Environmental Modelling & Software 73 (2015) 175–188

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

Nonlinear regression in environmental sciences using extreme learning machines: A comparative evaluation

Aranildo R. Lima [a, *], Alex J. Cannon [b, 1], William W. Hsieh [a]

[a] Department of Earth, Ocean and Atmospheric Sciences, University of British Columbia, Vancouver, BC, Canada
[b] Pacific Climate Impacts Consortium, University of Victoria, Victoria, BC, Canada

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form
15 May 2015
Accepted 4 August 2015
Available online xxx

Keywords:
Extreme learning machines
Support vector machine
Artificial neural network
Regression
Environmental science
Machine learning

ABSTRACT

The extreme learning machine (ELM), a single-hidden layer feedforward neural network algorithm, was tested on nine environmental regression problems. The prediction accuracy and computational speed of the ensemble ELM were evaluated against multiple linear regression (MLR) and three nonlinear machine learning (ML) techniques – artificial neural network (ANN), support vector regression and random forest (RF). Simple automated algorithms were used to estimate the parameters (e.g. number of hidden neurons) needed for model training. Scaling the range of the random weights in ELM improved its performance. Excluding large datasets (with large number of cases and predictors), ELM tended to be the fastest among the nonlinear models. For large datasets, RF tended to be the fastest. ANN and ELM had similar skills, but ELM was much faster than ANN except for large datasets. Generally, the tested ML techniques outperformed MLR, but no single method was best for all the nine datasets.

1. Introduction

Linear models are simple, fast, and often provide adequate and interpretable descriptions of how the predictors affect the outputs. In particular, for prediction purposes they sometimes outperform fancier nonlinear models (Hastie et al., 2009). However, environmental problems are generally complex with many components, such as trends, seasonality, interactions between predictors, and nonlinear relationships (Kuligowski and Barros, 1998; Cawley et al., 2007). Predictions by linear models may have accuracy limitations due to the generalization (and extrapolation) of the environmental problems by linear functions.

To overcome the limitations of linear models, nonlinear machine learning (ML) methods have been successfully used in environmental problems (Cherkassky et al., 2006). Artificial neural networks (ANN), support vector regression (SVR) and random forests (RF) have been used to solve hydrological problems, such as the prediction of 10-day reservoir inflows, downscaling precipitation, and prediction of water resource variables (e.g. flow, water level, nitrate, salinity and suspended sediment concentration) (Tripathi et al., 2006; Chen et al., 2010; Maier et al., 2010; Cannon, 2012b; Rasouli et al., 2012). Due to the large amount of research activity in hydrology using ANN models, good review papers (Maier and Dandy, 2000; Abrahart et al., 2012; Maier et al., 2010) are available in the hydrological literature. For hydrological problems, gradient-based optimization methods are widely used to train ANN models (Maier et al., 2010). This is also true for environmental problems such as forecasting wind power (Kusiak et al., 2009) and equatorial Pacific sea surface temperatures (Aguilar-Martinez and Hsieh, 2009). However, nonlinear optimization by gradient descent-based learning methods is computationally expensive and may easily converge to local minima. Nature inspired evolutionary computation algorithms have been successfully applied to ANN training (Leung et al., 2003; Chen and Chang, 2009), however there is no consensus on their superior skills or convergence speed (Solomatine and Ostfeld, 2008; Piotrowski and Napiorkowski, 2011). Consequently, there is still a need for better and faster ANN training algorithms.

The extreme learning machine (ELM) algorithm for single-hidden layer feedforward neural networks (SLFNs) was proposed by Huang et al. (2006). The ELM algorithm implements an SLFN similar in structure to a traditional ANN model, but the ELM randomly chooses the weights leading to the hidden nodes or

* Corresponding author. Department of Earth, Ocean and Atmospheric Sciences, University of British Columbia, 2020 – 2207 Main Mall, Vancouver, BC, V6T 1Z4, Canada.
E-mail address: arodrigu@eos.ubc.ca (A.R. Lima).
[1] Current address: Climate Data and Analysis Section, Climate Research Division, Environment Canada, Canada.

http://dx.doi.org/10.1016/j.envsoft.2015.08.002
1364-8152/

neurons (HN) and analytically determines the weights at the output layer (Schmidt et al., 1992). Once the activation function has been chosen, the only parameter to be tuned in the ELM is the number of HN. ELM has been used in different research areas (Sun et al., 2008; Huang et al., 2011) and has been found to produce good generalization performance with learning times that are thousands of times faster than traditional gradient-based ANN training algorithms.

Correct adjustment of the model parameters is mandatory for building a successful predictive model. For example, in ANN models, the number of HN, the regularization parameter (i.e. weight penalty parameter), and the algorithm for network training all need to be specified (Haykin, 1998; Hsieh, 2009). Unfortunately (with very few exceptions), there is no unique formula to estimate the model parameters before training starts, so they are usually determined by time-consuming trial and error. In addition, many published articles do not reveal the details of parameter estimation, hiding pitfalls and misuses of the technique employed in the studies (Zhang, 2007; Maier et al., 2010; Wu et al., 2014). For ELM the correct adjustment of the parameter is also crucial. Similar to ANN, the optimal number of HN is problem dependent and unknown in advance (Huang et al., 2006; Guorui et al., 2009; Liu and Wang, 2010). Thus, the process to find the optimal number of HN is also by the trial and error procedure, necessitating multiple ELM runs (Parviainen and Riihimaki, 2013).

The objective of this study is to test whether the claims of ELM having good generalization performance, requiring less computational time than other common nonlinear ML methods and having only one parameter (the number of HN) to adjust, are applicable in the context of environmental sciences. We applied ELM to nine very different datasets. Multiple linear regression (MLR) (with stepwise selection of predictors) was used as the benchmark to compare accuracy (based on the root mean squared error and mean absolute error) and computational time. We also used three nonlinear techniques that have previously been successfully applied to environmental problems as ML benchmarks, namely gradient-based ANN (Cannon and Lord, 2000; Schoof and Pryor, 2001; Krasnopolsky, 2007), SVR (Tripathi et al., 2006; Lima et al., 2013), and RF (Ibarra-Berastegi et al., 2011). We combined the ML techniques with simple automated procedures to search for optimal parameters.

Section 2 describes the datasets used in our study. The regression methods are presented in Sections 3, 4 and 5. Results and discussion of the experiments are given in Section 6, followed by summary and conclusion in Section 7.

2. Data description

In our present peer-reviewed publication system, publications biased in favour of new methods are common because authors tend to present their new methods in the best possible light to enhance their chances of passing the peer review. For instance, an author might test their new method against a traditional method on two different datasets. If the new method fails against a traditional method on one of the datasets, the author could ignore the failure and write a journal paper describing the new method using only the successful case (Hsieh, 2009; Zhang, 2007; Elshorbagy et al., 2010a). Environmental datasets span a broad range in terms of size, number of predictors, degree of nonlinearity, signal-to-noise ratio, etc. Cherkassky et al. (2006) grouped the environmental sciences applications in three domains: climate, Earth and ocean (which is closely related to climate because global processes on the Earth and in the ocean directly influence the climate), and hydrology. Thus, to explore the applicability of the ELM as a prediction tool in environmental sciences, we selected nine different datasets from the three domains, with four of them in climate (ENSO, YVR, PRECIP and TEMP), two in Earth and ocean (WIND and SO_2), and three in hydrology (SFBAY, FRASER and STAVE) (see the Appendix for details). While the selected datasets by no means span the whole range of environmental datasets, they do provide better insight into the advantages and disadvantages of various methods than a single dataset.

As part of the ML development process, the available data are generally divided into training and testing subsets. The training set is used to build the ML models and the testing set is used to determine the generalization ability of the trained models. Table 1 shows the specifications of the datasets and the basic descriptive statistics of the predictand (i.e. response variable) of the training and testing sets.

One of the most important steps in the ML modelling process is the determination of an appropriate set of predictors (Cherkassky et al., 2006; Solomatine and Ostfeld, 2008; Maier et al., 2010). Usually, they are determined on an ad hoc basis or by using a priori system knowledge (Maier and Dandy, 2000). Since the focus of this research is the comparison of ML models, emphasis was not given to the identification of the optimal predictors for each particular dataset.

3. Extreme learning machines

The key element of an ANN is its distributed, nonlinear structure. An ANN is composed of a large number of highly interconnected neurons (or perceptrons) divided in layers (input, hidden, output), but working in concert to solve a prediction problem (Haykin, 1998).

The multilayer perceptron (MLP) architecture is probably the most popular type of ANN (Kuligowski and Barros, 1998; Cawley et al., 2003; Haylock et al., 2006) and consists of at least one hidden layer sandwiched between the input and output layers. Training of the ANN model involves adjusting the parameters iteratively so the error between the model output $\hat{\mathbf{y}}$ and the predictand data $\mathbf{y}$ is minimized. The backpropagation algorithm is often used to calculate the gradient of the error function, with a gradient-descent approach used to reduce the mean squared error iteratively, which could be time consuming. There are also others issues to consider such as the number of learning epochs, learning rate, stopping criteria, regularization and/or local minima. To overcome some of these issues, an algorithm called extreme learning machine (ELM) for SLFNs can be used.

The SLFN can be defined as:

$$\hat{\mathbf{y}}_j = \sum_{i=1}^{L} \boldsymbol{\beta}_i s(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) + \boldsymbol{\beta}_0, \quad (j = 1, \ldots, N) \tag{1}$$

Table 1
Specification of the tested datasets.

Datasets	# observations		# predictors	Predictand			
				Mean		Std. dev.	
	Train	Test		Train	Test	Train	Test
ENSO	264	120	8	−0.049	−0.025	0.902	0.800
SFBAY	475	157	6	1.177	1.145	1.226	1.359
WIND	279	100	160	36,952	40,655	28,610	29,165
FRASER	2557	1095	4	1.779	1.693	0.547	0.528
YVR	3286	1694	3	1.451	1.444	0.457	0.460
STAVE	5258	1260	25	3.134	3.088	0.909	0.919
PRECIP	4213	2074	106	0.984	0.981	0.342	0.341
TEMP	7117	3558	106	−0.002	−0.010	1.010	0.983
SO_2	15,110	7533	27	3.125	3.147	0.929	0.924

where $\mathbf{x}_j \in \mathbb{R}^d$ and $\hat{\mathbf{y}}_j \in \mathbb{R}^m$ are the model input and output, respectively, N the number of cases, $\mathbf{w}_i$ and b_i the weight and bias parameters in the hidden layer (with L hidden nodes), $\boldsymbol{\beta}_0$ the bias parameter in the output layer, and the sigmoidal activation function s is

$$s(x) = \frac{1}{1 + \exp(-x)}. \tag{2}$$

Huang et al. (2006) proved that the $\mathbf{w}_i$ and b_i parameters can be randomly assigned (Schmidt et al., 1992) if the activation function is infinitely differentiable, so only the $\boldsymbol{\beta}$ parameters need to be optimized when minimizing the mean squared error between the model output $\hat{\mathbf{y}}$ and the target data $\mathbf{y}$. Thus, in the ELM approach, training an SLFN is equivalent to simply finding the least-squares solution of the linear system (Huang et al., 2011), $\mathbf{H}\boldsymbol{\beta} = \mathbf{Y}$, where the hidden layer output matrix $\mathbf{H}$ of dimension $N \times (L+1)$ is

$$\mathbf{H} = \begin{bmatrix} 1 & s(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & s(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & s(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & s(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}, \tag{3}$$

the $\boldsymbol{\beta}$ parameter matrix of dimension $(L+1) \times m$ and the target data matrix $\mathbf{Y}$ of dimension $N \times m$ are

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_0^{\mathrm{T}} \\ \boldsymbol{\beta}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{\beta}_L^{\mathrm{T}} \end{bmatrix}, \quad \text{and} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{y}_N^{\mathrm{T}} \end{bmatrix}. \tag{4}$$

Algebraically, the linear system for $\boldsymbol{\beta}$ is solved via the Moore–Penrose generalized inverse $\mathbf{H}^{\dagger}$ (Huang et al., 2011),

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{Y}. \tag{5}$$

The bias parameter $\boldsymbol{\beta}_0$ is not essential for ELM, and we tested ELM both with and without $\boldsymbol{\beta}_0$ (where the first column of ones in $\mathbf{H}$ and $\boldsymbol{\beta}_0^{\mathrm{T}}$ in Eqs. (3) and (4) are deleted, as in Huang et al., 2006).

3.1. Ensemble of ELM

Like many other learning algorithms the stochastic nature of ELM means that different trials of simulations may yield different results. The random assignment of weight and bias parameters in the hidden layer makes each ELM distinct, so we used an ensemble (or committee) of ELMs. We repeatedly ran the ELM for E times with the same dataset and network structure (number of HN). The output of the ELM ensemble is the averaged output of the individual ensemble members. As shown in Sun et al. (2008) and Liu and Wang (2010), output from an ELM ensemble is more stable than from a single ELM.

3.2. Range of weights and ELM predictions

Proper random initialization of the weights is one of the most important prerequisites for fast convergence of feedforward ANN models like MLP. The effectiveness of random weight initialization depends on the initial weight distribution. A balance must exist to ensure that the sigmoidal function does not remain completely linear nor become saturated near the asymptotic limits of 0 and 1. If the weight distribution is too narrowly centred about the origin, both activation and error signals will die out on their way through the network. If it is too broad, the saturated activation function will block the backpropagated error signals from passing through the node.

Often a uniform distribution, spread over a fixed interval $[-r, r]$, is used for the weights. Thimm and Fiesler (1997) gave a review of random weight initialization methods for MLP and found that r should be of the form

$$r = a \cdot F^{-0.5}, \tag{6}$$

where F is the fan-in of the node (i.e. the number of incoming weights). Using hyperbolic tangent as the activation function, they found $a \approx 1$ to be a reasonable value.

Parviainen and Riihimaki (2013) raised questions about meaningfulness of choosing model complexity based on HN only, as is traditionally done with ELM. Usually, only the number of HN is carefully selected, and a fixed variance or interval parameter is used for the distribution from which the random weights are drawn. The width of the interval is seen as simply a constant instead of a model parameter. As weights are random variables, diversity (i.e. not having similar values for all weights) is an important factor because model complexity is affected by the variance of the distribution. Small variance means small weights and linear models, while large variance can produce larger weights and nonlinear models.

Also, depending on how the random weights and bias parameters are selected for the hidden nodes, the learning time for ELM can decrease dramatically (Liu and Wang, 2010). Parviainen and Riihimaki (2013) found that using a different variance can improve performance, achieving a similar effect as regularizers in traditional ANN models. Thus they concluded that the weight variance should also be a parameter of ELM.

Huang et al. (2012) proposed an ELM using a regularization parameter (i.e. weight penalty), as the only parameter to be adjusted. They set the number of HN to 1000 and searched 50 different values (e.g. $[2^{-24}, 2^{-23}, \ldots, 2^{24}, 2^{25}]$) to find the best weight penalty parameter. They found that the generalization performance was not sensitive to the number of HN as long as it was set large enough ($\geq$1000 for their experiments). Two considerations: First, the intervals and ranges of the 1-dimensional (1-D) optimization have to be chosen carefully to avoid bad generalization performance (if the 1-D is too coarse) or high computational cost (if the 1-D is too refined). Second, if the number of HN is set large then the 1-D search will perform operations on large matrices. Thus there is no guarantee that the computational cost will be less than a simple sequential search of the number of HN.

Model selection with two parameters is slower than with only one. To avoid this drawback, we adopted the approach used by Thimm and Fiesler (1997) to define a reasonable range for weight initialization. Because we used the logistic sigmoidal function, we chose $a = 2$ in Eq. (6) (equivalent to choosing $a = 1$ with the hyperbolic tangent function). Similarly, the random bias parameter of the HN in our ELM is uniformly distributed within $[-2, 2]$.

3.3. Network structure

Finding an appropriate network structure for ANNs involves the selection of a suitable number of HN. The optimal network structure is a balance between generalization ability and network complexity. If network complexity is too low (i.e. too few HN), the network might be unable to capture the true nonlinear relationship. If network complexity is too high, the network might have decreased generalization ability due to overfitting (i.e. fitting to noise in the data) and increased training time. However, the choice of the optimal number of HN is problem dependent, i.e. different phenomena will require distinct number of HN to build a successful regression model. In general, the number of HN is selected empirically (Huang et al., 2006; Sun et al., 2008; Guorui et al., 2009; Liu and Wang, 2010).

For an ELM, the range in which the optimal number of HN can be found is far wider (sometimes by orders of magnitude) than for an ANN using iterative nonlinear optimization. Therefore, vigilant care

must be taken with the automated procedure used to select the number of HN. Automatic procedures generally use a predefined step size to find a solution for a given problem. However, by starting with small step sizes the algorithm could spend a considerable amount of time searching for an optimal solution. Starting with large step sizes, the algorithm could miss parsimonious models (models with fewer HN). Parsimonious models are desirable because increasing the number of HN leads to increasing computational time and potential to overfit. In other words, the appropriate fixed step size for a sequential/1-D search is hard to determine. We used the hill climbing algorithm to find the number of HN of the ELM.

4. Hill climbing

Hill climbing is a simple mathematical optimization technique that starts with an arbitrary solution to a problem, then attempts to find a better solution (in our case smaller root mean squared error, RMSE) by incrementally changing a single element of the solution (in our case the number of HN). If the change produces a better solution, an incremental change (often by a defined step size) is made towards the new solution. Iterations continue until no further improvements can be found. In the hill climbing algorithm used here, the step size is automatically adjusted by the algorithm. Thus it shrinks when the probes do poorly and it grows when the probes do well, helping the algorithm to be more efficient and robust (Yuret, 1994). The necessary steps for implementing the hill climbing algorithm are shown in Algorithm 1.

For each candidate solution a *k*-fold cross-validation within the training set is performed to avoid overfitting. In the *k*-fold cross-validation, the training set is split into *k* smaller sets. Of the *k* sets, a single set is retained as the validation data for validating the model, and the remaining $k-1$ sets are used as training data. This procedure is then repeated *k* times so every fold has been used for validation. Thus, the final skills of the candidate is based on validation using all *k* folds.

```
1  begin
2      // Variable Initialization;
3      acce  // Defines the factor of expansion and contraction of the search space;
4      step.size ← initialStepSize  // Defines the expansion and contraction of the search space;
5      ensemble.number // number of ensemble members;
6      ensemble.number.training // smaller than ensemble.number;
7      candidates ← [−1/acce, 0, 1/acce, acce] // vector containing the solution candidates;
8      best.HN ← initialnumberHN // best number of hidden neurons;
9      best.rmse ← ∞ // best root mean squared error (RMSE);
10     // Searching the best number of HN;
11     while Best model = False do
12         for j=1 to 4 do
13             for e = 1 to ensemble.number.training do
14                 HN.candidate ← round(best.HN + step.size*candidates[j]) // number of HN
                   according to the candidate and the step size;
15                 for n = 1 to Number of folds used for cross-validation do
16                     Evaluate(ELM using HN.candidate) using Eq.(5);
17                 end
18             end
19         end
20         best.solution ← smaller RMSE among the cross-validated candidates;
21         best.j ← index j of the smallest RMSE among the cross-validated candidates;
22         if best.rmse < best.solution then
23             // Checking if the best candidate is the same as in the previous step;
24             if (best.HN = round(best.HN + step.size*candidates[best.j])) then
25                 Best model = True
26             else
27                 // Adjusted according with the best candidate;
28                 best.rmse ← best.solution;
29                 best.HN ← round(best.HN + step.size*candidates[best.j]);
30                 step.size ← maximum(1,round(step.size*candidates[best.j]));
31             end
32         else
33             Best model = True
34         end
35     end
36     // Building the final model with the best number of HN;
37     for e = 1 to ensemble.number do
38         Evaluate(ELM with best.HN) using Eq.(5);
39     end
40 end
```

Algorithm 1: Hill climbing algorithm used to identify the optimum number of HN in the ELM models.

5. Benchmark models

For the sake of comparison we used four different prediction models (one linear and three nonlinear): MLR, bootstrap-aggregated ensemble ANN with an automated procedure to find the number of HN (henceforth just called ANN for brevity), random forest (RF), and support vector regression with evolutionary strategy (SVR-ES).

5.1. Artificial neural network

In this work the architecture used consists of an MLP neural network with one hidden layer and the analytical calculation of the gradient is made via backpropagation.

Bootstrap aggregation (bagging) (Breiman, 1996) is an ensemble method which allows greater model stability and more efficient use of training data relative to early stopping methods (methods which stop when the validation error is at a minimum instead of continuing optimization until errors on the training data are minimized) (Cannon and Whitfield, 2002). Bagging generates different training datasets by bootstrapping (i.e. resampling with replacement from the available cases). The resampled dataset are used to train the models in the ensemble. The out-of-bag cases (i.e. cases not selected in a bootstrap sample) can be used as a validation dataset, e.g. for determining when to stop training with the training dataset.

As the ANN with backpropagation requires fewer HN than ELM, a sequential, small 1-D search is often used to select the optimal number. The best number of HN is determined by the ensemble of ANN models based on the out-of-bag RMSE. To find the optimal number of HN we sequentially increased the number of HN one by one, until the maximum number of HN was achieved or consecutive increments were without improvements.

5.2. Random forest

Conceptually simple yet powerful, decision tree models can be applied to datasets with large numbers of cases and predictors and are extremely resistant to outliers. Among the decision trees used in ML, the classification and regression tree (CART) model, first introduced by Breiman et al. (1984), is the most commonly used. A regression tree grows via binary recursive partitioning using the sum of squares as the minimization criterion. A random forest (RF) (Breiman, 2001) is a bootstrap ensemble of many decision trees (CART). Each tree is grown over a bootstrap sample from the training dataset using a randomly selected subset of the available predictors at each decision branch. The average output of all trees is the output of the RF model.

5.3. Support vector regression with evolutionary strategy

SVR, a kernel method, maps the input data $\mathbf{x}$ into a higher dimensional feature space using a nonlinear mapping function $\mathbf{\Phi}$. The nonlinear regression problem between $\mathbf{x}$ and the predictand $\mathbf{y}$ can thus be converted to a linear regression problem between $\mathbf{\Phi}(\mathbf{x})$ and $\mathbf{y}$. The global minimum solution of the linear regression problem is achieved without nonlinear optimization, hence there are no local minima in the objective function.

The performance of an SVR model is highly dependent on the choice of the kernel function and the parameters (Hastie et al., 2009; Hsieh, 2009). The use of a suitable nonlinear kernel function in SVR allows it to be fully nonlinear, while the use of a linear kernel function restricts SVR to a linear model. To find the optimal SVR parameters, Lima et al. (2013) proposed the SVR-ES which was faster than a grid search. It combines some of the parameters estimation proposed by Cherkassky and Ma (2004) and a simple evolutionary strategy called "uncorrelated mutation with p step sizes" (Eiben and Smith, 2003). In this study, we used a nonlinear kernel, the radial basis function (RBF) (i.e. the Gaussian kernel), in the SVR-ES.

6. Experimental results

To demonstrate the practical use of the ELM as a prediction method for environmental sciences, experiments were performed on the datasets described in the Appendix. The experiments were divided into three groups. First, a comparison was made between the nonlinear methods using the automated procedures to find the parameters (Section 6.1). Second, to check if the automated procedures used to find the number of HN performed satisfactorily, a separate set of experiments involving a 1-D search was performed to determine approximately the optimal number of HN for ELM-E, ELM-S and ANN (Section 6.2). Third, to compare the ELM-S and ANN while excluding the time used to find the number of HN, another set of experiments was performed given the number of HN (Section 6.3).

All the prediction models can be built using the free R software environment for statistical computing (R Core Team, 2014). For RF we used the R package randomForest (Liaw and Wiener, 2002). We developed the SVR-ES using the R package e1071 (Meyer et al., 2014) and R native libraries. ANN uses the monmlp package (Cannon, 2012a). MLR, from the R package stats, uses stepwise selection of predictors (Venables and Ripley, 2002) based on the Akaike information criterion (AIC). We also tested MLR using Bayesian information criterion (BIC) and a 5-fold cross-validation. However we obtained similar results with minor differences in the tested cases. We developed the ELM using the R package MASS (Venables and Ripley, 2002) and R native libraries. SVR-ES and ELM code are freely available from the project website: http://forai.r-forge.r-project.org/. The experiments were realized using a R version 3.0.3 environment running in a i5-2500, 3.30 GHz CPU and 8 GB-RAM computer.

To measure the performance of nonlinear methods relative to the MLR, we calculated the skill score (SS) of the mean absolute error (MAE) and of the root mean squared error (RMSE). The SS is defined by:

$$\mathrm{SS} = \frac{A - A_{\mathrm{ref}}}{A_{\mathrm{perfect}} - A_{\mathrm{ref}}}, \tag{7}$$

where A is a particular measure of accuracy, A_{perfect} is the value of A for a set of perfect forecasts (zero for MAE or RMSE), and A_{ref} is the value of A computed over the set of reference forecasts (in our case the MAE or RMSE value of the MLR model). Positives SS means better performance than the reference model and negative values, worse performance.

6.1. Automated procedures

The experiments were designed to be fully automated, i.e. no human intervention was necessary to tune the model parameters. As the potential advantage claimed by the ELM practitioners is the powerful combination of better generalization with low training/prediction time, we also measured the cpu time necessary to train and test the models. For model timing, we took into account the time used to tune the parameters automatically, to train the model with the estimated parameters, and to generate predictions for the test data using the trained model.

Data were standardized (zero mean and unit variance) and divided into a training set and an independent test set to test the

trained model. The test set was kept completely unseen during the training process. The RF and ANN performed their own split-sample validation via the out-of-bootstrap samples in the bootstrap aggregation. The SVR-ES and the ELM with hill climbing performed their own 5-fold cross-validation when choosing the best parameters (number of HN for ELM and C, ε and γ for SVR, which control the regularization, the insensitivity of the error norm and the width of the Gaussian kernel function, respectively, in SVR).

Thirty different trials were performed with different seeds for the random number generator. At each trial an ensemble of 30 members were used for the ELM, ANN, and SVR-ES and the averaged output of the individual members used as the output of the ensemble. The ANN setup was as follows: the range of initial uniformly-distributed random weights was [−0.5, 0.5], and the maximum number of iterations was 5000. The stopping criterion for the automatic choice of the number of HN was three consecutive increments without improvements in the objective function or the number of HN reached 50. For RF, we used 500 as the number of generated trees, 1 as the minimum size of terminal nodes, and a search in the randomForest package was applied to find the optimal number of predictor variables to be randomly selected at each split (as determined by out-of-bag error estimates). For the SRV-ES, C and ε were initialized by the Cherkassky and Ma (2004) guidelines and γ by 0.001, and the stopping criterion for the evolutionary algorithm was 50 consecutive generations without improvements or the number of generations reached 500. After the three parameters were tuned, we further performed bagging (30 times) to improve the SVR final results.

To check some issues raised in Section 3, we performed a preliminary experiment with ELM. First we tested the original setup used by Huang et al. (2006) in which the predictors were normalized to the range [0, 1] and the predictand to [−1, 1]. Although Huang et al. (2006) did not mention the range of the weights used, the paper provides a link to a downloadable ELM code where the weights are randomly assigned from a uniform distribution of [−1, 1] and the bias parameters from a uniform distribution of [0, 1]. Also the output node is linear and without a bias parameter. We averaged the results of 30 trials of simulations and we used hill climbing to find the optimal number of HN with $E = 1$ (i.e. a single ELM for each trial). The RMSE SS from this original ELM is represented in Fig. 1 under ELM-O. Then we reran with $E = 30$ (i.e. 30 ensemble members for each trial), with results indicated in Fig. 1 by ELM-E. With $E = 30$ and having added the bias parameter in the output layer we ran ELM-B. For the final model based on Section 3.2, predictors and predictand were standardized to zero mean and unit standard deviation, the bias parameters in the HN were randomly chosen from a uniform distribution of [−2, 2], and the weights in the HN were randomly assigned from a uniform distribution of $[-r, r]$, with r scaled by Eq. (6) (with $a = 2$). This ELM-S model was run with $E = 30$ and with a bias parameter in the output node (Fig. 1). As expected, ELM-O was unstable (with a few exceptions), hence the hill climbing algorithm was not able to find the proper number of HN. Consequently ELM-O performed poorly compared to the others and in some cases to MLR. Huang et al. (2006) gradually increased the number of HN by increments of 5 (using cross-validation) to find the optimal number of HN for ELM. ELM-E improved the stability of ELM-O and the hill climbing performed satisfactorily. The only noticeable improvement between ELM-E and ELM-B is in SFBAY. Hence the bias parameter in the output layer was retained in ELM-S as the SS might improve at almost no cost. Overall ELM-S using Eq. (6) to scale the range of weights tended to have better SS than the other three setups.

Next we compare the ELM-S model performance over test data to those of the other models (RF, ANN and SVR-ES) in terms of the MAE and RMSE SS (Figs. 2 and 3) and the computer time (Fig. 4) over the nine datasets. In addition, Tables 2–4 show respectively MAE, RMSE, and computer time over the nine datasets.

For the ENSO dataset, all the nonlinear methods performed better than MLR (SS = 0). All the nonlinear models had similar RMSE SS (Fig. 3). Overall RF tended to perform better than the other nonlinear models based on the MAE SS (Fig. 2). For the mean time spent to build the models (Fig. 4 and Table 4), all the models were fast and the ELM-S ran at 2.18×, 62.3× and 7.45× the speed of RF, ANN and SVR-ES respectively.

For the SFBAY dataset, the nonlinear models all had SS advantages over MLR. ANN had the best RMSE SS followed by ELM-S,

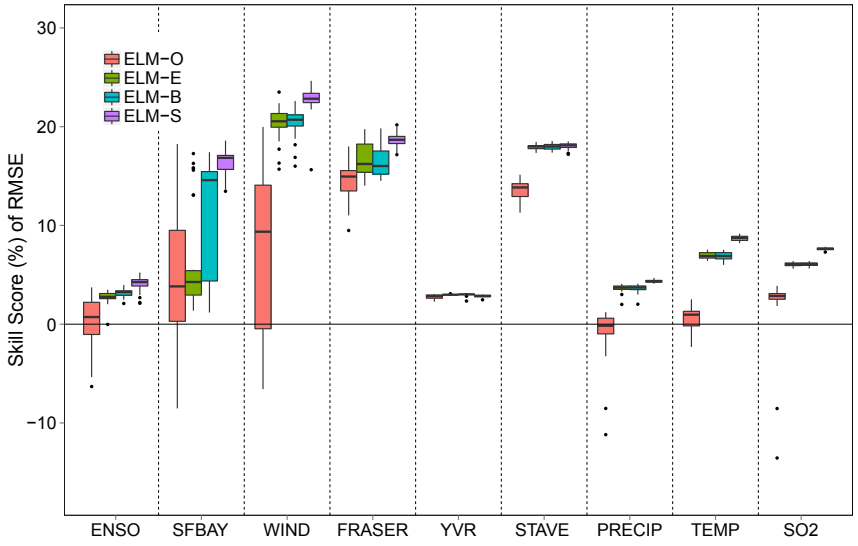


Fig. 1. Comparison between different ELM setups, with the "waistline" of the boxplot showing the median SS of the 30 trials for the nine datasets. The 25th and 75th percentiles are shown, respectively, as the bottom and top of each box. Distance between the 25th and 75th percentiles is the interquartile range (IQR). The upper whisker extends from the upper hinge to the highest value that is within 1.5 IQR of the hinge. The lower whisker extends from the lower hinge to the lowest value within 1.5 IQR of the hinge. Data beyond the end of the whiskers are outliers and are plotted as points. The four models are the original single ELM (ELM-O), the ensemble of 30 models (ELM-E), the ensemble with output bias parameter (ELM-B) and the ELM ensemble with scaled weights (ELM-S).

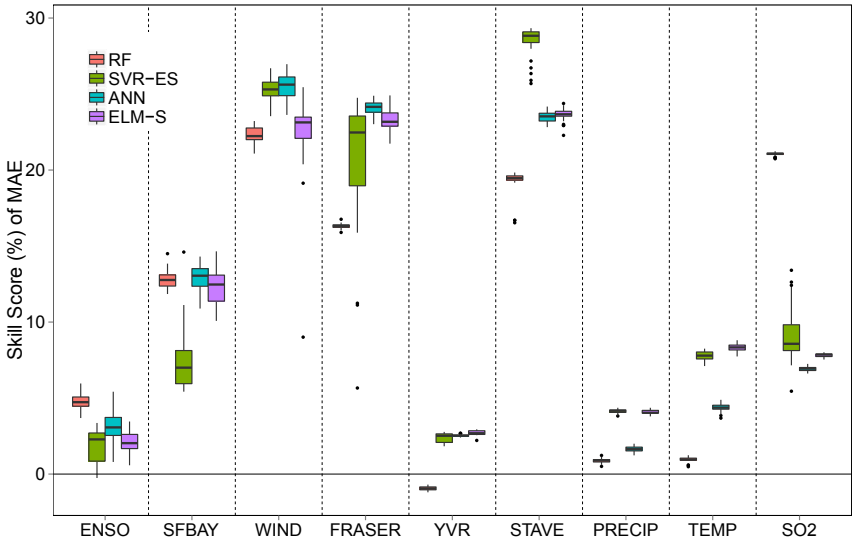


Fig. 2. Boxplot of the MAE SS of the 30 trials.

while SVR-ES and RF had similar RMSE SS. RF, ANN and ELM-S had similar MAE SS. Overall ANN tended to perform better than the other nonlinear models. ELM-S ran at 1.90×, 32.7× and 12.2× the speed of RF, ANN and SVR-ES, respectively.

For the WIND dataset, even the worst nonlinear model (RF) had a much better SS than the MLR model, and ANN slightly led the other nonlinear models in the MAE and RMSE SS. Overall ANN tended to perform better than the other nonlinear models. WIND has a relatively large number of predictors (160) when compared to the number of training cases (279). This could explain the large standard deviation in the ANN's computing time (Fig. 4 and Table 4), as adding one extra HN means 162 more weights to adjust in the ANN. The ELM-S ran at 5.34×, 118× and 8.86× the speed of RF, ANN and SVR-ES, respectively.

For the FRASER dataset, again all nonlinear models had a large advantage over MLR based on the SS, with ANN leading in the SS. This dataset has more observations than the ones mentioned before (Table 1), but it has only four predictors so ANN had less variability. The ELM-S ran at 2.03×, 37.8× and 32.8× the speed of RF, ANN and SVR-ES, respectively. Overall ANN tended to perform better than the others techniques for the small datasets (ENSO, SFBAY, WIND and FRASER). The powerful combination of ANN and bagging is likely responsible for the better performance as bagging is helpful when the number of cases is not large.

For the YVR dataset, RF performed worse than MLR in SS while the other nonlinear methods all outperformed MLR. Overall SVR-ES, ANN and ELM-S had similar skills. As expected for precipitation problems (similarly in PRECIP), the skills were not as high as in others problems due to the complexity and difficulty of forecasting the phenomenon itself (Kuligowski and Barros, 1998; Cawley et al., 2007). ELM-S kept its speed performance, running at 3.38×, 13.0× and 60.8× the speed of RF, ANN and SVR-ES respectively.

For the STAVE dataset, SVR-ES tended to perform best overall based on the MAE SS, though ELM-S was slightly ahead in the RMSE SS. RF was the fastest (but with the lowest SS among the nonlinear models), ANN and ELM-S had similar computational time and SVR-ES was the slowest.

For PRECIP, ELM-S and SVR-ES attained the highest SS. However

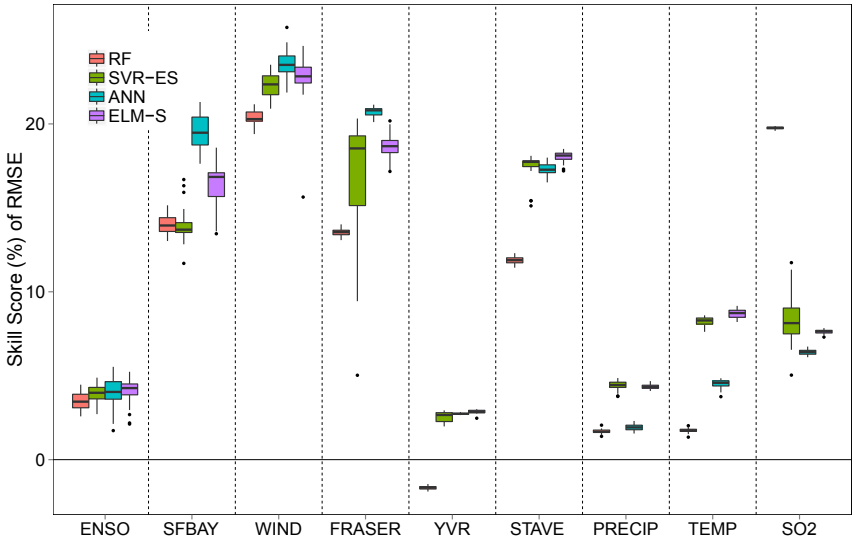


Fig. 3. Boxplot of the RMSE SS of the 30 trials.

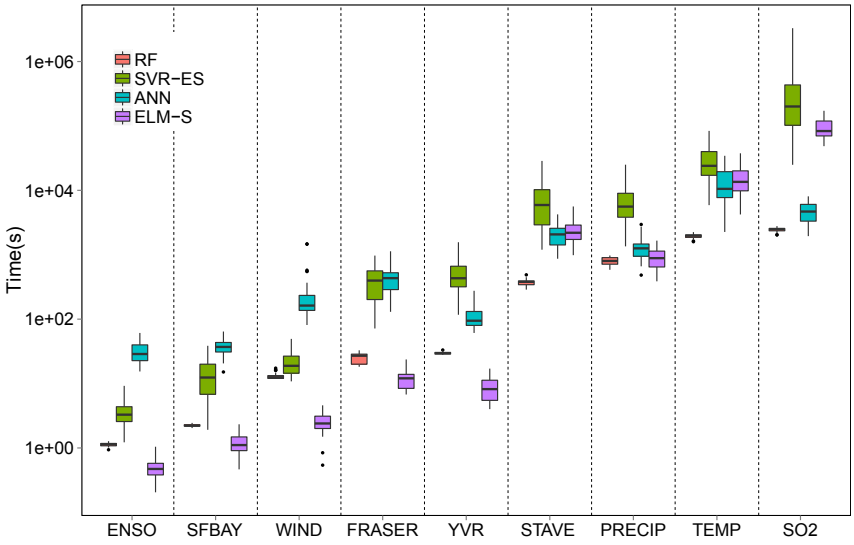


Fig. 4. The total computer time used for building (including testing) a model (RF, ELM-S, ANN or SVR-ES), plotted on a log scale (base 10).

Table 2
Mean and standard deviation (in parenthesis) of MAE (over 30 trials) from the test set using RF, SVR-ES, ANN, and ELM-S. The model with the lowest MAE of each dataset is shown in boldface.

	MAE				
	MLR	RF	SVR-ES	ANN	ELM-S
ENSO	0.437	**0.416**(0.002)	0.429(0.005)	0.423(0.005)	0.428(0.003)
SFBAY	0.652	0.569(0.004)	0.603(0.013)	**0.568**(0.006)	0.571(0.008)
WIND	22,848	17,742(112)	17,070(164)	**17,016**(188)	17,730(645)
FRASER	0.240	0.201(0.000)	0.191(0.011)	**0.182**(0.001)	0.184(0.002)
YVR	0.363	0.366(0.000)	0.354(0.001)	**0.353**(0.000)	**0.353**(0.001)
STAVE	0.201	0.162(0.002)	**0.144**(0.002)	0.153(0.001)	0.153(0.001)
PRECIP	0.235	0.233(0.000)	**0.225**(0.000)	0.231(0.000)	**0.225**(0.000)
TEMP	0.213	0.211(0.000)	0.196(0.001)	0.203(0.001)	**0.195**(0.001)
SO_2	0.629	**0.497**(0.001)	0.572(0.012)	0.586(0.001)	0.580(0.001)

the nonlinear models had only minor improvements over the linear model. The computational time was highest for SVR-ES, followed by ANN, ELM-S and RF.

For TEMP, ELM-S had the best SS around 10% (Figs. 2 and 3) followed closely by SVR-ES. ANN and ELM-S had comparable times but RF was again the fastest and SVR-ES the slowest.

For SO_2, RF had much better SS (Figs. 2 and 3) than the other three nonlinear methods, all of which performed better than MLR. In SS, SVR-ES showed large scatter, while ELM-S did better than ANN. However, for computational time, ELM-S performed worse than ANN and RF (Fig. 4). This occurred because the number of HN needed by ELM-S was large (Fig. 5). RF, the fastest model, ran at 2.02× the speed of ANN, which was the second fastest. Thus RF is clearly the best method for this dataset.

Table 6 summarizes the results by showing the average performance (rank) over the tested datasets. However, one must avoid trying to identify a single "winner" method as the datasets are very different from each other and they came from different domains in the environmental sciences.

The number of steps used to find the number of HN by the ANN sequential procedure and the ELM-S hill climbing procedure are similar. Although hill climbing uses adaptive steps, the computational cost necessary to solve Eq. (5) is still high when using a large number of HN. Fig. 4 shows ELM-S losing its time advantage when

Table 3
Mean and standard deviation (in parenthesis) of RMSE (over 30 trials) from the test set using RF, SVR-ES, ANN, and ELM-S. The model with the lowest RMSE of each dataset case is shown in boldface.

	RMSE				
	MLR	RF	SVR-ES	ANN	ELM-S
ENSO	0.550	0.531(0.003)	0.528(0.003)	**0.527**(0.005)	**0.527**(0.004)
SFBAY	1.050	0.903(0.006)	0.904(0.011)	**0.845**(0.011)	0.878(0.014)
WIND	27,772	22,109(116)	21,586(200)	**21,226**(244)	21,477(414)
FRASER	0.306	0.264(0.001)	0.255(0.011)	**0.242**(0.001)	0.248(0.002)
YVR	0.439	0.447(0.001)	0.428(0.001)	**0.427**(0.000)	**0.427**(0.000)
STAVE	0.294	0.259(0.001)	0.242(0.002)	0.243(0.001)	**0.241**(0.001)
PRECIP	0.289	0.285(0.000)	**0.277**(0.001)	0.284(0.001)	**0.277**(0.000)
TEMP	0.275	0.270(0.000)	0.252(0.001)	0.263(0.001)	**0.251**(0.001)
SO_2	0.802	**0.644**(0.001)	0.734(0.013)	0.751(0.001)	0.741(0.001)

Table 4
Mean and standard deviation (in parenthesis) of total computer time (in seconds) used for building (including testing) a model. The model with the lowest computer time of each dataset is shown in boldface. The number of HN is also shown.

	RF	SVR-ES	ANN		ELM-S	
	Time	Time	Time	HN	Time	HN
ENSO	1(0)	4(2)	32(12)	6(2)	< **1**(0)	11(3)
SFBAY	2(0)	14(9)	38(11)	6(2)	**1**(0)	17(5)
WIND	13(2)	22(9)	290(346)	2(2)	**2**(1)	24(7)
FRASER	25(4)	404(244)	465(270)	15(6)	**12**(4)	31(7)
YVR	30(1)	532(337)	114(49)	3(2)	**9**(3)	19(4)
STAVE	**375**(46)	8019(6750)	2050(772)	16(3)	2533(1217)	415(49)
PRECIP	**811**(107)	6774(5083)	1327(569)	3(1)	930(349)	282(41)
TEMP	**1971**(164)	30,147(18,311)	13,845(7429)	14(3)	14,846(7386)	771(138)
SO_2	**2475**(199)	446,950(689,120)	5006(1673)	13(3)	94,768(34,676)	1440(162)

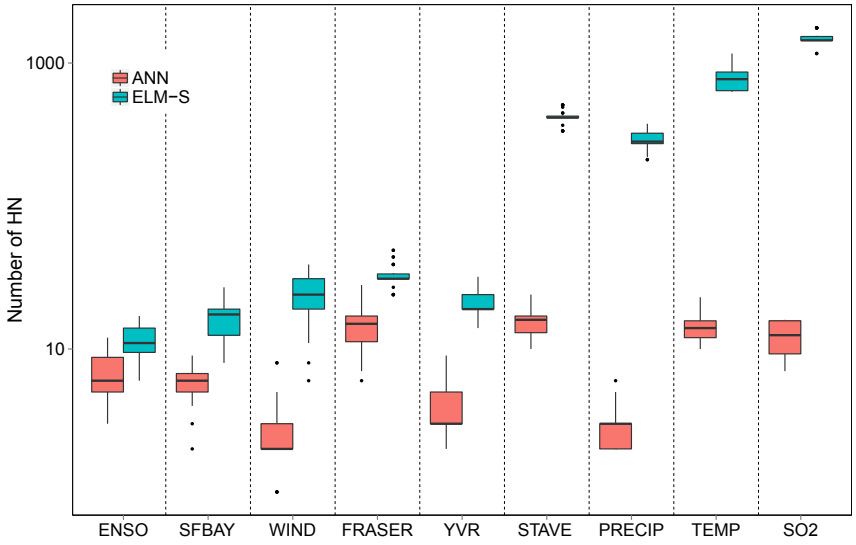


Fig. 5. Boxplot of the number of HN (plotted on a log scale) used in the ANN and ELM-S models as chosen by the automated procedures (30 trials).

large numbers of HN (Fig. 5 and Table 4) have been used.

Also associated with the size of the training set (including number of cases and number of predictors) is the number of HN used by the ELM-S (Table 4) which is larger than the number of HN used by ANN. In particular, ELM needs more HN than ANN to compensate for the randomness of the parameters $\mathbf{w}_i$ and b_i in Eq. (1). However as mentioned in Section 3 this drawback is compensated by ELM having to find the least-squares solution of a linear system, which does not have multiple minima and is faster than nonlinear optimization. Even if ELM uses far more HN than ANN, it does not imply ELM uses more adjustable weight and bias parameters than ANN. If the feedforward neural network has d inputs, one layer of L HN and m outputs, the number of adjustable parameters is $(L+1)m$ for ELM and $(d+1)L+(L+1)m$ for ANN. For TEMP, with $d=106$ and $m=1$, $L=771$ for ELM and $L=14$ for ANN (Table 4) give 772 adjustable parameters for ELM and 1513 for ANN.

Liu et al. (2012) pointed out that ELM has superior computational time compared to the SVR, and this superiority increases drastically as the size of training set grows. Similar to Liu et al. (2012), Fig. 4 and Table 4 show that ELM-S is faster than SVR-ES

Table 5
Comparison of MAE, RMSE and time (in seconds) from the training and testing sets using the average number of HN selected by the automated procedures (Table 4). The lowest computer time (in seconds), test MAE and test RMSE of each dataset are shown in boldface.

	MAE				RMSE				Time			
	ANN		ELM-S		ANN		ELM-S		ANN		ELM-S	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
ENSO	0.317	**0.419**	0.336	0.429	0.423	**0.523**	0.444	0.530	3.334	**0.004**	**0.023**	0.005
SFBAY	0.471	**0.563**	0.480	0.571	0.659	**0.838**	0.674	0.870	4.903	**0.006**	**0.079**	**0.006**
WIND	14,067	**17,160**	15,109	17,556	17,834	**21,354**	18,423	21,372	12.679	**0.005**	**0.138**	0.016
FRASER	0.162	**0.184**	0.169	0.185	0.224	**0.243**	0.231	0.249	33.339	**0.026**	**0.573**	0.194
YVR	0.351	**0.353**	0.351	**0.353**	0.425	**0.427**	0.426	**0.427**	11.821	**0.014**	**0.431**	0.057
STAVE	0.150	**0.153**	0.144	**0.153**	0.250	0.243	0.239	**0.240**	107.639	**0.040**	**93.440**	1.077
PRECIP	0.226	0.231	0.210	**0.225**	0.279	0.284	0.260	**0.276**	85.886	**0.047**	**35.757**	1.867
TEMP	0.193	0.203	0.174	**0.195**	0.248	0.263	0.225	**0.251**	557.260	**0.197**	**434.336**	9.577
SO_2	0.580	0.589	0.523	**0.581**	0.742	0.754	0.672	**0.741**	**275.208**	**0.202**	3469.477	26.964

Table 6
Ranks based on the MAE and RMSE (over 30 trials) for the test data, with rank 1 being the best. The bottom row gives the mean for each method.

	MAE rank				RMSE rank			
	RF	SVR-ES	ANN	ELM-S	RF	SVR-ES	ANN	ELM-S
ENSO	1	4	2	3	4	3	1	1
SFBAY	2	4	1	3	3	4	1	2
WIND	4	2	1	3	4	3	1	2
FRASER	4	3	1	2	4	3	1	2
YVR	4	3	1	1	4	3	1	1
STAVE	4	1	2	2	4	2	3	1
PRECIP	4	1	3	1	4	1	3	1
TEMP	4	2	3	1	4	2	3	1
SO_2	1	2	4	3	1	2	4	3
Average	3.1	2.4	2.0	2.1	3.5	2.6	2.0	1.6

in all studied cases and SVR-ES starts to lose speed advantage to ANN when the size of the training set starts to grow. Figs. 2 and 3 and Tables 2 and 3 show that the SS of ELM-S were generally as good as (and in some cases better than) the SS of the SVR-ES. However ELM-S and SVR-ES only start to overcome the ANN SS when the size of the dataset is large (i.e. PRECIP, TEMP and SO_2).

6.2. 1-D search

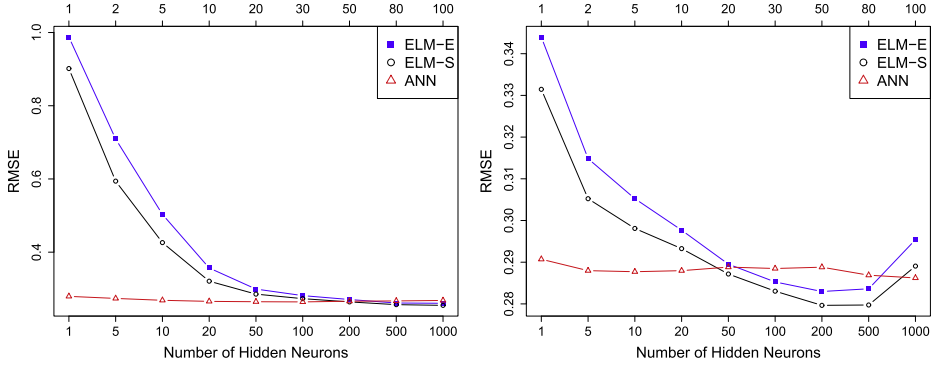
To check if the automated procedure used to find the number of HN performed satisfactorily, a separate set of experiments involving a 1-D search using all the predictors was performed to determine approximately the optimal number of HN for ELM-E, ELM-S and ANN. The tested cases were 1, 5, 10, 20, 50, 100, 200, 500, and 1000 HN for the ELM-E/ELM-S and 1, 2, 5, 10, 20, 30, 50, 80, and 100 HN for the ANN, with 30 ensemble members used in all three models.

As shown in Figs. 6 and 7, large differences in RMSE were present among the various choices for the number of HN. Also, the weight range for ELM had a noticeable effect, as seen in the difference in the RMSE between ELM-S and ELM-E. To achieve similar RMSE, ELM-E needs more HN than ELM-S. Similar to Parviainen and Riihimaki (2013), our results show that if the weight range was not carefully chosen a large number of HN could be necessary, with the increased network size (consequently network complexity) slowing down the computation. Thus, it is possible to save computational time and improve the RMSE by adjusting the range of the weights in advance using Eq. (6). However, as shown in Fig. 7, the computational time taken to train the models (ELM-S and ELM-E) is similar if they use the same number of HN.

For the ENSO dataset, the 1-D search found the optimal number of HN to be 10 for the ANN and between 10 and 20 for the ELM-S. Hence for ENSO, ELM-S was indeed faster than ANN using backpropagation and it had good generalization. The first impression is that the automated procedure used for the ANN was underestimating the number of HN (mean of 7 over 30 trials in Fig. 5), but for the ANN (using the 1-D search) the RMSE difference between 5 HN and 10 HN was only 0.1%. Since the difference in RMSE in this plateau region was small, the automated procedure for ANN worked well selecting the parsimonious models with good accuracy. A plateau is a flat region of the search space where the value returned by the target function is indistinguishable from the value returned for nearby regions. It can be a flat local maximum (or minimum), from which no uphill (downhill) exit exists, or a shoulder, from which progress is possible (Russell and Norvig, 2003). As outlined in Cannon and Whitfield (2002) the ANN plateau is due to stability from bagging, thus there is little penalty for selecting more HN than are strictly necessary. Fig. 5 and Table 4 also show the number of HN chosen by the hill climbing procedure for ELM-S, where the average number of HN over the 30 trials was 11, close to the best range of values (10–20) found by the 1-D search. Thus the hill climbing for ELM-S also worked well.

For SFBAY the 1-D search found the optimal number of HN to be 5 for the ANN and 20 for the ELM. For automated procedures the average number of HN was 6 for ANN and 17 for ELM. For WIND the 1-D search found as the optimal number of HN 2 for the ANN and 20 for the ELM, while the automated procedures selected 3 and 26 for the ANN and ELM, respectively. For FRASER the 1-D search found the optimal number of HN 10 for the ANN and 50 for the ELM, while the automated procedures selected 15 and 31 for the ANN and ELM, respectively. For YVR the 1-D search found as the optimal number of HN 10 for the ANN and 10 for the ELM, while the automated procedures selected 3 and 19 for the ANN and ELM, respectively. Again the impression of underestimation/overestimation is due to the plateau region. For STAVE, the 1-D search found the optimal number of HN to be 30 for the ANN and 500 for the ELM-S, versus 16 for ANN and 415 for ELM-S from the automated procedures.

For PRECIP (Fig. 6b), the 1-D search found the optimal number of HN to be around 5 for the ANN and identified a plateau between 200 and 500 for the ELM-S. Thus the automated procedures, which selected 3 and 282 HN for ANN and ELM-S, respectively, worked properly. For TEMP (Fig. 6a) the 1-D search found the optimal number of HN to be 30 for the ANN. We extended the search until 2000 HN (not shown in the figure) for the ELM-S and the wide



(a) TEMP (b) PRECIP

Fig. 6. RMSE of the training set for (a) TEMP and (b) PRECIP using a 1-D search over the number of HN. The number of HN used by the 1-D search for ELM-E/ELM-S is provided along the bottom axis and for ANN, along the top axis.

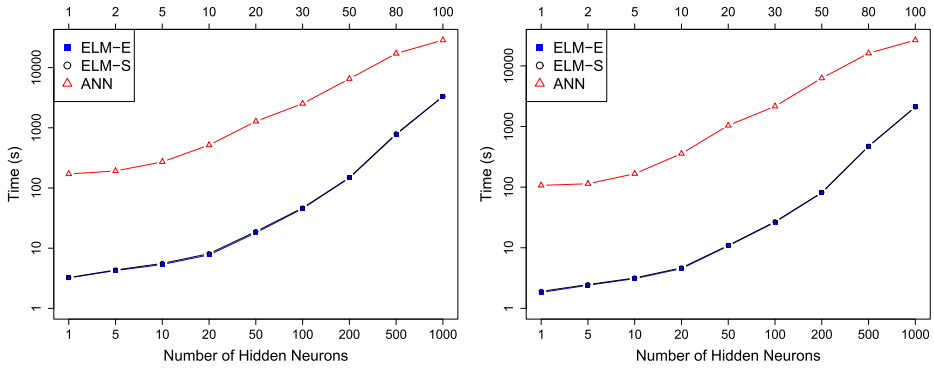


Fig. 7. Computational time (in seconds) versus the number of HN for (a) TEMP and (b) PRECIP plotted on a log scale. The number of HN used by the 1-D search for ELM-E/ELM-S is provided along the bottom axis and for ANN along the top axis. The ELM-E and ELM-S curves essentially overlapped in (a) and (b).

region between 500 and 1000 was where the optimal solution was located. The automated procedures selected 14 and 771 HN for ANN and ELM-S.

For SO_2 1000 HN was not enough to cover the necessary range of HN used by ELM-S, thus we extended the search until 2000. The difference between ELM-S using 1000 and 2000 was around 1%, again a plateau region. Basically with a large number of HN the susceptibility of the model to the increment of HN will decrease. Consequently, adding a few HN in models with a large number of HN will not make any difference because the variability will not be larger than that from using different initial random weights. As expected when the number of HN is high (Fig. 4, Table 4) the time necessary to train and predict with the models is also high. For this dataset ELM-S needed a complex structure with many HN to get the best result. For ANN the best number of HN found by the 1-D search was 30 (versus 13 from automated procedure), however the RMSE difference between 10 HN and 30 was less than 1%. Since the difference in RMSE in this plateau region was small, the automated procedure for ANN worked well, though none of the ANN (nor the ELM-S) results were able to beat the outstanding RF results.

Hill climbing algorithms have the advantage that they are often quick to identify a good solution to the problem, which is sometimes all that is required in practical applications. However, the downside is that if the problem exhibits numerous local optima significantly worse than the global optimum, no guarantees can be offered for the quality of solution found (Eiben and Smith, 2003). However, as the adjustment of the number of HN in the ELM-S often does not show numerous local optima (Fig. 6), the hill climbing algorithm is a practical heuristic to solve the problem.

6.3. ELM-S vs. ANN

In order to compare the ELM-S and ANN excluding the time used to find the optimal number of HN, a separate set of experiments was performed with the number of HN set to equal the average number of HN found earlier by the automated procedures (Table 4).

Similar to Section 6.1, data were standardized (zero mean and unit variance) and divided into a training set and an independent test set to test the trained model. Only one trial was performed, but an ensemble of 30 members was used for the ELM-S and ANN, and the averaged output of the individual members used as the output of the ensemble.

Table 5 shows the results which in general are similar to Section 6.1. Overall ANN has better MAE and RMSE for small datasets (ENSO, SFBAY, WIND and FRASER) and ELM-S better for large datasets (PRECIP, TEMP, SO_2). Except for SO_2, where ELM-S used a large number of HN (1440), the training of ELM-S was faster than the training of ANN. For testing, ANN was faster than ELM-S, due to the larger number of HN required by the ELM-S (1440 HN for SO_2 and 771 HN for TEMP). In practice, the testing time is insignificant compared to the training time, but the computer time needed for finding the optimal number of HN should not be ignored.

7. Summary and conclusion

In summary, we used nine environmental datasets to test four nonlinear prediction methods. As expected, no single method was best for every problem in every situation (Zhang, 2007; Elshorbagy et al., 2010b). All the tested nonlinear methods (RF, ELM-S, ANN, and SVR-ES) were suitable for the problems presented here, outperforming in most of the cases the linear method (MLR). In particular for the SFBAY, WIND, FRASER, STAVE and SO_2 datasets, the nonlinear methods performed much better than MLR.

It is possible to do a fine adjustment of the parameters to achieve better prediction skills for all the nonlinear methods presented. As mentioned in Zhang (2007), however, it is important to take into consideration the computational time required to identify the optimal parameters, especially if time consuming trial and error methods are used.

We used simple automated procedures to find the parameters and build the models. The proposed automated procedures performed satisfactorily and were able to find suitable parameters in a reasonable amount of time.

Excluding the large datasets, ELM-S tended to be the fastest among the nonlinear models, followed by RF. For the large datasets, RF tended to be the fastest. ELM-S tended to have higher skill scores than RF - e.g. for the RMSE SS (Fig. 3), ELM-S was surpassing RF except for the SO_2 dataset.

For smaller datasets, SVR-ES tended to be faster than ANN, but when the number of data points (cases) became large, the computing time needed to find the optimal parameters and to train the model exceeded the time taken by ANN.

Among the four nonlinear models, in terms of skill scores, ANN tended to be the best for the smaller datasets and ELM-S for the larger datasets. This could be due to the fact that ANN used bagging which gave it an advantage when overfitting is more of a concern for datasets with relatively few training data. For datasets with many training cases, overfitting is not an issue, and using bootstrap sampling in bagging meant ANN ensemble members were trained with fewer data points than the ELM-S ensemble members which did not use bootstrapping to leave out data during training.

The ELM algorithm is indeed a promising approach for a range of

prediction problems in the environmental sciences. The model is relatively simple to implement and can be very fast to train on small to medium sized datasets. Also, performance in terms of SS is comparable to the other nonlinear models. For small to medium datasets, ELM-S requires a moderate number of HN. However, it starts to lose the speed advantage when the datasets are large, with numerous data points and predictors. For large datasets, the optimal number of HN tends to be large and finding the optimal number of HN becomes costly computationally. However, ELM-S appeared to run efficiently for datasets with large number of data points but few predictors (YVR) or large number of predictors but few data points (WIND). Compared individually against ANN, ELM-S requires a larger number of HN to compensate for the randomness of the weights used in the HN.

As suggested by Parviainen and Riihimaki (2013) the range of the random weights in ELM should also be considered a parameter for achieving the best results. Scaling the weights by Eq. (6) in ELM-S, we were able to improve the SS of ELM-E in seven of the nine datasets (Fig. 1). Also, computational cost may be lowered by scaling the range of the weights in advance, which may lead to more parsimonious models (fewer HN required). As we chose a constant value of $a = 2$ in Eq. (6), further work on how to optimally scale the weights for ELM is warranted.

For future work, we would investigate the online sequential extreme learning machine (OSELM) (Liang et al., 2006), which is more suitable for real-time applications. Additionally, datasets with different numbers of predictors and data points, procedures to screen out irrelevant predictors, and other automated procedures to find parameters are worth further study. Using bagging with ELM-S should also be considered to improve stability and accuracy.

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada via a Discovery Grant and an Engage Grant to W. Hsieh. The authors are grateful to the reviewers for their valuable comments.

Appendix A. Datasets description

Appendix A.1. ENSO

The El Niño-Southern Oscillation (ENSO) is the dominant mode of inter-annual ocean-atmosphere variability in the equatorial Pacific. The global influence of ENSO plays a substantial role in shaping seasonal climate anomalies, e.g. in snowpack (Hsieh and Tang, 2001) and streamflow (Gobena et al., 2013) in western Canada. Four regional ENSO indexes have been defined based on sea-surface temperature (SST) anomalies in the equatorial Pacific: Niño 1 + 2 (0°–10° S, 80° W–90° W), Niño 3 (5° N–5° S, 90° W–150° W), Niño 3.4 (5° N–5° S, 120° W–170° W) and Niño 4 (5° N–5° S, 150° W–160° E). There is no universal single definition for El Niño events but it is suggested that El Niño occurs if 5-month running means of SST anomalies in the Niño 3.4 region exceed 0.4 °C for 6 months or more (Trenberth, 1997).

The Canadian Seasonal to Interannual Prediction System (CanSIPS), the real-time extension of the Coupled Historical Forecast Project (CHFP2), has been operational at Environment Canada's Canadian Meteorological Centre (CMC) since December 2011. It combines 12-month ensemble forecasts from two versions of the Canadian Climate Modelling and Analysis (CCCma) coupled global climate model (CanCM3 and the CanCM4) (Merryfield et al., 2013). We used eight CHFP2 predictors, namely the ensemble mean and the ensemble spread of Niño 1 + 2, 3, 3.4, and 4, to make seasonal ENSO forecasts. Starting from January of 1979, the predictand is the observed Niño 3.4 index at 5-month lead time.

Appendix A.2. SFBAY

In reviews of prediction and forecasting in water resources and environmental engineering using ANNs, Maier et al. (2010) and Wu et al. (2014) found that the vast majority of studies focused on flow prediction with very few applications to water quality. Ammonium concentration is an important component in the delicate estuarine ecosystems. Disturbances on its concentrations could inhibit other chemical reactions, affecting directly the ecosystems (Dugdale et al., 2007).

The SFBAY dataset consists of selected observations and variables (1993–2004) from U.S. Geological Survey water quality stations in south San Francisco Bay (Jassby and Cloern, 2014). In this dataset, only the cases not containing missing values were considered (632 of 23,207 cases). The predictand was ammonium concentration and the six predictors were depth, chlorophyll, dissolved oxygen, suspended particulate matter, salinity and water temperature, all at zero lead time. As the predictand has a right-skewed distribution, the logarithm transformation was applied first to improve the regression results (Cawley et al., 2007).

Appendix A.3. WIND

Interest in wind-generated electricity has grown during the last few years as wind power is considered a source of clean energy. In some electricity markets, wind has become a significant source of energy. Wind power forecasting is an important tool for efficiently operating power systems, as it helps in electricity trading, planning maintenance of the wind farms, and energy storage operations (Foley et al., 2012).

The data used here came from a wind farm consisting of 53 turbines located in northern Colorado. This dataset was originally used in the American Meteorological Society's 2012 Wind Power Prediction Contest and the objective was to predict the total wind power from the 53 turbines of a wind farm (predictand).

The predictors were forecast variables from three different numerical weather prediction (NWP) models: Global Forecast System (GFS), North American Mesoscale (NAM) and Weather Research & Forecasting (WRF). Due to differences in the resolution of the NWP models, forecasts were provided at four adjacent grid points for NAM and WRF, and at one grid point for GFS. The forecast variables were temperature, pressure, relative humidity, and the u and v components of wind velocity. These variables were provided at different levels in the vertical, with some at or near ground level, some at levels adjacent to the wind turbine hub height of 80 m above ground, and others provided as average quantities throughout the vertical layer surrounding the turbine hub.

As wind power is mainly proportional to the cube of the wind speed (Mathew, 2006), the u and v components were replaced by the cube of the wind speed. The goal was to make a 24-h forecast, which was done four times per day.

Appendix A.4. FRASER

The dataset FRASER, consists of daily observations of suspended sediment concentration (SSC) (mg L^{-1}) and stream discharge (Q) ($m^3 s^{-1}$) for the years 1970–1979 at the Fraser River at Hope station in British Columbia, Canada. The relationship between SSC and Q is nonlinear and time dependent (Cannon, 2012b).

To reduce the non-Gaussian nature of the distribution, the $\log_{10}$ transformation was applied to the predictand (the SSC at zero lead time) and the four predictors were logQ, dQ5, dQ30, and dQ90 (5-, 30-, and 90-day moving averages of the daily changes in Q).

Appendix A.5. YVR

The limited temporal and spatial scales of NWP models (Kuligowski and Barros, 1998) is one of the factors that contributes to the difficulty of modelling precipitation. To forecast precipitation at local scales, general circulation model (GCM) outputs cannot be used directly due to the coarse spatial resolution, so statistical downscaling is often performed (Cannon, 2011; Chen et al., 2010; Haylock et al., 2006).

The YVR dataset is a simple real-world precipitation downscaling dataset described by Cannon (2011). The predictand data consists of thirty years of daily precipitation (1971–2000) from the Vancouver International Airport YVR (WMO station 71,892, 49° 12.0′N, 123° 10.8′W).

Precipitation is an example of mixed distribution where the predictand is partly discrete (days with or without precipitation) and partly continuous (the amount of precipitation observed). Therefore precipitation is often modelled separately by occurrence (usually by a logistic regression model) and by amount (regression model fitted to the amount of precipitation on "wet" days) (Cawley et al., 2007; Chen et al., 2010). As our focus is to compare regression models, we used as predictand the precipitation amount on wet days (defined as days with precipitation >0.001 mm). However the nonlinear models presented here can also be used for classification (to separate wet and dry days). As the predictand was not normally distributed, we instead predicted the fourth root of the predictand, as commonly done for precipitation amounts (Khan et al., 2006).

The three predictors were sea-level pressure, 700-hPa specific humidity, and 500-hPa geopotential height from the nearest National Centers for Environmental Prediction/National Center for Atmospheric Research (NCEP/NCAR) Reanalysis (Kalnay et al., 1996) grid point at 50° N, 122.5° W.

Appendix A.6. STAVE

Short lead time streamflow forecasts based on recent and forecasted meteorological conditions, and hydrological observations are important for flood warning, hydroelectric power scheduling and regulating reservoir storage for optimum use of available water resources (Sene, 2010; Gobena et al., 2013).

This dataset was originally used by Rasouli et al. (2012). They applied GFS reforecast output, climate indexes and local meteo-hydrological observations to forecast daily streamflow (predictand) for the Stave River above Stave Lake in southern British Columbia, Canada at lead times of 1–7 days during 1983–2001. They found that local observations plus the GFS NWP output were generally best as predictors at shorter lead times (1–4 days), while local observations plus climate indexes were best at longer lead times (5–7 days).

For 1 day ahead prediction, we used 25 predictors, (maximum temperature, temperature range, precipitation, streamflow at day t, streamflow at day $t-1$, accumulated precipitation, heating, air pressure at 500 m, mean sea level pressure, precipitable water, relative humidity, 2 m air temperature, temperature, wind amplitude and wind phase at 10 m), same as Rasouli et al. (2012) but excluding the seasonal and climate indexes (i.e. we used only the local observations and the GFS outputs). We applied the logarithm transformation to the streamflow as it has a right-skewed distribution.

Appendix A.7. PRECIP

The next three datasets were originally used at the WCCI-2006 Predictive Uncertainty in Environmental Modelling Challenge (Cawley et al., 2007) and they are freely available from the challenge website (http://theoval.cmp.uea.ac.uk/~gcc/competition/). As the datasets were provided as part of a prediction contest, detailed descriptions of the predictors were not provided to entrants and are hence unknown.

In the PRECIP dataset the predictors supply large-scale circulation information, such as might be obtained from a GCM, and the predictand is the daily precipitation amount recorded at Newton Rigg, a relatively wet station located in the northwest of the United Kingdom. As above for YVR, we used only data from the wet days for the predictand and applied the fourth root transform to alleviate the non-Gaussian distribution. The PRECIP dataset contains 106 predictors.

Appendix A.8. TEMP

With similar large-scale circulation features to those used for the PRECIP dataset, the TEMP dataset, also with 106 predictors, is another statistical downscaling problem, but the predictand is the more normally distributed daily maximum temperature at the Writtle station in the southeast of the United Kingdom.

Appendix A.9. SO_2

Local emissions and meteorological conditions such as high atmospheric pressure and temperature inversions could lead to poor dispersion of atmospheric pollutants due to the stagnant conditions. The predictand is the concentration of sulphur dioxide (SO_2), an atmospheric pollutant, in urban Belfast. The SO_2 concentration is a right skewed variable with a heavy tail, thus the logarithm transformation was applied. Also only the days with SO_2 concentration >0.001 μg m^{-3} were considered as predictand. To predict the SO_2 concentration 24 h in advance, meteorological conditions and current SO_2 levels were supplied as 27 predictors.

References

Abrahart, R.J., Anctil, F., Coulibaly, P., Dawson, C.W., Mount, N.J., See, L.M., Shamseldin, A.Y., Solomatine, D.P., Toth, E., Wilby, R.L., 2012. Two decades of anarchy? emerging themes and outstanding challenges for neural network river forecasting. Prog. Phys. Geogr. 36, 480–513.

Aguilar-Martinez, S., Hsieh, W.W., 2009. Forecasts of tropical Pacific sea surface temperatures by neural networks and support vector regression. Int. J. Oceanogr. 2009, 13. Article ID 167239.

Breiman, L., 1996. Bagging predictors. Mach. Learn. 24, 123–140.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. Wadsworth Inc.

Cannon, A.J., 2011. Quantile regression neural networks: Implementation in R and application to precipitation downscaling. Comput. Geosci. 37, 1277–1284.

Cannon, A.J., 2012a. Monmlp: Monotone Multi-layer Perceptron Neural Network. http://CRAN.R-project.org/package=monmlp. R package version 1.1.2.

Cannon, A.J., 2012b. Neural networks for probabilistic environmental prediction: conditional density estimation network creation and evaluation (CaDENCE) in R. Comput. Geosci. 41, 126–135.

Cannon, A.J., Lord, E.R., 2000. Forecasting summertime surface-level ozone concentrations in the lower Fraser Valley of British Columbia: an ensemble neural network approach. J. Air Waste Manage. Assoc. 50, 322–339.

Cannon, A.J., Whitfield, P.H., 2002. Downscaling recent streamflow conditions in British Columbia, Canada using ensemble neural network models. J. Hydrol. 259, 136–151.

Cawley, G.C., Haylock, M., Dorling, S.R., Goodess, C., Jones, P.D., 2003. Statistical downscaling with artificial neural networks. In: Proceedings of the European Symposium on Artificial Neural Networks (ESANN-2003), pp. 167–172.

Cawley, G.C., Janacek, G.J., Haylock, M.R., Dorling, S.R., 2007. Predictive uncertainty in environmental modelling. Neural Netw. 20, 537–549.

Chen, S.T., Yu, P.S., Tang, Y.H., 2010. Statistical downscaling of daily precipitation using support vector machines and multivariate analysis. J. Hydrol. 385, 13–22.

Chen, Y., Chang, F.J., 2009. Evolutionary artificial neural networks for hydrological systems forecasting. J. Hydrol. 367, 125–137.

Cherkassky, V., Krasnopolsky, V., Solomatine, D., Valdes, J., 2006. Computational intelligence in earth sciences and environmental applications: issues and challenges. Neural Netw. 19, 113–121.

Cherkassky, V., Ma, Y., 2004. Practical selection of SVM parameters and noise estimation for SVM regression. Neural Netw. 17, 113–126.

Dugdale, R.C., Wilkerson, F.P., Hogue, V.E., Marchi, A., 2007. The role of ammonium and nitrate in spring bloom development in San Francisco Bay. Estuar. Coast. Shelf Sci. 73, 17–29.
Eiben, A.E., Smith, J.E., 2003. Introduction to Evolutionary Computing. In: Natural Computing Series. Springer, Berlin.
Elshorbagy, A., Corzo, G., Srinivasulu, S., Solomatine, D.P., 2010a. Experimental investigation of the predictive capabilities of data driven modeling techniques in hydrology – Part 1: concepts and methodology. Hydrol. Earth Syst. Sci. 14, 1931–1941.
Elshorbagy, A., Corzo, G., Srinivasulu, S., Solomatine, D.P., 2010b. Experimental investigation of the predictive capabilities of data driven modeling techniques in hydrology – Part 2: application. Hydrol. Earth Syst. Sci. 14, 1943–1961.
Foley, A.M., Leahy, P.G., Marvuglia, A., McKeogh, E.J., 2012. Current methods and advances in forecasting of wind power generation. Renew. Energy 37, 1–8.
Gobena, A.K., Weber, F.A., Fleming, S.W., 2013. The role of large-scale climate modes in regional streamflow variability and implications for water supply forecasting: a case study of the Canadian Columbia River Basin. Atmos. Ocean 51, 380–391.
Guorui, F., Guang-Bin, H., Qingping, L., Gay, R., 2009. Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans. Neural Netw. 20, 1352–1357.
Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, second ed. In: Springer Series in Statistics. Springer.
Haykin, S., 1998. Neural Networks – a Comprehensive Foundation, second ed. Pearson Education.
Haylock, M.R., Cawley, G.C., Harpham, C., Wilby, R.L., Goodess, C.M., 2006. Downscaling heavy precipitation over the United Kingdom: a comparison of dynamical and statistical methods and their future scenarios. Int. J. Climatol. 26, 1397–1415.
Hsieh, W.W., 2009. Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels. Cambridge University Press, New York, NY, USA.
Hsieh, W.W., Tang, B., 2001. Interannual variability of accumulated snow in the Columbia Basin, British Columbia. Water Resour. Res. 37, 1753–1759.
Huang, G.B., Wang, D., Lan, Y., 2011. Extreme learning machines: a survey. Int. J. Mach. Learn. Cybern. 2, 107–122.
Huang, G.B., Zhou, H., Ding, X., Zhang, R., 2012. Extreme learning machine for regression and multiclass classification. Syst. Man, Cybern. Part B IEEE Trans. Cybern. 42, 513–529.
Huang, G.B., Zhu, Q.Y., Siew, C.K., 2006. Extreme learning machine: theory and applications. Neurocomputing 70, 489–501. Neural Networks: Selected Papers from the 7th Brazilian Symposium on Neural Networks (SBRN '04).
Ibarra-Berastegi, G., Saénz, J., Ezcurra, A., Elías, A., Diaz de Argandoña, J., Errasti, I., 2011. Downscaling of surface moisture flux and precipitation in the Ebro Valley (Spain) using analogues and analogues followed by random forests and multiple linear regression. Hydrol. Earth Syst. Sci. Discuss. 8, 1951–1985.
Jassby, A.D., Cloern, J.E., 2014. wq: Exploring Water Quality Monitoring Data. http://CRAN.R-project.org/package=wq. R package version 0.4-1.
Kalnay, E., Kanamitsu, M., Kistler, R., Collins, W., Deaven, D., Gandin, L., Iredell, M., Saha, S., White, G., Woollen, J., Zhu, Y., Leetmaa, A., Reynolds, R., Chelliah, M., Ebisuzaki, W., Higgins, W., Janowiak, J., Mo, K.C., Ropelewski, C., Wang, J., Jenne, R., Joseph, D., 1996. The NCEP/NCAR 40-Year reanalysis project. Bull. Am. Meteorol. Soc. 77, 437–471.
Khan, M.S., Coulibaly, P., Dibike, Y., 2006. Uncertainty analysis of statistical downscaling methods. J. Hydrol. 319, 357–382.
Krasnopolsky, V.M., 2007. Neural network emulations for complex multidimensional geophysical mappings: applications of neural network techniques to atmospheric and oceanic satellite retrievals and numerical modeling. Rev. Geophys. 45.
Kuligowski, R.J., Barros, A.P., 1998. Localized precipitation forecasts from a numerical weather prediction model using artificial neural networks. Weather Forecast. 13, 1194–1204.
Kusiak, A., Zheng, H., Song, Z., 2009. Wind farm power prediction: a data-mining approach. Wind Energy 12, 275–293.
Leung, F., Lam, H., Ling, S., Tam, P., 2003. Tuning of the structure and parameters of a neural network using an improved genetic algorithm. IEEE Trans. Neural Netw. 12, 79–88.
Liang, N.Y., Huang, G.B., Saratchandran, P., Sundararajan, N., 2006. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. 17, 1411–1423.
Liaw, A., Wiener, M., 2002. Classification and regression by randomForest. R News 2, 18–22.
Lima, A.R., Cannon, A.J., Hsieh, W.W., 2013. Nonlinear regression in environmental sciences by support vector machines combined with evolutionary strategy. Comput. Geosci. 50, 136–144.
Liu, N., Wang, H., 2010. Ensemble based extreme learning machine. Signal Process. Lett. IEEE 17, 754–757.
Liu, X., Gao, C., Li, P., 2012. A comparative analysis of support vector machines and extreme learning machines. Neural Netw. 33, 58–66.
Maier, H.R., Dandy, G.C., 2000. Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. Environ. Model. Softw. 15, 101–124.
Maier, H.R., Jain, A., Dandy, G.C., Sudheer, K., 2010. Methods used for the development of neural networks for the prediction of water resource variables in river systems: current status and future directions. Environ. Model. Softw. 25, 891–909.
Mathew, S., 2006. Wind Energy: Fundamentals, Resource Analysis and Economics. Springer.
Merryfield, W.J., Lee, W.S., Boer, G.J., Kharin, V.V., Scinocca, J.F., Flato, G.M., Ajayamohan, R.S., Fyfe, J.C., Tang, Y., Polavarapu, S., 2013. The Canadian seasonal to interannual prediction system. Part I: models and initialization. Mon. Weather Rev. 141, 2910–2945.
Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., 2014. e1071: Misc Functions of the Department of Statistics (e1071). TU Wien. http://CRAN.R-project.org/package=e1071. R package version 1.6-4.
Parviainen, E., Riihimaki, J., 2013. A connection between extreme learning machine and neural network kernel. In: Fred, A., Dietz, J., Liu, K., Filipe, J. (Eds.), Knowledge Discovery, Knowledge Engineering and Knowledge Management, Volume 272 of Communications in Computer and Information Science. Springer, Berlin Heidelberg, pp. 122–135.
Piotrowski, A.P., Napiorkowski, J.J., 2011. Optimizing neural networks for river flow forecasting - evolutionary computation methods versus the Levenberg-Marquardt approach. J. Hydrol. 407, 12–27.
R Core Team, 2014. R: a Language and Environment for Statistical Computing. http://www.R-project.org/.
Rasouli, K., Hsieh, W.W., Cannon, A.J., 2012. Daily streamflow forecasting by machine learning methods with weather and climate inputs. J. Hydrol. 414–415, 284–293.
Russell, S.J., Norvig, P., 2003. Artificial Intelligence: a Modern Approach. Pearson Education.
Schmidt, W.F., Kraaijveld, M., Duin, R.P., 1992. Feedforward neural networks with random weights. In: Pattern Recognition, 1992. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on, vol. II. IEEE, pp. 1–4.
Schoof, J., Pryor, S., 2001. Downscaling temperature and precipitation: a comparison of regression-based methods and artificial neural networks. Int. J. Climatol. 21, 773–790.
Sene, K., 2010. Hydrometeorology: Forecasting and Applications. Springer.
Solomatine, D.P., Ostfeld, A., 2008. Data-driven modelling: some past experiences and new approaches. J. Hydroinf. 10, 322.
Sun, Z.L., Choi, T.M., Au, K.F., Yu, Y., 2008. Sales forecasting using extreme learning machine with applications in fashion retailing. Decis. Support Syst. 46, 411–419.
Thimm, G., Fiesler, E., 1997. High-order and multilayer perceptron initialization. IEEE Trans. Neural Netw. 8, 349–359.
Trenberth, K.E., 1997. The definition of El Niño. Bull. Am. Meteorol. Soc. 78, 2771–2777.
Tripathi, S., Srinivas, V., Nanjundiah, R.S., 2006. Downscaling of precipitation for climate change scenarios: a support vector machine approach. J. Hydrol. 330, 621–640.
Venables, W., Ripley, B., 2002. Modern Applied Statistics with S. Statistics and Computing. Springer.
Wu, W., Dandy, G.C., Maier, H.R., 2014. Protocol for developing ANN models and its application to the assessment of the quality of the ANN model development process in drinking water quality modelling. Environ. Model. Softw. 54, 108–127.
Yuret, D., 1994. From Genetic Algorithms to Efficient Optimization. Master's thesis. Massachusetts Institute of Technology.
Zhang, G.P., 2007. Avoiding pitfalls in neural network research. IEEE Trans. Syst. Man Cybern. Part C 37, 3–16.