Available online at www.sciencedirect.com
SCIENCE DIRECT
Neural Networks 19 (2006) 225–235
Neural Networks
www.elsevier.com/locate/neunet

2006 Special issue

Machine learning approaches for estimation of prediction interval for the model output

Durga L. Shrestha *, Dimitri P. Solomatine

Department of Hydroinformatics and Knowledge Management, UNESCO-IHE Institute for Water Education, P.O. Box 3015, 2601 DA Delft, The Netherlands

Abstract

A novel method for estimating prediction uncertainty using machine learning techniques is presented. Uncertainty is expressed in the form of the two quantiles (constituting the prediction interval) of the underlying distribution of prediction errors. The idea is to partition the input space into different zones or clusters having similar model errors using fuzzy *c*-means clustering. The prediction interval is constructed for each cluster on the basis of empirical distributions of the errors associated with all instances belonging to the cluster under consideration and propagated from each cluster to the examples according to their membership grades in each cluster. Then a regression model is built for in-sample data using computed prediction limits as targets, and finally, this model is applied to estimate the prediction intervals (limits) for out-of-sample data. The method was tested on artificial and real hydrologic data sets using various machine learning techniques. Preliminary results show that the method is superior to other methods estimating the prediction interval. A new method for evaluating performance for estimating prediction interval is proposed as well.

Keywords: Prediction interval; Model uncertainty; Prediction interval coverage probability; Mean prediction interval

1. Introduction

In solving the problems of model-based forecasting of natural phenomena, for example, in hydrologic forecasting for flood management, decision makers require not only accurate forecasting of certain variables but also the uncertainty estimates associated with the forecasts. In weather prediction, the forecasts are typically given together with the associated uncertainty, but in other areas, e.g. in water management the prevailing format of forecasting is deterministic, i.e. in the form of a point forecast. It does not take into account the various sources of uncertainties including model uncertainty, input uncertainty and parameter uncertainty. Incorporating prediction uncertainty into deterministic forecasts helps to enhance the reliability and credibility of the model outputs.

Uncertainty of the model output can be estimated using several approaches. The first approach is to forecast the model outputs probabilistically and it is often used in hydrological modeling (Krzysztofowicz, 2000). The second approach is to estimate uncertainty by analyzing the statistical properties of the model errors that occurred in reproducing the observed historical data. This approach has been widely used in statistical (Harnett & Murphy, 1980; Wonnacott & Wonnacott, 1996) and machine learning communities (Heskes, 1997; Nix & Weigend, 1994; Penny & Roberts, 1997) for time series forecasting; uncertainty is estimated in terms of confidence interval or prediction interval. The third approach is to use simulation and re-sampling based techniques, generally referred to as ensemble, or Monte Carlo methods (one of the versions of such approach used in hydrologic modeling is a generalized likelihood uncertainty estimator, GLUE (Beven & Binely, 1992)). The fourth approach is based on fuzzy theory based method (Maskey, Guinot, & Price, 2004). This approach provides a non-probabilistic approach for modeling uncertainty.

The first and the third approaches mentioned require the prior distributions of the uncertain input parameters or data to be propagated through the model to the outputs. While the second approach requires certain assumptions about the data and the errors and obviously the relevancy and accuracy of such approach depends on the validity of these assumptions. The last approach requires knowledge of the membership function of the quantity subject to the uncertainty.

This paper presents a novel approach to estimating uncertainty of the model output using machine learning

* Corresponding author. Tel.: +31 15 2151 764.
E-mail address: d.shrestha@unesco-ihe.org (D.L. Shrestha).

0893-6080/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2006.01.012

techniques. In this paper, uncertainty is expressed in the form of two quantiles (constituting the prediction interval) of the underlying distribution of model errors. The idea is to partition the input space into different zones or clusters having similar model errors using fuzzy *c*-means clustering. Prediction interval (PI) is constructed for each cluster on the basis of empirical distributions of the errors associated with all instances belonging to the cluster under consideration. PI for each instance in input space is then calculated according to the grade of their memberships in each cluster. Different machine learning models are trained to estimate the underlying functional relationship between the input variables and the computed prediction intervals (limits) for in-sample data (training data), and then applied to estimate prediction interval for the out-of-sample data (validation data).

The proposed method is employed to estimate the prediction limit or interval using linear regression and three machine learning techniques—locally weighted regression, artificial neural networks, and model trees—for artificial and hydrological data sets. The performance of the novel method is compared to other methods.

2. Prediction interval

An interval forecast is usually comprised of the upper and lower limits between which a future unknown value (e.g. a point forecast) is expected to lie with a prescribed probability. This limit is called prediction limit or bound, while the interval is called the prediction interval (PI) (see Fig. 1). The prescribed probability is called confidence level. PI should be distinguished from confidence interval (CI). The CI is the term relating to the accuracy of our estimate of the true regression (statisticians formulate this differently—as estimation of the mean value of the outputs (Harnett & Murphy, 1980)). In contrast, the PI deals with the accuracy of our estimate with respect to the observed target value. It should be noted that the PI is wider than the CI (Heskes, 1997). In real world applications, PI is of more practical use than CI because prediction interval is concerned with the accuracy with which we can predict the observed target value itself, but not just the accuracy of our estimate of the true regression.

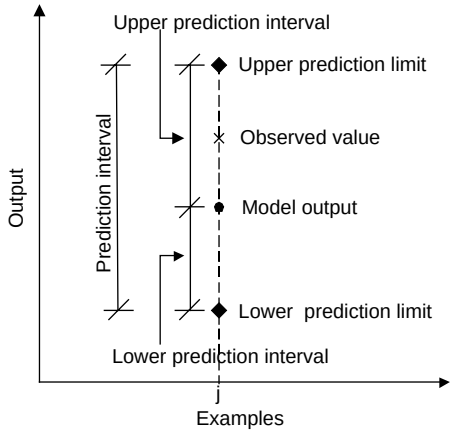

Fig. 1. Terminology used in the paper.

The following sub-sections briefly present the methods for constructing PI for the model outputs.

2.1. Prediction interval for linear regression

In regression problems, the task is to estimate an unknown function $f(\mathbf{x};\theta)$ given a set of input-target pairs $D=\{x_i, t_i\}$, $i=1,\ldots,n$, where θ is the true values of the set of parameters of the regression model and n is number of data. The function $f(\mathbf{x};\theta)$ is related with the true target t by Eq. (1)

$$t_i = f(\mathbf{x}_i;\theta) + e_i \tag{1}$$

where e_i is the model error and is assumed to be independently and identically (iid) distributed with variance σ^2 and the distribution has the form $N(0, \sigma^2)$. The least square estimate of parameters θ is $\hat{\theta}$, which is obtained by minimizing the cost function

$$C(\theta) = \sum_{i=1}^{n} [t_i - y_i]^2 \tag{2}$$

$$y_i = f(\mathbf{x}_i;\hat{\theta}) \tag{3}$$

where y_i is the model output. The model output may not necessarily match the target t due to various reasons including the presence of noise in the data, limited number of data points, non-linear relationship between the dependent and independent variables, and the errors in estimating the parameters. In order to simplify the formulation, it is assumed that the regression model is linear and univariate; so Eq. (3) can be rewritten as

$$y_i = \hat{a}x_i + \hat{b} \tag{4}$$

where $\hat{a}$ and $\hat{b}$ are estimated parameters of the linear regression.

Most of the methods to construct $100(1-\alpha)\%$ prediction limit for the model output typically assume the error has Gaussian distribution with zero mean and σ standard deviation and are

$$\begin{aligned} \mathrm{PL}^U &= y + z_{\alpha/2}\sigma \\ \mathrm{PL}^L &= y + z_{\alpha/2}\sigma \end{aligned} \tag{5}$$

where PL^U and PL^L are the upper and lower prediction limits, respectively, $z_{\alpha/2}$ is the value of the standard normal variate with cumulative probability level of $\alpha/2$. Since the prediction limits in Eq. (5) are symmetric about y, it is assumed that the prediction is unbiased. Consequently, the model error variance σ^2 equals the mean squared error (MSE), often called the prediction mean squared error (PMSE) (Chatfield, 2000). However, σ^2 is not known in practice and is estimated from the data. An unbiased estimate of σ^2 with $n-p$ degrees of freedom, denoted by s^2, is given by the formula

$$s^2 = \mathrm{SSE}/(n-2) = \frac{1}{n-2}\sum_{i=1}^{n}(t_i - y_i)^2 \tag{6}$$

where p is number of parameters in the model and SSE is the sum squared error.

If the error variance s^2 is not constant in the output space, then Eq. (6) should be modified as follows (Harnett & Murphy,

1980; Wonnacott & Wonnacott, 1996)

$$s_{y_i}^2 = s^2(1 + 1/n + (x_i - \bar{x})^2/(n-1)s_x^2) \quad (7)$$

where s_x^2 and $\bar{x}$ are the sample variance and mean, respectively. It can be seen that the error variance for the output y_i is always larger than s^2 and it is dependent on how far x_i is away from $\bar{x}$. For multivariate linear regression (7) can be modified as

$$s_{y_i}^2 = s^2(1 + \mathbf{x}_i^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}\mathbf{x}_i) \quad (8)$$

where X is a matrix of input space, appended to a column of 1's as the leftmost column and $\mathbf{x}_i$ is ith row of matrix X. In this paper, this method will be referred to as linear regression variance estimator (LRVE) method.

2.2. Prediction interval for non-linear regression

Derivation of the error variance s^2, and hence computation of the prediction interval is not so easy in most practical situations, especially for multivariate models containing many equations or depending on non-linear relationship (e.g. neural networks). In order to simplify the computation, most of the techniques to compute prediction interval for neural network models including the technique proposed by Chryssolouris, Lee, and Ramsy (1996), typically make a strong assumption of constant variance of data in the input space. On the other hand, locally generalizing networks such as radial basis function networks have been extended to estimate the prediction limits on neural network predictions (e.g. Leonard, Kramer, & Ungar, 1992). Chinnam and Ding (1998) extended the concept of local neighborhoods for global and local approximating neural networks to estimate the prediction limits using self-organizing feature maps (SOFM).

Re-sampling based techniques have been also reported in the literature to estimate s^2 and thus to compute the PI for the neural networks model (e.g. Heskes, 1997; Nix & Weigend, 1994; Penny & Roberts, 1997; Tibshirani, 1996). Typically, these techniques are based on the premise that the error variance s^2 can be decomposed into three terms: model bias, model variance, and target noise (Geman, Bienenstock, & Doursat, 1992). Model variance can be estimated by an ensemble of networks, while the target noise is estimated by training an auxiliary network on the residuals of the committee predictions. Most of such techniques neglect the contribution of the model bias to the total error variance. These techniques are generally computationally time consuming, as the ensemble should include many networks to ensure a reliable estimate.

3. Machine learning techniques

A machine learning technique is an algorithm that estimates (induces) a hitherto unknown mapping (or dependency) between a system's inputs and its outputs from the available data (Mitchell, 1998). As such a dependency is discovered, it can be used to predict (or effectively deduce) the future system's output from the known input values.

Machine learning technique, on the basis of input data, tries to identify ('learn') the target function describing how the real system behaves. *Learning* (or *training*) here is the process of minimizing the difference between observed data and model output.

In this paper, we used artificial neural networks (ANN), locally weighted regression (LWR) and M5 model trees. A brief description of the two latter follows.

3.1. M5 model trees

A model tree (MT) is similar to decision tree, but includes the multivariate linear regression functions at the leaves and able to predict continuous numeric values attributes. Thus, model tree is analogous to a piecewise linear function. Model tree learns efficiently and can tackle tasks with very high dimensionality—up to hundreds of attributes. As compared to other machine learning techniques, model tree learning is fast and the results are interpretable. Furthermore, it is very easy to implement model tree algorithms, which can handle large number of attributes.

The algorithm known as M5 algorithm is used for inducing model tree (Quinlan, 1992), which works as follows. Suppose that a collection T of training examples is available. Each example is specified by the values of a fixed set of input attributes; and has an associated target (output) value. The aim is to construct a model that relates a target value of the training example to the values of their input attributes.

Tree-based models are constructed by divide-and-conquer method. The set T is either associated with a leaf, or some test is chosen that splits T into subsets corresponding to the test outcomes and the same process is applied recursively to the subsets. The splitting criterion for the M5 model tree algorithm is based on treating the standard deviation of the class values that reach a node as a measure of the error at that node, and calculating the expected reduction in error as a result of testing each attribute at that node.

After examining all possible splits, M5 chooses the one that maximizes the expected error reduction. Splitting in M5 ceases when the class values of all the instances that reach a node vary very slightly, or only a few instances remain. Such division often results in the trees with the many layers that must be reduced, for instance by replacing a subtree with a leaf. In the final stage, 'smoothing' is performed to compensate for the discontinuities that occur between the adjacent linear models at the leaves of the pruned tree. Details of this process can be found in Quinlan (1992) and Witten and Frank (2000) and in a paper in this issue (Bhattacharya & Solomatine, 2006). An approach of building model trees in a non-greedy fashion was proposed by Solomatine and Siek (2004, 2006).

3.2. Locally weighted regression

Locally weighted regression (LWR) is inspired by the instance-based methods. Like instance-based learning,

it performs all 'learning' at prediction time. Although LWR resembles model trees in that uses linear regression to fit models locally to particular areas of instance space, it does so in quite a different way. LWR generates local models at prediction time by giving higher weight to instances in the neighborhood of the particular instance. More specifically, it weights the training instances according to their distance to the test instance and builds a linear regression on the weighted data. Training instances close to the test instance receive a higher weight and those far away—a lower one.

In order to use LWR, it is required to decide on a distance-based weighting scheme for the training instances. A common choice is to weight the instances according to the inverse of their Euclidean distance from the test instance. Another possibility is to use the Euclidean distance in conjunction with a Gaussian kernel function. However, there is no clear evidence that the choice of weighting function is critical. More important is the selection of a 'smoothing parameter' that is used to scale the distance function—the distance is multiplied by the inverse of this parameter. If it is set to a small value, only instances very close to the test instance will receive significant impact on the model.

Like model trees, locally weighted linear regression is able to approximate non-linear functions. One of its main advantages is that it is ideally suited for incremental learning: all training is done at prediction time; so new instances can be added to the training data at any time. However, like all instance-based methods, it is slow at deriving a prediction for a new instance. First, the training instances must be scanned to compute their weights; then a weighted linear regression is performed on these instances. Also, like other instance-based methods, locally weighted regression provides little information about the global structure of the training data set. More details about locally weighted regression can be found in Atkeson, Moore, and Schaal (1997).

4. Background of fuzzy *c*-means clustering

Clustering involves the task of partitioning a data set into a number of homogenous clusters with respect to a suitable similarly measure. Due to the fuzzy nature of many practical problems, a number of fuzzy clustering methods have been reported in the literature following the general fuzzy set theory outlined by Zadeh (1965). In the traditional hard clustering (e.g. *k*-means), each data point is assumed to be in exactly one cluster. This condition can be relaxed and allow for each instance to belong to a cluster with some degree, interpreted as a 'fuzzy' membership in a cluster, so that a point may belong to several clusters with some degree in the range [0, 1]. Usually membership functions, which indicate the degree to which the data points belong to the different clusters, are determined based on a distance function.

The most known method of fuzzy clustering is the fuzzy *c*-means (FCM), initially proposed by Dunn (1973) and generalized by Bezdek (1981). FCM algorithm attempts to partition a set of given data $X=\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ where each data point $\mathbf{x}_k$ is a vector in R^p. The fuzzy partitioned space M_{fc} takes the form of

$$M_{\mathrm{fc}} = \left\{ U_{i,j} \in R^{c\times n} : \begin{array}{ll} \mu_{i,j} \in [0,1] & \forall\, i,j \\ \sum_{i=1}^{c} \mu_{i,j} = 1 & \forall\, j \\ 0 < \sum_{j=1}^{n} \mu_{i,j} < n & \forall\, i \end{array} \right\} \tag{9}$$

where c is the number of clusters, n is the number of data points, U is the partition matrix that contains the membership values $\mu_{i,j}$ of each data j for each cluster i. The aim of the FCM algorithm is to find an optimal fuzzy c-means partition that minimizes the following objective function

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{j=1}^{n} \mu_{i,j}^{m} \|x_j - v_i\|^2 \tag{10}$$

where $V=(v_1,\ldots, v_c)$ is a matrix of unknown cluster centers, $\|\cdot\|$is the Euclidean norm and the weighting exponent m in [1, ∞] is a constant that influences the membership values.

One of the problems of most clustering algorithms is that it is usually necessary to assume first a certain structure to cluster the data set. Another problem is the difficulty in determining the optimal number of clusters. Different validity measures are reported in the literatures such as the Bezdek's partition coefficient (Bezdek, 1981), the Xie–Beni's separation index (Xie & Beni, 1991). The latter is given by:

$$S = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n} \mu_{i,j}^{2} \|x_j - v_i\|^2}{n \min_{i\neq j} \|v_i - v_j\|^2} \tag{11}$$

Separation index S aims to quantify the ratio of the total variation within clusters and the separation of clusters. Choosing the correct number of clusters should minimize this index.

5. Methodology

LRVE approach requires many assumptions (often unrealistic and difficult to check) to use Eq. (7) for constructing PI for linear regression. Furthermore, this method is difficult to use in case of non-linear or multivariate models containing many equations. For non-linear problems, there are methods based on re-sampling whose computational cost is, however, very high. This section describes a new method for estimating uncertainty (prediction interval) of model outputs.

Due to the various sources of uncertainty as mentioned in the Section 1, it is not surprising that the model outputs do not match the observed values well. The proposed method is based on an idea that the historical residuals (errors) between the model outputs and the corresponding observed data are the best available quantitative indicators of the discrepancy between the model and the modeled real world system or process, and give the valuable information that can be used to assess the model uncertainty. These residuals are often functions of the model input's values and can be modeled.

The input dataset can be partitioned into the n clusters corresponding to different values or distributions of historical residuals. It can be assumed that the region in the input space that is associated with any particular cluster has similar residuals or residuals with similar distributions. Having identified the clusters, the PIs for each cluster are computed from empirical distributions of the corresponding historical residuals. For instance, in order to construct $100(1-\alpha)\%$ PI, the $(\alpha/2)\times 100$ and $(1-\alpha/2)\times 100$ percentile values are taken from empirical distribution of residuals for lower and upper prediction interval, respectively. Typical value for α is 0.05, which corresponds to 95% predicton interval. If the input space is divided into crisp clusters, e.g. by k-means clustering, and each instance belongs to exactly one cluster, this computation is straightforward. However, in the case of fuzzy clustering, e.g. by fuzzy c-means method, where each instance belongs to more than one cluster and is associated with several membership grades, the computation of the above percentiles should take this into account. To calculate PI, the instances should first be sorted with respect to the corresponding errors in ascending order. The following expression gives the lower prediction interval (see Fig. 1 for terminology) for cluster i (PIC^L)

$$\mathrm{PIC}_i^L = e_j \quad j{:}\sum_{k=1}^{j}\mu_{i,k} < \alpha/2\sum_{j=1}^{n}\mu_{i,j} \tag{12}$$

where j is the maximum value of it that satisfies the above inequality, e_j is the error (residual) associated with the instance j (instances are sorted), $\mu_{i,j}$ is the membership grade of jth instance to cluster i. Similar type of expression can be obtained for the upper PI (PIC^U). This is illustrated in Fig. 2.

Once the PI is computed for each cluster, the PI for each instance in input space can be computed; note that this computation also depends upon the clustering technique employed. For example, if crisp clustering is employed, then the PI for each instance in the particular cluster is the same as that of the cluster. However, in the case of fuzzy clustering, the so called 'fuzzy committee' approach (Solomatine & Siek, 2006) is used and PI is computed using the weighted mean of PI of each cluster as

$$\mathrm{PI}_j^L = \sum_{i=1}^{c}\mu_{i,j}\mathrm{PIC}_i^L \quad \mathrm{PI}_j^U = \sum_{i=1}^{c}\mu_{i,j}\mathrm{PIC}_i^U \tag{13}$$

where PI_j^L and PI_j^U are the lower and upper prediction interval for jth instance, respectively. Once the lower and upper prediction interval for each input instance is obtained, prediction limits are computed by simply adding model output to them

$$\mathrm{PL}_j^L = y_j + \mathrm{PI}_j^L \quad \mathrm{PL}_j^U = y_j + \mathrm{PI}_j^U \tag{14}$$

where PL_j^L and PL_j^U are the lower and upper prediction limits for jth instance, respectively. Having these, it is possible to construct two independent mapping functions that estimate an underlying functional relationship between an input $\mathbf{x}$ and the computed prediction limits as

$$\mathrm{PL}^L = f_U^L(\mathbf{x};\theta^L) \quad \mathrm{PL}^U = f_U^U(\mathbf{x};\theta^U) \tag{15}$$

where the mapping functions $f_U^L(\cdot)$ and $f_U^U(\cdot)$ estimate the lower and upper prediction limits, respectively, θ^{L} and θ^{U} are the parameters of the functions. Note that mapping functions can take any form, from linear regression to non-linear functions such as neural networks. For example, given a set of n data pairs $\{\mathbf{x}_j, \mathrm{PL}_j^L\}, j = 1,\ldots,n$, we can train neural networks to estimate an underlying function relating input data $\mathbf{x}$ to the computed lower prediction limit. It is worthwhile to mention that the target variable of mapping function might be either the prediction interval or prediction limit. The mapping function $f_{\mathrm{U}}(\cdot)$ will be further referred to as the local uncertainty estimation model (LUEM).

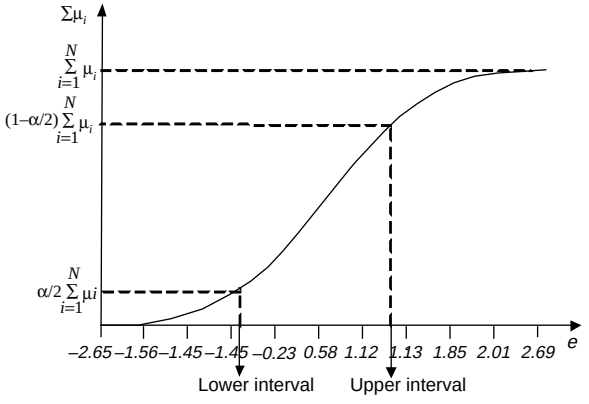

Fig. 2. Computation of the prediction interval in case of using fuzzy c-means clustering.

6. Validation

It is often difficult to validate LUEM due to unavailability of the necessary data. The reason is that having the prediction limits as targets in the in-sample data while training the LUEM, we do not have them in the out-of-sample (test) data to evaluate the performance of such model. However, the test data set, which is used to evaluate the generalization of the model predicting target t, can be also used to evaluate the performance of LUEM. Since the target values of the prediction limits are not present in this test data set, the performance of LUEM cannot be measured using classical residuals based (e.g. root mean squared error) measure. A possible solution is to split the training data used to predict the target t, into two parts. The first part will be used to train the LUEM and the second part to evaluate the generalization. It is worthwhile to point out that such split is not always possible when data is scarce.

Even though target values of the prediction limit are not present in the test data set, it is interesting to know whether

target values t are inside the estimated prediction limits. Since, by definition, the prediction limits enclose the true but unknown value $(1-\alpha)\%$ of times on average (typically 95%), we assess the performance of LUEM by evaluating the prediction interval coverage probability (PICP). The PICP is the probability that the target of an input pattern lies within the prediction limits and is estimated by the corresponding frequency as follows

$$\text{PICP} = \frac{1}{V}\text{count } j \qquad j : \text{PL}_j^L \le t_j \le \text{PL}_j^U \tag{16}$$

where V is number of data in the test set. If the clustering technique and the LUEM are optimal, then the PICP value will be consistently close to the $(1-\alpha)\%$.

Yet another performance measure for the prediction limit can be introduced as well. This is the mean prediction interval (MPI) calculated across all points in the test data set. It measures the ability to enclose target values inside the prediction bounds and can be estimated by:

$$\text{MPI} = \frac{1}{V}\sum_{j=1}^{V}\left[\text{PL}_j^U - \text{PL}_j^L\right] \tag{17}$$

7. Experimental set up

7.1. Data sets

The method was tested on both artificial and real hydrologic data sets. The artificial data is generated using bivariate input variables $\mathbf{x}$ uniformly distributed between [0, 1]. The true targets t_t are given by:

$$t_t = f_t(\mathbf{x}) \tag{18}$$

More specifically in our case

$$t_t = b_1 x_1 + b_2 x_2 \tag{19}$$

where b_1 and b_2 are linear regression coefficients and arbitrary values of these coefficients: 12 and 7 were used in the experiments. Since the noise is inherent to any real data set, we also added additive noise ε to t_t to obtain new target t and it is estimated by the function f_f as:

$$t = t_t + \varepsilon = f_f(\mathbf{x}) \tag{20}$$

The noise ε had Gaussian distribution $N(0, \sigma_t/\text{SNR})$. SNR is signal to noise ratio defined by

$$\text{SNR} = \sigma_t/\sigma_\varepsilon \tag{21}$$

where σ_t and σ_ε are the standard deviations of the target t and noise ε, respectively. We varied the noise level by taking the following SNR values: 1, 3, 5 and 7. Total of 1500 instances were generated, and the two-thirds of these selected randomly constituted the training data set, and the rest—the test data set.

The hydrologic data sets used in this experiment related to the river flows prediction in the Sieve catchment in Italy (Solomatine & Dulal, 2003; Solomatine & Shrestha, 2004). Prediction of river flows Q_{t+i} several hours ahead ($i = 1, 3$ or 6) is based on using the previous values of flow ($Q_{t-\tau q}$) and previous values of rainfall ($\text{Re}_{t-\tau r}$), where τq being between 0 and 2 h and τr being between 0 and 5 h. The sought regression models were based on 1854 examples and were formulated as follows:

$$\text{SieveQi model} \rightarrow Q_{t+i} = f_f\left(\{\text{RE}_{t-\tau r}\}_{\tau r=0}^{\text{nr}(j)}, \{Q_{t-\tau q}\}_{\tau q=0}^{\text{nq}(j)}\right) \qquad i : \{1,3,6\},\ j : \{1,2,3\},\ nr : \{5,3,0\},\ nq : \{2,1,0\} \tag{22}$$

Test data consisted of 300 instances. Fig. 3 shows Sieve catchment.

The summary of data sets is given in Table 1.

7.2. Procedure

For a given data set, the prediction model f_f was fitted on the training data set to estimate the target t. Then the trained model was used to predict the targets in the test data set. It is also possible that the method can be applied directly to the model outputs without running the prediction model given that all or part of the input variables (most influencing variables) to the prediction model are available. Having the model outputs for the test data set, a LUEM regression model was constructed to estimate the PI on the test data set as follows. The fuzzy c-means clustering technique was first employed to construct the PI for each cluster in the training data set and then to construct the PI for each instance in the training data set. Note that the input to the LUEM may constitute part or all of input variables, which are used in the prediction model. The targets for the LUEM are the upper and lower prediction limits, which are computed from prediction intervals by adding model outputs.

In principle, the structure or even class of the prediction model and the LUEM can be different (e.g. we may use neural networks for the prediction model and linear regression for the LUEM). For the purpose of comparison to other methods, in this paper we employed the same class of regression model. The prediction limits were constructed for 95% confidence level unless specified. The performance of the LUEM is assessed by the PICM and the MPI introduced in Section 6.

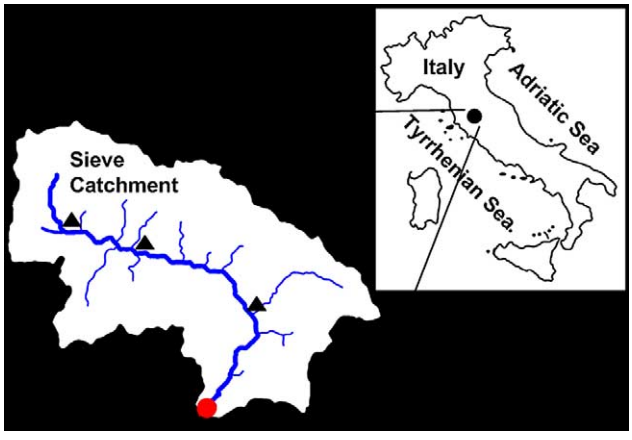

Fig. 3. Hydrological data set used in the research (Sieve catchment).

Table 1
Summary of input data to the models

Data set	Experiment	Target variable	Input variable to prediction model, clustering, LUEM	Model	
				Prediction	LUEM
Artificial	SNR1	t	x_1, x_2	LR	LR
	SNR3	t	x_1, x_2	LR	LR
	SNR4	t	x_1, x_2	LR	LR
	SNR7	t	x_1, x_2	LR	LR
Hydrological	SieveQ1Bivar	Q_{t+1}	Re_{t-5}, Q_t	LR	LR
	SieveQ3Bivar	Q_{t+3}	Re_{t-2}, Q_t	LR	LR
	SieveQ1	Q_{t+1}	Re_t, Re_{t-1}, Re_{t-2}, Re_{t-3}, Re_{t-4}, Re_{t-s}, Q_t, Q_{t-1}, Q_{t-2}	LR, LWR, ANN, MT	LR, LWR, ANN, MT
	SieveQ3	Q_{t+3}	Re_t, Re_{t-1}, Re_{t-2}, Re_{t-3}, Q_t, Q_{t-1},	LR, LWR, ANN, MT	LR, LWR, ANN, MT
	SieveQ6	Q_{t+6}	Re_t, Q_t	LR, LWR, ANN, MT	LR, LWR, ANN, MT

For the artificial data set, linear regression was employed to predict both target t values and the prediction limits using least square methods. The optimal number of clusters in fuzzy c-means clustering was identified using the Xie–Beni index S (Xie & Beni, 1991). Since the minimal values of these indices are not so pronounced, the experiments were repeated with the different numbers of clusters.

For the hydrologic data sets, bivariate linear regression was used as regression model first using only the two most influencing input variables (variables with the highest correlation with the output). Then the input variables set was extended according to Eq. (22). To estimate the effect of models' complexity on the prediction limits, experiments were also conducted using locally weighted regression, model trees and artificial neural networks as both prediction models and LUEMs.

8. Results and discussion

8.1. Clustering and finding the optimal number of clusters

Fig. 4 shows Xie–Beni separation index S used to determine the optimal numbers of clusters for the hydrological data sets. The optimal number of clusters corresponds to the minimum value of the S index. The optimal number of clusters are between 4 and 6. We also validated these optimal number of clusters by estimating prediction limits using different number of clusters. The results are presented in Fig. 5. The number of clusters corresponding to PICP close to 95% of confidence level is 6. However, the number of clusters corresponding to the minimum MPI is 4. In order to balance PICP and MPI, the number of clusters chosen for all the experiments was 5. Note that all the experiments were performed with the exponential exponent m equal to 2. As the value of m approaches to unity, clustering is close to the hard clustering (e.g. k-means), and as the value of m increases, degree of fuzziness increases.

Fig. 6 shows clustering of input examples in Sieve catchment for 1 h ahead prediction of runoff (SieveQ1 data set). The results show that input examples with very high runoff have maximum membership grades to Cluster 1 (denoted by C1). Whereas input examples with very low values of runoff have maximum membership grades to cluster 5 (C5). Table 2 shows the computed PIs for clusters for SieveQ1 data set using 95% degree confidence level (i.e. $\alpha = 0.05$). The results from the prediction model (in this case MT) are biased and model underestimates the target value Q_{t+1} in SieveQ1 data set. Bias in the model predictions

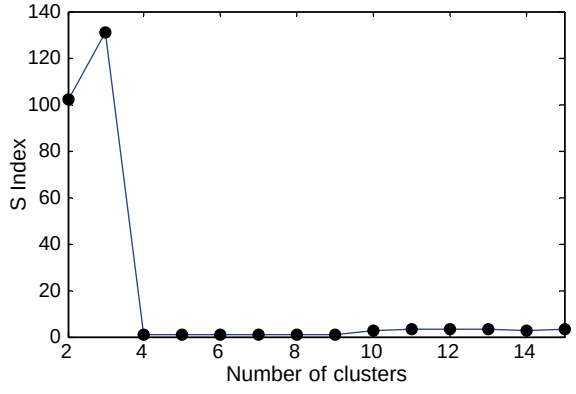

Fig. 4. Xie–Beni separation index S to find the optimal number of clusters.

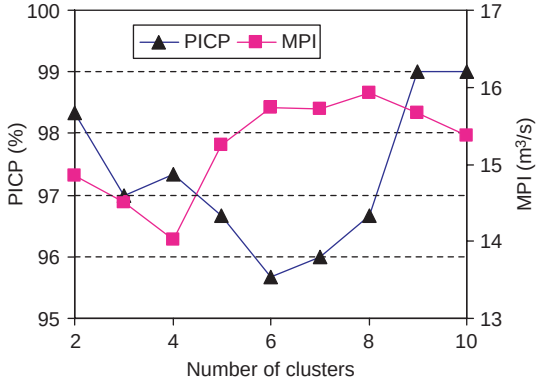

Fig. 5. Sensitivity of PICP and MPI with different number of clusters for SieveQ1 data set. M5 model tree was employed as the prediction and the LUEM model.

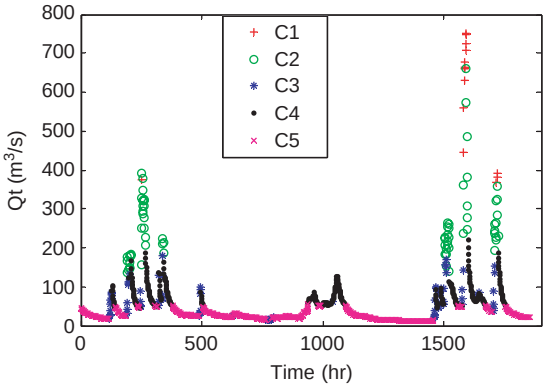

Fig. 6. Clustering of input examples in SieveQ1 training data set using fuzzy *c*-means clustering. The figure shows the maximum memberships of the input examples to the clusters.

comes from many reasons including skewed distribution of training data, selection of non-optimal value of model parameters, optimization techniques, etc. For example, hydrological river flow data is often fitted with lognormal probability distribution (Kottegoda, 1980). Consequently, computed PIs are asymmetrical and absolute values of upper PIs are greater than that of lower PIs in SieveQ1 data set.

In order to select the most important influencing variables of the model including unsupervised learning such as clustering, several approaches have been reported in the literature. We used the linear correlation analysis. Fig. 7 shows the correlation analysis of the computed upper prediction intervals in the training data set and input data for different values of lag time. It can be observed that the upper prediction intervals have the highest correlation with Re_{t-5} (lag time is 5 h) and that it decreases slowly as lag time decreases. Similarly, runoff of the present time (Q_t) contains much more information than that of previous values (up to 2 h lag) of runoff to estimate upper prediction intervals of 1 h ahead runoff (Q_{t+1}) as upper prediction intervals have highest correlation with Q_t. Similar results were also obtained for lower prediction intervals.

Table 2
Cluster centers and computed prediction intervals for each cluster in SieveQ1 data set

Cluster ID	Cluster center			Prediction interval	
	Re_{t-5} (mm/day)	Q_t (m^3/s)	Q_{t-1} (m^3/s)	Lower (m^3/s)	Upper (m^3/s)
C1	5.81	656.99	644.19	−106.99	134.84
C2	1.96	260.01	257.53	−52.50	66.28
C3	0.56	119.38	119.75	−18.05	24.57
C4	0.08	59.74	60.38	−6.08	6.7
C5	0.04	23.00	23.05	−1.01	1.42

Table 3
Comparison of results on test data set for linear regression models

Data set	Experiment	Prediction error	LRVE approach		LUEM approach	
			PICP (%)	MPI	PICP (%)	MPI
Artificial	SNR1	4.01	92.67	15.33	93.60	15.17
	SNR3	1.34	94.80	5.24	95.60	5.21
	SNR4	0.77	96.60	3.22	95.40	3.05
	SNR7	0.58	95.40	2.28	95.00	2.26
Hydrological	SieveQ1Bivar	6.13	98.00	40.98	99.93	25.61
	SieveQ3Bivar	14.18	98.00	86.90	98.00	61.76

8.2. Artificial data set

The two performance indicators: the PICP and the MPI were calculated for the LUEM and the LRVE approach for the artificial data set (see Table 3) and are visualized in Fig. 8. It is worthwhile to mention that linear regression was also employed as the LUEM. It can be seen that the LUEM appears to perform better for data with high noise if compared to the LRVE approach. For the lower noise, the PICPs of the LUEM are close to $(1-\alpha)\%$. However, The PICP for the LRVE approach considerably deviates from

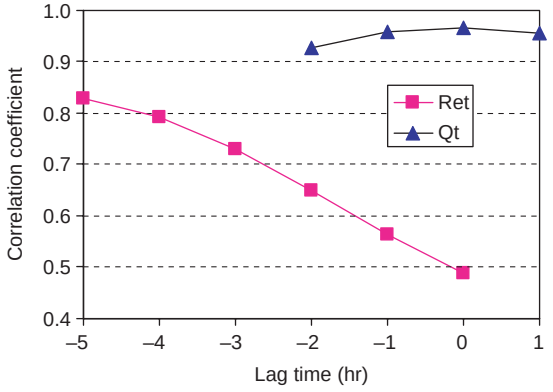

Fig. 7. Correlation between computed upper prediction interval with input data Q_t and Re_t for different values of lag time. Upper prediction interval has highest correlation with Re_{t-5} (lag time is 5 h). Similarly upper prediction interval has highest correlation with Q_t.

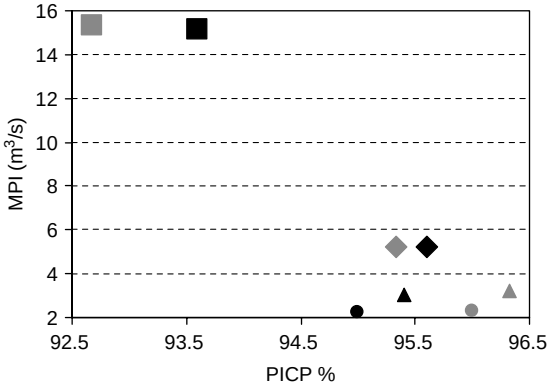

Fig. 8. Comparison of performance on artificial data set. Black blocks refer to the LUEM, the grey blocks—to the LRVE approach. Bigger size of blocks are associated with higher noise (SNR = 1), small circles are associated with low noise (SNR = 7).

Table 4
Results on test data set for hydrological data set using M5 model tree as prediction model and LUEM

Experiment	Prediction error (m^3/s)	LUEM approach		UIM approach	
		PICP (%)	MPI (m^3/s)	PICP (%)	MPI (m^3/s)
SieveQ1	3.61	96.67	15.25	91.33	11.80
SieveQ3	13.67	95.67	43.27	89.33	40.58
SieveQ6	22.89	97.67	96.34	91.33	81.6

the desired 95% confidence level. For all noise levels, the MPI for the LRVE approach are wider than those of the LUEM. Note that the uncertainty increases as the noise increases.

8.3. Hydrological data set

The performance of the LUEM is compared to that of the LRVE approach in case of the hydrologic data set with the bivariate input variable (see Table 4). It is observed that the LUEM shows superior performance for both time leads of forecasts with respect to both the PICP and the MPI.

The comparison of performance for the hydrological data set (in this case more lagged input variables were used) using various machine learning techniques is shown in Fig. 9. We employed multiple linear regression, LWR, ANN and MT to predict runoff 1, 3 and 6 h ahead. We also used these methods to estimate the prediction intervals (i.e. as LUEM). The results show that the performance of MT is better than that of the other models. The performance of linear regression and LWR are comparable.

Fig. 10 shows the computed prediction limits for 95% confidence level in SieveQ1 test data set using MT as both prediction model and the LUEM. It is observed that 96.67% of the observed data points are enclosed within the prediction limits. This value is very close to the desired value of 95%.

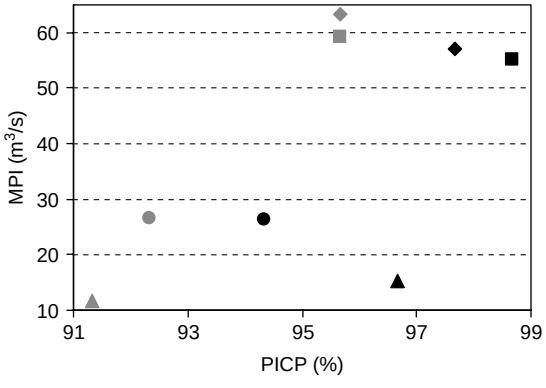

Fig. 9. Comparison of performance using different machine learning techniques for SieveQ1 data set. Black blocks refer to the LUEM methods, the grey blocks—to the LRVE approach. Rhombus, square, circular and triangle shapes represent linear regression, locally weighted regression, artificial neural network and model tree, respectively.

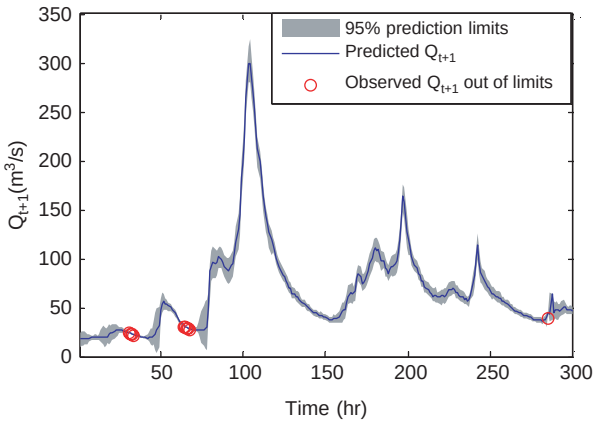

Fig. 10. Computed prediction limits for SieveQ1 test data set. M5 model tree was employed to predict 1 h ahead runoff. M5 model tree was also used to estimate uncertainty percentiles (prediction limits).

We compared the results with the uniform interval method (UIM) that constructs single PI from the empirical distribution of errors on the whole training data and is applied uniformly to the test data set. The LUEM performs consistently better than the UIM as PICPs of LUEM are closer to the desired confidence level (in this case 95%).

Fig. 11 shows the deviation of the PICPs from the desired confidence level for the hydrologic data sets using MT. The PIs were constructed for various confidence levels ranging from 10 to 99%. It is to be noticed that the PICPs are very close to the desired confidence levels at values higher than 80% and in practice the PI are constructed around this value. Furthermore, it can be noted that the PI are too narrow in most of the cases as the PICPs are below the straight line. Such evidence was also reported in (Chatfield, 2000). In these cases, the LUEM underestimates uncertainty of the model outputs.

Fig. 12 presents a fan chart showing the MPI for the hydrologic data sets with the different forecast lead times and

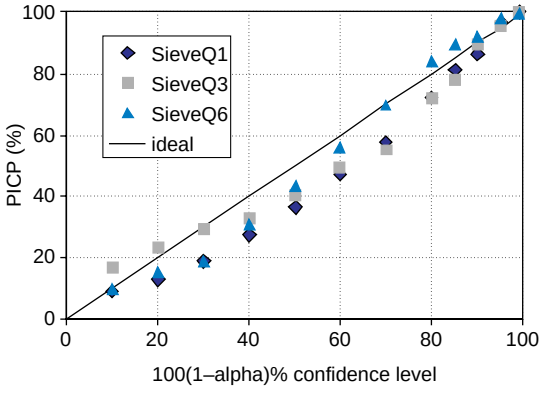

Fig. 11. The prediction interval coverage probability (PICP) for different values of confidence level. M5 model tree was employed as the prediction model and the LUEM.

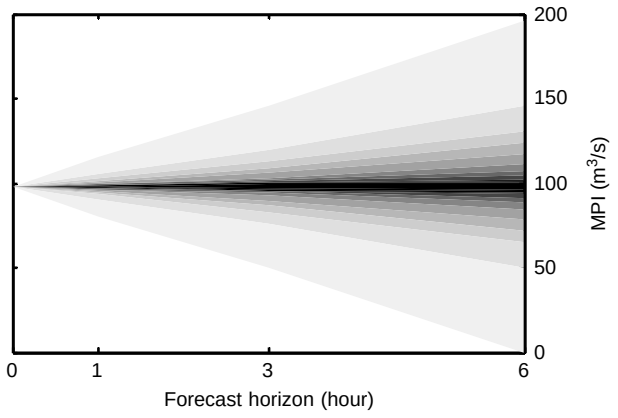

Fig. 12. A fan chart showing mean prediction interval for flow prediction in Sieve catchment up to 6 h ahead. The darkest strip covers 10% probability and the lightest 99%.

the different confidence levels. It is evident that the width of the PI increases with the increase of the confidence level. Moreover, it is also illustrated that the width of PI increases as forecast lead-time increases. Thus, model forecast uncertainty increases as the lead-time increases.

9. Conclusions

A novel method to estimate the uncertainty (expressed as prediction interval) of the model outputs using machine learning techniques is presented. The computed prediction interval explicitly takes into account all sources of uncertainty of the model outputs without attempting to separate the contribution given by the different sources of uncertainty. The methodology is independent of the prediction model structure, as it requires only the model outputs. Unlike the existing techniques, the methodology does not require the knowledge of prior distribution of parameters and does not rely on any assumptions about the distribution of errors. The presented methodology, however, computes upper and lower interval independently, whereas the analytical method in linear regression (LRVE) and re-sampling approaches compute upper and lower prediction interval symmetrically.

The methodology was applied to the data-driven prediction (regression) models based on both artificial and real hydrologic data sets, and was compared to LRVE approach typically used with the linear regression models; the superiority of the new method was demonstrated. We also demonstrate that the new method performs consistently better than the uniform interval method.

The next step in this research is to perform the detailed analysis for selecting the best input variables for the clustering technique and the uncertainty model, and to test the different clustering techniques. A challenge is to extend and generalize the proposed method in the direction of estimating more parameters characterizing uncertainty (and not only two quantiles), ideally resulting in the density estimation. Finally, the performance of the presented method should be compared with that of the other existing uncertainty estimation techniques.

Acknowledgements

The work described in this publication was parlty supported by the European Community's Sixth Framework Programme through the grant to the budget of the Integrated Project FLOOD*site*, Contract GOCE-CT-2004-505420.

References

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review*, *11*(1–5), 11–73.

Beven, K. J., & Binely, J. (1992). The future of distributed models: Model calibration and uncertainty prediction. *Hydrological Processes*, *6*, 279–298.

Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Norwell, MA: Kluwer.

Bhattacharya, B., & Solomatine, D. P. (2006). Machine learning in sedimentation modeling. *Neural Networks*, this issue, doi: 10.1016/j.neunet.2006.01.007.

Chatfield, C. (2000). *Time-series forecasting*. London/Boca Raton, FL: Chapman & Hall/CRC press.

Chinnam, R. B., & Ding, J. (1998). Prediction limit estimation for neural network models. *IEEE Transactions on Neural Networks*, *9*(6), 1515–1522.

Chryssolouris, G., Lee, M., & Ramsy, A. (1996). Confidence interval prediction for neural-network models. *IEEE Transactions on Neural Networks*, *7*, 229–236.

Dunn, J. (1973). A fuzzy relative of the isodata process and its use in detecting compact, well-separated clusters. *Journal of Cybernetics*, *3*(3), 32–57.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, *4*, 1–58.

Harnett, D. L., & Murphy, J. L. (1980). *Introductory statistical analysis*. Reading, MA: Addison-Wesley.

Heskes, T. (1997). Practical confidence and prediction interval. In M. Mozer, et al. (Ed.), *Advances in neural information processing systems: Vol. 9* (pp. 176–182). Cambridge, MA: MIT Press.

Kottegoda, N. T. (1980). *Stochastic water resources technology*. New York: McMillan.

Krzysztofowicz, R. (2000). The case for probabilistic forecasting in hydrology. *Journal of Hydrology*, *249*, 2–9.

Leonard, J. A., Kramer, M. A., & Ungar, L. H. (1992). Using radial basis functions to approximate a function and its error bounds. *IEEE Transactions on Neural Networks*, *3*(4), 624–627.

Maskey, S., Guinot, V., & Price, R. K. (2004). Treatment of precipitation uncertainty in rainfall–runoff modelling. *Advances in Water Resources*, *27*, 889–898.

Mitchell, T. M. (1998). *Machine learning*. New York: McGraw-Hill.

Nix, D., & Weigend, A. (1994). Estimating the mean and variance of the target probability distribution. *Proceeding of the International Joint Conference in Neural Networks, IEEE* (pp. 55–60).

Penny, W. D., & Roberts, S. J. (1997). Neural network predictions with error bars. Research report TR-97-1, Department of Electrical and Electronic Engineering, Imperial College, London.

Quinlan, J. R. (1992). Learning with continuous classes. *Proceedings of the fifth Australian Joint Conference on AI* (pp. 343–348), World Scientific, Singapore.

Solomatine, D. P., & Dulal, K. N. (2003). Model tree as an alternative to neural network in rainfall-runoff modelling. *Hydrological Sciences Journal*, *48*(3), 399–411.

Solomatine, D. P., Shrestha, D. L. (2004). AdaBoost.RT: A boosting algorithm for regression problems. *Proceeding of the international joint conference in neural networks*, Budapest, Hungary (pp. 1163–1168).

Solomatine, D. P., & Siek, M.B. (2004). Semi-optimal Hierarchical Regression Models and ANNs. *Proceeding of the international joint conference in neural networks*, Budapest, Hungary (pp. 1173–1177).

Solomatine, D. P., & Siek, M. B. (2006). Modular learning models in forecasting natural phenomena. *Neural Networks*, this issue, doi: 10.1016/j.neunet.2006.01.008.

Tibshirani, R. (1996). A comparison of some error estimates for neural network models. *Neural Computation*, *8*(1), 152–153.

Witten, I. H., & Frank, E. (2000). *Data mining: Practical machine learning tools and techniques with java implementations*. Los Altos, CA: Morgan Kaufmann.

Wonnacott, T. H., & Wonnacott, R. J. (1996). *Introductory statistics*. New York: Wiley.

Xie, X. L., & Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(8), 841–847.

Zadeh, L. (1965). Fuzzy sets. *Information and Control*, *8*, 338–352.