, $\boldsymbol{\beta}$, GMM_C, $p(C)$

▷ **Testing step**

Require: Test data $\mathbf{X}_t$, weights $\boldsymbol{w}_k$, $\boldsymbol{\beta}$, GMM_C and $p(C)$ for each class C

1: Form $\mathbf{H}_t$ by using the weights $\boldsymbol{w}_k$
2: Calculate outputs: $\mathbf{Y}_t = \mathbf{H}_t\boldsymbol{\beta}$
3: **For each** class C **do**
4: Use GMM_C to calculate $p(Y_t \mid C)$ for each sample
5: **End for**
6: Calculate $p(C \mid Y_t) \propto p(Y_t \mid C)p(C)$ for each sample
7: **Return** Conditional probabilities $p(C \mid Y_t)$ for each class for each sample

To evaluate performances of this model for each sample, we consider the class with the highest conditional probability as the result of classification. A second criterion is presented and used in the Experiments Section 3.4 in order to evaluate the quality and the applicability of the predicted probabilities.

2.5 Refine the Training for GMM

It is obvious that GMM built of the ELM outputs would inherit the error of the ELM model. To avoid this error accumulation, we are proposing to build GMM using only the correct classifications of the ELM. This training approach will be denoted by suffix 'r' added to the corresponding GMMs. Compared to the algorithm presented in Algorithm 1, the only change is an additional step between steps 3 and 4 of the training phase: delete the rows of $\mathbf{Y}$ corresponding to misclassified samples. In the Experiments Section 3.4, it is shown that this second approach is especially relevant when the original multiclass classification task is challenging.

3 Experiments

In the following subsections, three methodologies are compared using several classification tasks. These compared methods are the original ELM, and the two variants of the proposed combination of ELM and GMM: ELM-GMM and ELM-GMMr.

3.1 Datasets

Six different datasets have been chosen for the experiments: three small datasets and three large ones. Datasets are collected from the University of California at Irvine (UCI) Machine Learning Repository [33] and they have been chosen by the overall heterogeneity in terms of number of samples, variables, and classes for classification problems. Furthermore, the large datasets have high number of variables and a large number of classes. This is done in order to validate the quality of the predicted probabilities.

Table 1. Information about the selected datasets

			Samples	
Dataset	Variables	Classes	Train	Test
Wisconsin Breast Cancer	30	2	379	190
Pima Indians Diabetes	8	2	512	256
Wine	13	3	118	60
Image Segmentation	18	7	1540	770
First-Order Theorem Proving	51	6	4078	2040
Cardiotocography	21	10	1417	709

Table 1 summarizes the different attributes for the six datasets. All datasets have been preprocessed in the same way. Two thirds of the points are used to create the training set and the remaining third is used as the test set. The First-Order Theorem Proving dataset has predefined training, validation and testing

sets in proportion of 2:1:1. We have have performed random permutation for the validation set. Afterwards, the result of permutation for the validation set was split in two parts to be added to the test and train sets in such a way that the resulting ratio between these sets becomes 2:1. Then for all datasets, the training set is standardized to zero mean and unit variance, and the test set is also standardized using the same mean and variance calculated and used for the training set. Because the test set is standardized using the same parameters as for the training set, it is most likely not exactly zero mean and unit variance.

It should also be noted that the proportions of the classes have been kept balanced: each class is represented in an equal proportion, in both training and test sets. This is important in order to have relevant test results.

3.2 Experimental Procedure

All experiments have been run on the same Windows machine with 16 GB of memory (no swapping for any of the experiments) and 3.6 GHz processor, single-threaded execution on one single core, for the sake of comparisons.

Because ELM is a single hidden-layer feed-forward neural network with randomly assigned weights $\boldsymbol{w}_k$, we run each method 1000 times and average its performance. We also compute the optimal value of neurons for ELM on each step using the PRESS Leave-One-Out Cross-Validation technique [25,34] with a maximum number of neurons equal to 300 based on the performance results obtained by [11].

3.3 Results

Table 2 shows the test results for the three models and six datasets. In this Table 2, each GMM is built of the ELM outputs for a certain dataset. In that table, we have removed ELMs from the names of the global methodologies for the sake of clarity.

Comparing the accuracies of ELMs to the ones of the GMM variants, some datasets (Wine, Cardiotocography) are showing that the GMM is providing a clear improvement. In the other cases, the results are not notably different, but never statistically worse. The First-Order Theorem Proving dataset is the only situation where ELM-GMM performs clearly worse, but ELM-GMMr is again better than the original ELM. For all datasets, ELM-GMMr provides similar or better results than ELM-GMM.

3.4 Reevaluate Performance of Probability Classification Methods

When calculating the performance of a probability-based classification method by just picking the class with the highest probability and treating it as a result of classification, we lose the advantage of the probability itself.

There are several possible solutions to take into account the predicted probabilities. One of the most simple solutions is to consider a classification to be

Table 2. Correct classification rates (and standard deviation in brackets) for all six datasets obtained using 3 different methods. "Wisc. B.C." for Wisconsin Breast Cancer dataset, "Pima I.D." for Pima Indians Diabetes dataset, "Image Seg." for Image Segmentation dataset, "F.-O. T.P." for First-Order Theorem Proving dataset and "Card." for Cardiotocography dataset

	Wisc. B.C.	Pima I.D.	Wine	Image Seg.	F.-O. T.P.	Card.
ELM	95.05 (1.49)	70.90 (1.69)	93.00 (2.92)	93.67 (0.66)	52.34 (0.77)	73.63 (1.34)
GMM	95.00 (1.84)	70.40 (2.22)	94.00 (3.16)	93.96 (0.68)	50.42 (0.90)	76.62 (1.27)
GMMr	95.00 (1.49)	70.98 (1.51)	96.67 (2.81)	93.92 (0.64)	52.45 (0.75)	76.59 (1.29)

correct if one of the two highest probabilities is for the correct class. If the predicted probabilities were not meaningful, the increase of performance measured by this second criterion would be limited. For example, in the Cardiotocography dataset with a total of 10 classes the improvement is close to 15%. The standard deviation is decreased. The correct class is nearly certainly one of the two most probable predicted classes. Eight classes are then certainly discarded. Similar considerations can be made for the First-Order Theorem Proving dataset, for which the improvement is even more significant, and the other examples.

Table 3 shows the resulting improvement in accuracy for those datasets with more than two classes. This second criterion is imperfect, and will be replaced in further works. For example, probabilistic classification will be investigated in order to provide a probability distribution of the performances of the proposed methodologies.

Table 3. Comparing the improvement in classification accuracy when considering top 2 labels. "Image Seg." for Image Segmentation dataset, "F.-O. T.P." for First-Order Theorem Proving dataset and "Card." for Cardiotocography dataset

	Wine	Image Seg.	F.-O. T.P.	Card.
GMM	94.00 (3.16)	93.96 (0.68)	50.42 (0.90)	76.62 (1.27)
GMM 2	99.50 (0.81)	97.89 (0.43)	68.48 (0.85)	91.42 (0.86)
GMMr	96.67 (2.81)	93.92 (0.64)	52.45 (0.75)	76.59 (1.29)
GMMr 2	99.50 (0.81)	97.83 (0.39)	68.79 (0.89)	91.03 (0.91)

4 Conclusions and Further Works

The proposed methodology is based on the well-known ELMs that has been shown to provide accurate classification results.

Including GMM as postprocessing preserves the qualities of ELMs. Based on the results obtained on six datasets, it has been shown that the provided predicted probabilities are accurate, useful and robust.

The drawback of the given methodology is an increase of the overfitting risk based on the fact that both ELM and GMM are trained on the same training sets. Furthermore, the optimal number of neurons for the original ELM is probably not optimal when the GMMs are added. In the future, selecting the optimal number of neuron for the proposed global methodology will be rigorously investigated.

Comparison with Sparse Bayesian Extreme Learning Machines [8] will also be done in the future, and computational times will be compared.

As described in the Experiments Section, there are needs to develop a better criterion to evaluate the quality and the advantages of dealing with probability outputs.

In the future, the proposed methodology will be tested on very large datasets, including more than one million samples, several hundreds of input variables and ten to twenty classes. For example, to perform website classification [35] where the number of given output classes is very large, and the number of samples is nearly unlimited.

References

1. Qi, X., Davison, B.D.: Web page classification: Features and algorithms. ACM Comput. Surv. **41**(2) (February 2009) 12:1–12:31
2. Patil, A.S., Pawar, B.: Automated classification of web sites using naive bayesian algorithm. In: Proceedings of the International MultiConference of Engineers and Computer Scientists. Volume 1. (2012)
3. Dahl, G., Stokes, J.W., Deng, L., Yu, D.: Large-scale malware classification using random projections and neural networks. In: Proceedings IEEE Conference on Acoustics, Speech, and Signal Processing, IEEE SPS (May 2013)
4. Rieck, K., Trinius, P., Willems, C., Holz, T.: Automatic analysis of malware behavior using machine learning. J. Comput. Secur. **19**(4) (December 2011) 639–668
5. Miche, Y., Akusok, A., Hegedus, J., Nian, R.: A Two-Stage Methodology using K-NN and False Positive Minimizing ELM for Nominal Data Classification. Cognitive Computation (2014) 1–26
6. Akusok, A., Veganzones, D., Björk, K.M., Séverin, E., du Jardin, P., Lendasse, A., Miche, Y.: ELM Clustering–Application to Bankruptcy Prediction–. In: International work conference on TIme SEries. (2014) 711–723
7. Sirola, M., Talonen, J., Lampi, G.: SOM based methods in early fault detection of nuclear industry. In: ESANN. (2009)
8. Luo, J., Vong, C.M., Wong, P.K.: Sparse bayesian extreme learning machine for multi-classification. Neural Networks and Learning Systems, IEEE Transactions on **25**(4) (2014) 836–843
9. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing **70**(1–3) (2006) 489–501
10. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Transactions on Neural Networks **17**(4) (2006) 879–892
11. Miche, Y., van Heeswijk, M., Bas, P., Simula, O., Lendasse, A.: TROP-ELM: A double-regularized ELM using LARS and Tikhonov regularization. Neurocomputing **74**(16) (September 2011) 2413–2421

12. Lendasse, A., Akusok, A., Simula, O., Corona, F., van Heeswijk, M., Eirola, E., Miche, Y.: Extreme Learning Machine: A Robust Modeling Technique? Yes! In: Proc. of Advances in Computational Intelligence - 12th International Work-Conference on Artificial Neural Networks, IWANN 2013, Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, Proceedings, Part I. (2013) 17–35
13. Yu, Q., van Heeswijk, M., Miche, Y., Nian, R., He, B., Séverin, E., Lendasse, A.: Ensemble delta test-extreme learning machine (dt-elm) for regression. Neurocomputing **129** (2014) 153–158 cited By 2.
14. Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: Op-elm: Optimally pruned extreme learning machine. Neural Networks, IEEE Transactions on **21**(1) (Jan 2010) 158–162
15. Miche, Y., Sorjamaa, A., Lendasse, A.: Op-elm: Theory, experiments and a toolbox. In Kůrková, V., Neruda, R., Koutník, J., eds.: Artificial Neural Networks - ICANN 2008. Volume 5163 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2008) 145–154
16. Cambria, E., Huang, G.B., Kasun, L.L.C., Zhou, H., Vong, C.M., Lin, J., Yin, J., Cai, Z., Liu, Q., Li, K., Leung, V.C., Feng, L., Ong, Y.S., Lim, M.H., Akusok, A., Lendasse, A., Corona, F., Nian, R., Miche, Y., Gastaldo, P., Zunino, R., Decherchi, S., Yang, X., Mao, K., Oh, B.S., Jeon, J., Toh, K.A., Teoh, A.B.J., Kim, J., Yu, H., Chen, Y., Liu, J.: Extreme Learning Machines. IEEE Intelligent Systems **28**(6) (2013) 30–59
17. Akusok, A., Veganzones, D., Miche, Y., Severin, E., Lendasse, A.: Finding originally mislabels with MD-ELM. In: Proc. of the 22th european symposium on artificial neural networks, computational intelligence and machine learning (ESANN'2014). (2014) 689–694
18. van Heeswijk, M., Miche, Y., Lindh-Knuutila, T., Hilbers, P., Honkela, T., Oja, E., Lendasse, A.: Adaptive ensemble models of extreme learning machines for time series prediction. In Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G., eds.: Artificial Neural Networks – ICANN 2009. Volume 5769 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2009) 305–314
19. Grigorievskiy, A., Miche, Y., Ventelä, A.M., Séverin, E., Lendasse, A.: Long-term time series prediction using op-elm. Neural Networks **51** (2014) 50–56 cited By 4.
20. Pouzols, F., Lendasse, A.: Evolving fuzzy optimally pruned extreme learning machine for regression problems. Evolving Systems **1**(1) (2010) 43–58
21. van Heeswijk, M., Miche, Y., Oja, E., Lendasse, A.: Gpu-accelerated and parallelized {ELM} ensembles for large-scale regression. Neurocomputing **74**(16) (2011) 2430 – 2437 Advances in Extreme Learning Machine: Theory and Applications Biological Inspired Systems. Computational and Ambient Intelligence Selected papers of the 10th International Work-Conference on Artificial Neural Networks (IWANN2009).
22. Yu, Q., Miche, Y., Eirola, E., van Heeswijk, M., Séverin, E., Lendasse, A.: Regularized extreme learning machine for regression with missing data. Neurocomputing **102** (2013) 45–51 cited By 9.
23. Benoît, F., van Heeswijk, M., Miche, Y., Verleysen, M., Lendasse, A.: Feature selection for nonlinear models with extreme learning machines. Neurocomputing **102** (2013) 111–124 cited By 8.
24. Akusok, A., Miche, Y., Hegedus, J., Nian, R., Lendasse, A.: A two-stage methodology using k-nn and false-positive minimizing elm for nominal data classification. Cognitive Computation **6**(3) (2014) 432–445 cited By 0.
25. Allen, D.M.: The relationship between variable selection and data agumentation and a method for prediction. Technometrics **16**(1) (February 1974) 125–127

26. Rao, C.R., Mitra, S.K.: Generalized Inverse of Matrices and Its Applications. John Wiley & Sons Inc (1971)
27. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
28. Eirola, E., Lendasse, A., Vandewalle, V., Biernacki, C.: Mixture of gaussians for distance estimation with missing data. Neurocomputing **131** (2014) 32–42
29. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological) **39**(1) (1977) pp. 1–38
30. McLachlan, G., Krishnan, T.: The EM Algorithm and Extensions. Wiley Series in Probability and Statistics. John Wiley & Sons, New York (1997)
31. Schwarz, G.: Estimating the dimension of a model. The annals of statistics **6**(2) (1978) 461–464
32. McLachlan, G.J., Peel, D.: Finite Mixture Models. Wiley Series in Probability and Statistics. John Wiley & Sons, New York (2000)
33. Lichman, M.: UCI Machine Learning Repository (2013) `http://archive.ics.uci.edu/ml`.
34. Myers, R.: Classical and Modern Regression with Applications. Bookware Companion Series. PWS-KENT (1990)
35. Akusok, A., Grigorievskiy, A., Lendasse, A., Miche, Y.: Image-based classification of websites. In Villmann, T., Schleif, F.M., eds.: Machine Learning Reports 02/2013. Volume ISSN: 1865-3960 of Machine Learning Reports., Saarbrücken, Germany, Workshop of the GI-Fachgruppe Neuronale Netze and the German Neural Networks Society in connection to GCPR 2013 (September 2013) 25–34 Proceedings of the Workshop - New Challenges in Neural Computation 2013.