CHAPTER

7

Artificial Neural Networks: Multilayer Perceptron for Ecological Modeling

Y.-S. Park[*,1], *S. Lek*[§]

*Kyung Hee University, Seoul, Republic of Korea and [§]UMR CNRS-Université Paul Sabatier, Université de Toulouse, Toulouse, France

[1]Corresponding author: E-mail: parkys@khu.ac.kr

OUTLINE

Ecological Model Types, Volume 28
http://dx.doi.org/10.1016/B978-0-444-63623-2.00007-4

7.1 INTRODUCTION

Artificial neural networks (ANNs) are neural computation systems which were originally proposed by McCulloch and Pitts (1943) and Metropolis et al. (1953). ANNs were widely used in the 1980s thanks to significant developments in computational techniques based on self-organizing properties and parallel information systems. Rumelhart et al. (1986) proposed a new learning procedure, backpropagation (BP), for networks of neuronlike units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the network and the desired output vector. This study contributed to the widespread use of ANNs in various research fields consequent to the development of the error BP rule in parallel distributed information processing frameworks. ANNs were also used extensively in the late 1980s to interpret complex and nonlinear phenomena in machine intelligence (Lippmann, 1987; Wasserman, 1989; Zurada, 1992; Haykin, 1994).

The development of ANNs was inspired by the characteristic functioning of the human brain, but they are only remotely related to their biological counterparts. ANNs do not approach the complexity of the brain, but there are two key similarities between biological neural networks and ANNs. First, the building blocks of both networks are simple computational devices that are highly interconnected. Second, the connections between neurons determine the function of the network. A human brain consists of approximately 10^{10} neurons, computing elements, which communicate through a connection network (approximately 10^4 connections per element). ANNs function as parallel distributed computing networks and are analogous to biological neural systems in some basic characteristics (Fig. 7.1) (Wasserman, 1989; Lek and Park, 2008). As shown in Fig. 7.1B, there are many input signals $[X = (x_1,x_2,\ldots,x_n)]$ to neurons. Each input is given a relative weight $[W = (w_1,w_2,\ldots,w_n)]$ which affects the impact of that input. This is analogous to the varying synaptic strengths of the biological neurons—some inputs are more important than others in the way they combine to produce an impulse. Weights are adaptive coefficients within the network that determine the intensity of the input signal. The output signal of a neuron is produced by the summation block, corresponding roughly to the biological cell body, which adds all of the weighted inputs algebraically (Wasserman, 1989; Zurada, 1992).

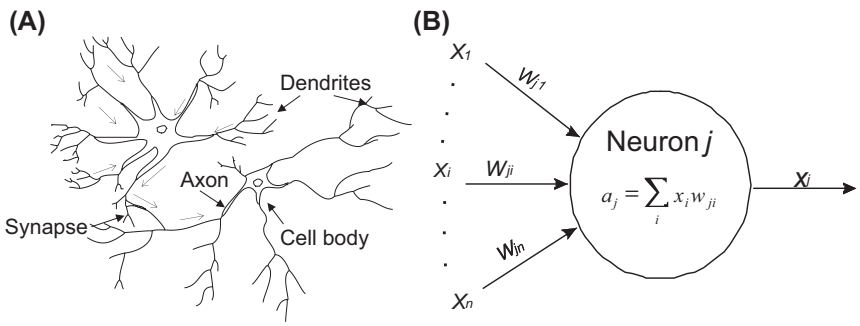

FIGURE 7.1 Schematic diagrams of (A) a biological neuron and (B) an artificial neuron as a basic processing element in a network. Each input value (x_i) is associated with a weight (w_{ji}), and the output value (X_j) can be transmitted to other units. *Arrows* indicate the direction of information flow.

ANNs are able to adjust their inner structures to provide optimal solutions, given enough data and proper initialization. If appropriate inputs are applied to an ANN, it can acquire knowledge from the environment, mimicking the functioning of a brain, and users can later recall this knowledge.

Several different types of ANNs have been developed, but two main categories can be easily recognized, according to their learning process (Lek and Park, 2008):

1. *Supervised* learning: This category utilizes a "teacher" who, in the learning phase, "tells" the ANN how well it performs or what the correct behavior should be.
2. *Unsupervised* learning: This category of ANNs autonomously analyzes the properties of the dataset and learns to reflect these properties in its output.

Both categories of ANNs have been used in ecology, with special attention to self-organizing map (SOM) for the unsupervised learning and multilayer perception (MLP) with a BP algorithm for the supervised learning.

7.2 MULTILAYER PERCEPTRON

7.2.1 Structure of MLPs

Multilayer perceptrons (MLPs) with BP learning algorithms, also called multilayer feed-forward neural networks, are very popular and are used more than other neural network types for a wide variety of problems. MLPs are based on a supervised procedure, i.e., the network builds a model based on examples in data with known outputs. An MLP has to extract this relation solely from the presented examples, which together are assumed to implicitly contain the necessary information for this relation.

An MLP comprises three layers (input, hidden, and output) with nonlinear computational elements (also called neurons and processing units). The information flows from input layer to output layer through the hidden layer (Fig. 7.2). All neurons from one layer are fully

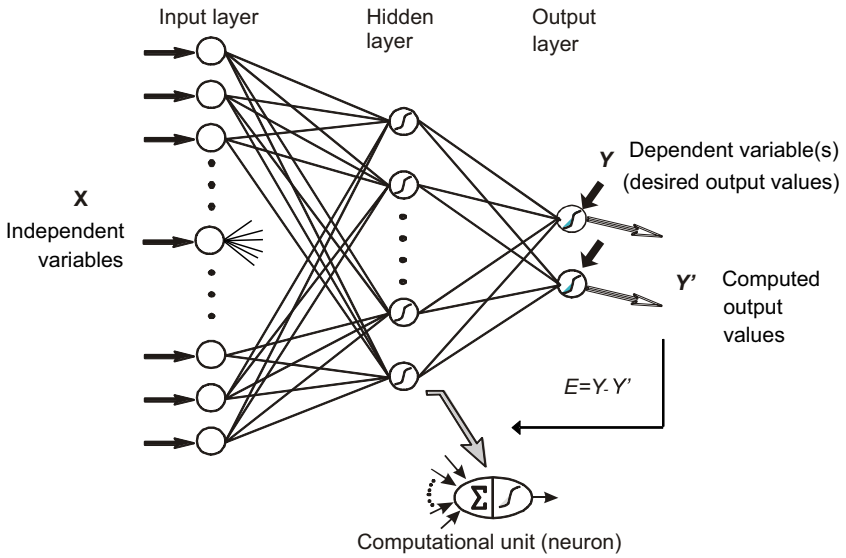

FIGURE 7.2 Three-layered feed-forward neural network with one input layer, one (or more) hidden layer(s), and one output layer: X, independent variables; Y, dependent variables; and Y′, values computed from the model.

connected to neurons in the adjacent layers. These connections are represented as weights (connection intensity) in the computational process. The weights play an important role in propagation of the signal in the network. They contain the knowledge of the neural network about the problem–solution relation. The number of neurons in the input layer depends on the number of independent variables in the model, whereas the number of neurons in the output layer is equal to the number of dependent variables. The number of output neurons can be single or multiple. In ecological modeling, environmental variables are generally given to the input layer as independent variables to predict biological variables, which are given to the output layer as target values corresponding to the given input values. Further, both the numbers of hidden layers and their neurons are dependent on the complexity of the model and are important parameters in the development of the MLP model.

7.2.2 Learning Algorithm

An MLP is trained/learned to minimize errors between the desired target values and the values computed from the model. If the network gives the wrong answer, or if the errors are greater than a given threshold, the weights are updated to minimize them. Thus, errors are reduced and, as a result, future responses of the network are likely to be correct. In the learning procedure, datasets of input and desired target pattern pairs are presented sequentially to the network. The learning algorithm of an MLP involves a forward-propagation step followed by a backward-propagation step. MLP algorithms are discussed extensively in the literature, including by Rumelhart et al. (1986) and Lin and Lee (1996) for fundamental concepts, and Lek and Guégan (2000) and Park et al. (2003) for ecological application.

An MLP learns with a algorithm in the following two phases (Lek and Guégan, 2000; Lek and Park, 2008):

7.2.2.1 Forward Propagation

The forward-propagation phase begins with presentation of an input pattern to the input layer. As in biological neural systems, in which dendrites receive signals from neurons and send them to the cell body (Fig. 7.1), an MLP receives information through input and output neurons and summarizes the information. MLP training is based on an iterative gradient algorithm to minimize error between the desired target and the computed model output.

The net input to neuron j of the hidden layer for pattern p ($NET_{p,j}$) is calculated as the summation of each output of the input layer ($x_{p,i}$; input value) multiplied by weight ($v_{p,ji}$). An activation function is applied to calculate the output of neuron j of the hidden layer ($z_{p,j}$) and the output of neuron k of the output layer ($o_{p,k}$) as follows (Chon et al., 2000):

$$f(NET) = \frac{1}{1 + \exp(-\lambda NET)} \tag{7.1}$$

where λ is an activation function coefficient, and NET is expressed either in $z_{p,j}$ or $o_{p,k}$ as follows:

$$z_{p,j} = f\left(\sum_i x_{p,i} v_{p,ji}\right) \tag{7.2}$$

$$o_{p,k} = f\left(\sum_j z_{p,j} w_{p,kj}\right) \tag{7.3}$$

where $v_{p,ji}$ and $w_{p,kj}$ are the weight of the connections between neuron i of the input layer and neuron j of the hidden layer, and that between neuron j of the hidden layer and neuron k of the output layer for pattern p, respectively. Weights are initialized as small random numbers.

Various transfer functions, such as linear functions, threshold functions, and sigmoid functions, can be used (Fig. 7.3). A sigmoid function is often used, because of its nonlinearity.

The learning algorithm modifies the weights ($v_{p,ji}$ and $w_{p,kj}$) to minimize the error. The sum of the errors (E_p) in each neuron in pattern p is calculated as follows:

$$E_p = \frac{1}{2}\sum_k \left(d_{\rho,k} - o_{\rho,k}\right)^2 \tag{7.4}$$

$$TE = \sum_\rho E_\rho \tag{7.5}$$

where $d_{p,k}$ is the target value corresponding to pattern p at neuron k, and TE is the total error during one iteration.

The forward-propagation phase continues as activation level calculations propagate forward to the output layer through the hidden layer(s). In each successive layer, every neuron sums its inputs and then applies a transfer function to compute its output. The output layer of the network then produces the final response, i.e., the estimated target value.

7.2.2.2 *Backward Propagation*

The error value associated with each neuron reflects the amount of error associated with that neuron. Consequently, the neuron is backpropagated for the appropriate weight adjustment. The weights at output neurons are updated as follows (Chon et al., 2000):

$$\delta_{p,k(o)} = o_{p,k}\left(1 - o_{p,k}\right)\left(d_{p,k} - o_{p,k}\right) \tag{7.6}$$

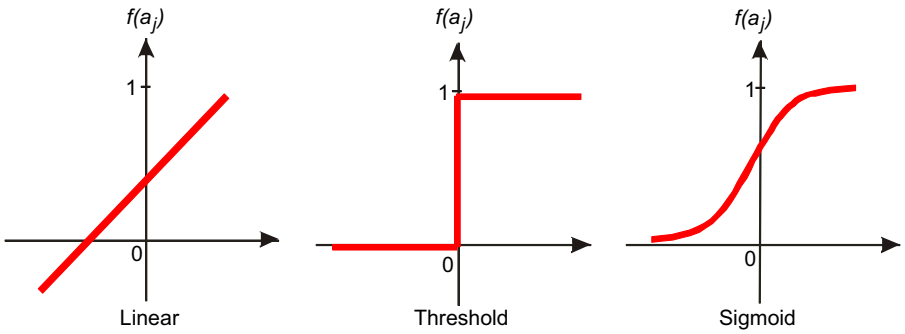

FIGURE 7.3 Three types of transfer functions commonly used in ANN models.

$$\Delta w_{p,kj}(t+1) = \eta \delta_{p,k(o)} o_{p,k} + \alpha \Delta w_{p,kj}(t) \tag{7.7}$$

$$w_{p,kj}(t+1) = w_{p,kj}(t) + \Delta w_{p,kj}(t+1) \tag{7.8}$$

where $\delta_{p,k(o)}$ is the error signal at neuron k of the output layer for pattern p, η is the learning rate coefficient, and α is the momentum coefficient.

Proper learning rate and momentum coefficient are chosen from experience. Both values have a range between zero and one. A large value for η can lead to instability in the network and unsatisfactory learning, whereas an overly small value can lead to excessively slow learning. Typically, the learning rate is varied to produce efficient learning in the network. For example, to obtain a better learning performance, the value of η may be high at the beginning and then decreases during the learning session. The momentum coefficient is defined to avoid oscillating weight changes. It contributes the impact of previous weights to calculate the current weight.

Adjustment of weights at the hidden layer is conducted in a similar manner to that at the output layer except for the error signal (δ), which is produced by summing all such products and then multiplying the derivative of the squashing function as follows (Wasserman, 1989):

$$\delta_{p,j(h)} = z_{p,j}(1 - z_{p,j}) \sum_k \delta_{p,k(o)} w_{p,kj} \tag{7.9}$$

where $\delta_{p,j(h)}$ is the error signal at neuron j of the hidden layer for pattern p.

The learning algorithm can be summarized as shown in Box 7.1 (Lin and Lee, 1996; Chon et al., 2000); the details of the algorithm can be found in Rumelhart et al. (1986), Lippmann (1987), Wasserman (1989), Zurada (1992), Lin and Lee (1996), Lek and Guégan (2000), and Chon et al. (2000).

7.2.3 Validation of Models

In the development of data-driven models, the dataset has to be split into two subsets which are respectively used for training and validation/testing. The proportion may be 1:1, 2:1, 3:1, etc. for these two sets (Fig. 7.4). However, the training set still has to be large enough to be representative of the problem and the test set has to be large enough to facilitate correct validation of the network. This procedure of partitioning the data is called *k-fold cross-validation*, sometimes also called the *hold-out procedure* (Utans and Moody, 1991; Efron and Tibshirani, 1995; Kohavi and Wolpert, 1996; Friedman, 1997; Lek and Guégan, 2000).

Both datasets contain input/output pattern pairs taken from real data. The training set is used to train the network, whereas the test set is used to assess the performance of the network. The training phase can be time-consuming depending on such factors as the network structure (numbers of input and output neurons, number of hidden layers, and number of neurons in each of the hidden layers), the number of samples in the training set, and the number of iterations.

The learning process of the model is followed by a test phase to ascertain the performance of the model and the model properties. In the test phase, the input data are fed into the network and the desired target values are compared with values computed by the network. The agreement or disagreement of these two sets is used to evaluate the performance of the model.

BOX 7.1

LEARNING ALGORITHM IN AN MLP

Step 0 (Input):

- Input a set of training pairs to the network.

Step 1 (Initialization):

- Initialize the weights to small random values.
- Set parameters such as learning rate, momentum coefficient, and TE_{max} (maximum tolerable error).

Step 2 (Training loop):

- Apply the network input pattern to the input layer.

Step 3 (Forward propagation):

- Propagate the signal forward through the network.
- Calculate the network output vector.

Step 4 (Measurement of output error):

- Calculate the errors for each of the outputs, the difference between the desired target, and the network output.

Step 5 (Error backpropagation):

- Propagate the errors backward to adjust the weights in a manner that minimizes the error.

Step 6 (One iteration loop):

- Check whether the entire set of training data has been cycled once.
- Repeat Steps 2 through 5 for the entire training dataset.

Step 7 (Check total error):

- Check whether the current total error (TE) is acceptable.
- If $TE < TE_{max}$, then terminate the training process and output the final weights; otherwise, repeat Steps 2 through 6 until the total error for the entire system is acceptably low, or the predefined number of iterations is reached.

The test phase is conducted in two steps (validation and test) when an appropriate model has to be selected from rival approaches. In this case, the dataset is subdivided into three different parts: training, validation, and testing. The models are evaluated using the validation dataset, and the best performing model is selected. Then, the accuracy of the selected model is estimated.

If the number of samples is not sufficient to permit splitting of the dataset into representative training and test sets, other strategies, such as cross-validation, may be used. In such a case, the dataset is divided into two parts with $n-1$ samples in one part and *one* sample in the other. The MLP is then trained with $n-1$ samples and tested with the last sample. The same network structure is repeated to use every sample in the n procedures. The results of these tests together facilitate determination of the performance of the model. This is called the *leave-one-out* or *Jackknife* procedure (Efron, 1979; Kohavi, 1995). This strategy is often used in ecology when only a small database is available (Lek and Guégan, 2000).

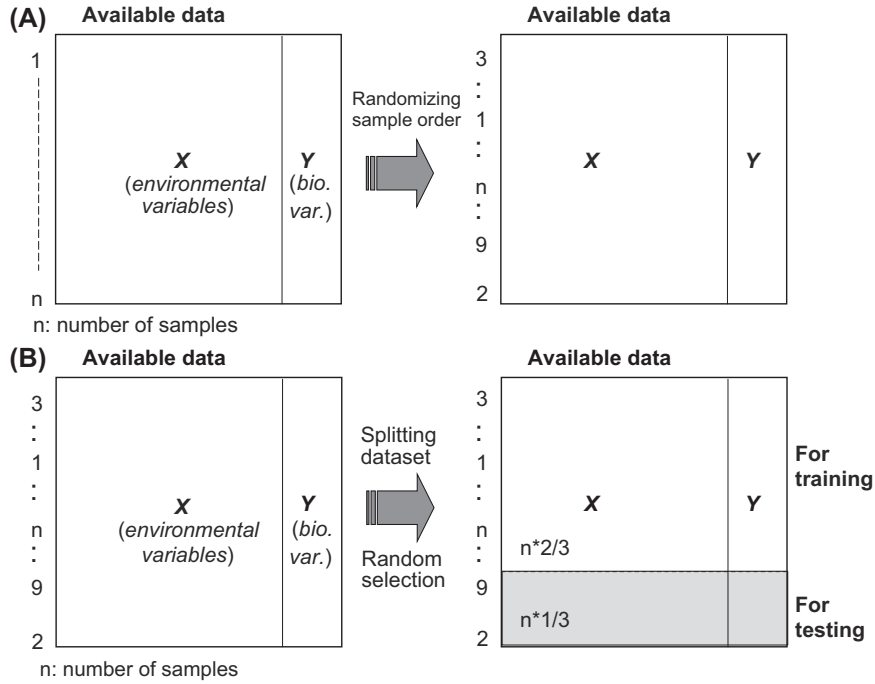

FIGURE 7.4 (A) Randomization of sample order in the dataset preparation procedure. (B) Splitting of the dataset into representative training and test sets. When the number of samples is sufficient, the dataset is divided into three parts if the validation procedure is involved in the analysis. X and Y represent independent variables and dependent variables, respectively.

7.2.4 Data Preprocessing

Ecological data are bulky, nonlinear, and complex—with noise, redundancy, internal relations, and outliers. Therefore, data-driven models are influenced by the characteristics of their datasets, although most models cover nonlinearity and complexity. To avoid these difficulties, datasets for learning the network should be carefully prepared by dealing with missing values, outliers and extremes, transformation, etc. Missing values are often replaced with mean values of the variables or estimated values from prediction models. Outliers and extremes are evaluated with box plot approaches.

Data transformation can be conducted in various ways according to the nature of the data variables. This is particularly important when variables have high variation and different units. In ecological modeling, (1) *logarithmic transformation* is most commonly used. This is especially useful when there is a high degree of variation within variables or a high degree of variation among attributes within a sample. To avoid the problem of log (0) being undefined, the number one (1) is added to all data points before applying the transformations. Meanwhile, (2) *standardization* (also called variance normalization) is very useful for studying environmental variables in ecological studies. It is a linear transformation carried out by scaling the values with mean $= 0$ and variance $= 1$ and making the data dimensionless without units. Another popular transformation is (3) *range normalization* (also called *min−max*

normalization)—a linear transformation which scales the values between zero and one in the range of the data.

Note that the same transformation should be applied to both training dataset and the test dataset for meaningful results.

Details on the diverse transformation methods are available in McCune and Grace (2002) and Legendre and Legendre (2012).

7.2.5 Overfitting

It is important to avoid overfitting (or overtraining) problems in the development of the model because in such cases the model loses generalization and is no longer applicable to other datasets which were not used to train the network. When a network is overfitted, it has a good memory for specific data. In such a case, the network cannot learn the general features inherently present in the training, but it can learn progressively more of the specific details of the training dataset perfectly. Several rules have been developed by various researchers regarding approximate determination of the required network parameters to avoid overfitting. Two parameters are responsible for this phenomenon: number of epochs and number of hidden layers (along with corresponding number of neurons for each hidden layer) (Lek and Guégan, 2000). Determination of an appropriate number for each of these parameters is the most crucial matter in MLP modeling. The optimum size of epochs, hidden layers, or hidden nodes is determined by trial-and-error using the training and test datasets. If the error in the training set decreases constantly according to the learning process, the error in the test set can increase after minima values. The training procedure must be terminated when the error in the test dataset is lowest. Otherwise, generalization of the model would no longer be possible. Geman et al. (1992) provide an excellent summary of the issues affecting generalization in neural networks.

7.2.6 Sensitivity Analysis

Sensitivity analysis is used to evaluate the behavior of a model and to ascertain the dependence of its outputs on its input parameters (Koda, 1993; Saltelli and Scott, 1997). Sensitivity analysis can be used to determine the following: (1) resemblance of a model to the system or processes being studied, (2) the main factors contributing to output variability, (3) the importance of the model parameters, (4) whether there are regions in the space with factors for which the model variation is maximum, (5) the optimal regions within the space of the factors for use in a subsequent calibration study (global optimization), and (6) if and which factors interact with each other (Saltelli et al., 2000).

MLPs are highly recognized for their high prediction power and applicability to extraction of information from complex, nonlinear relationships (Lek et al., 1996). However, the mechanisms that occur within an MLP network are often ignored. Therefore, an MLP is often considered a blackbox. Various methods have been explored to illustrate the role of variables in MLP models (Gevrey et al., 2003). They include (1) the *Partial derivatives method*, in which the partial derivatives of the output are calculated according to the input variables (Dimopoulos et al., 1995, 1999); (2) the *Weights method*, in which computation is carried out using the connection weights (Garson, 1991; Goh, 1995); (3) the *Perturbation method*, which corresponds to a

perturbation of the input variables (Scardi and Harding, 1999); (4) the *Profile method*, in which one input variable is successively varied while the others are kept constant at a fixed value (Lek et al., 1996); (5) the *Classical stepwise method*, in which the change in the error value is observed when an addition (forward) or an elimination (backward) step of the input variables is performed (Balls et al., 1996; Maier and Dandy, 1996); (6) the *Improved stepwise a* method, which uses the same principle as the *Classical stepwise method*, but the elimination of the input occurs when the network is trained and the connection weights corresponding to the input variable studied are also eliminated; and (7) the *Improved stepwise b* method, which also involves the network being trained and fixed step by step with one input variable at its mean value to note the consequences on the error. In their comparison of the methods, Gevrey et al. (2003) reported that the *Partial derivatives method* was the most useful as it gives the most complete results, followed by the *Profile* method, which gives the contribution profile of the input variables. Conversely, Olden et al. (2004) compared different methods by using Monte Carlo simulations with data exhibiting defined numeric relationships, and showed that (8) *a Connection weight approach* that uses raw input-hidden and hidden-output connection weights in the neural network provides the best methodology for accurately quantifying variable importance. However, Fischer (2015) reported that Garson's weight method is preferable to the connection weight method (Olden et al., 2004). Recently, Giam and Olden (2015) proposed a new (9) *Permutational R^2-based variable importance metric* that estimates the proportion of the total variance in the response variable which is uniquely associated with each predictor variable in both linear and nonlinear data contexts.

To conduct sensitivity analysis in an MLP, the network first has to be trained.

7.3 MLPs IN ECOLOGICAL MODELING

Following the pioneering works on ANNs, learning algorithms were applied in diverse ways in ecological modeling (Colasanti, 1991). Ever since MLPs were used to estimate biodiversity and the relationships between ecological community and their environment by Komatsu et al. (1994) and Lek et al. (1995) in ecological modeling, MLPs have been extensively implemented in various ecological studies, including nutrient flows (Nour et al., 2006; Schmid and Koskiaho, 2006), population dynamics (Elizondo et al., 1994; Baran et al., 1996; Aoki and Komatsu, 1997; Recknagel et al., 1997; Stankovski et al., 1998; Aussem and Hill, 1999), community changes (Tan and Smeins, 1996; Park et al., 2001b), determination of community types (Gevrey et al., 2004; Tison et al., 2005), remote sensing and GIS data analysis (Kimes et al., 1996; Keiner and Yan, 1998; Gross et al., 1999), effects of climate change (Lusk et al., 2001; Pearson et al., 2002), habitat and suitability and evaluation of environmental variables (Lek et al., 1996; Wagner et al., 2000; Park and Chung, 2006), and prediction of time series data (Chon et al., 2001).

Comprehensive overviews of ANN applications in ecological modeling have been compiled by Lek and Guégan (2000), Lek et al. (2005), and Recknagel (2006). There are also many valuable papers in the Special Issues of Ecological Modeling: Volume 120 (2–3) in 1999, Volume 146 (1–3) in 2001, Volume 160 (3) in 2003, Volume 195 (1–2) in 2006, Volume 203 (1–2) in 2007, and also in the Special Issues of Ecological Informatics: Volume 1 (3) in 2006, Volume 2 (3) in 2007, and Volume 29 in 2015.

7.4 ADVANTAGES AND DISADVANTAGES OF MLPs

7.4.1 Advantages of MLPs

1. MLPs only have a few parameters; thus, they can be used without prior knowledge, and the algorithms can be implemented easily.
2. MLPs create the required decision function directly through the learning process with a given dataset.
3. The learning process employed is adaptive and so MLPs can learn how to find the solution directly from the data being modeled.
4. MLPs can be applied in a wide range of fields to find solutions.
5. MLPs are used for discrimination, pattern recognition, empirical modeling, and many other tasks.
6. MLPs often provide more efficient results than conventional statistical methods when applied to the same tasks.
7. Whereas traditional linear models are inadequate for modeling data containing nonlinear characteristics, MLPs can represent both linear and nonlinear relationships.
8. Hybrid models that utilize both unsupervised and supervised learning algorithms also exist.
9. MLPs are effective for feature extraction of structure or patterns from static data as well as dynamic data (in both space and time), which are common in ecology.
10. These properties of MLPs can assist in the development of strategic tools for the management of the ecosystem.

7.4.2 Disadvantages of MLPs

1. An MLP's network requires a large number of patterns and a large number of iterations for effective learning.
2. The convergence of its learning process is dependent on the characteristics of the dataset.
3. Defining the number of neurons and layers in the hidden layer of an MLP is difficult. Consequently, many trials are required with different conditions to find the best combinations.
4. New training overwrites the properties of the existing network if existing data are not included in the new training process.
5. The network has to be retrained with both old and new data to reflect the properties of the new data to the trained network.
6. The problem of objectivity exists with MLPs, because the network is based on random effects and iterative calculations.
7. Different configurations of the network may have different convergences, depending on the initial training.
8. Because an MLP is a blackbox, it does not provide causality for events occurring in the system. However, some sensitivity analysis can help to evaluate the contribution of input variables on the model output.

7.4.3 Recommendations for Ecological Modeling

1. An MLP is a highly flexible function approximator for nonlinear and complex data which are commonly observed in ecological studies.
2. Consequently, it is a powerful tool for ecological modeling, particularly when the model is developed with limited information on the relationships between variables.
3. MLPs are applicable to both temporal and spatial data.
4. They can be useful for the prediction and discrimination of biological variables with their environmental variables.
5. They are useful for the assessment of species habitat suitability in conservation as well as restoration ecology.
6. They can be applied when large datasets are available.

7.5 EXAMPLE OF MLP USAGE IN ECOLOGICAL MODELING

7.5.1 Problem

Prediction of fish species richness in streams according to their environment gradients and evaluation for the contribution of their environmental variables.

7.5.2 Ecological Data

Fish community data and their environmental variables were used to illustrate application of MLP. The data were collected from 191 sample sites in the Adour−Garonne river network in France (Park et al., 2006). In the dataset, 34 species were recorded, and species richness was counted at each site. Among the environmental factors determining the spatial distribution of stream fish species, the upstream−downstream gradient is one of the key factors influencing stream fish assemblages, reflecting an increase in fish species richness with increasing stream size (Matthews, 1998). Water quality factors were not considered because the sampling sites were not heavily polluted. Therefore, two environmental factors, reflecting the position of each sampling site along the upstream−downstream gradient, were chosen: altitude and distance from source. In addition, the proportion (%) of various land use types in the basin surface area of each sampling site was used: urbanized artificial surface area and forest area. The proportion of agricultural area has a highly negative correlation with that of the forest area. Therefore, it was excluded from the analysis. The characteristics of the variables used in the model are given in Table 7.1.

7.5.3 Data Preparation

Variables have various units and variations. Therefore, independent variables (environmental variables) were transformed by variance normalization (standardization), resulting in unitless dimensions, and dependent variables (species richness) were transformed by using min−max normalization in the range zero to one. A dataset consisting of 191 samples was split into two subdatasets comprising a training dataset with 111 samples and a testing dataset with 80 samples.

TABLE 7.1 Characteristics of the Variables Used in the Model

Variable		*N*	Mean	SD	Range (Min–Max)	
Independent	Altitude (m)	191	255.4	233.3	1.0	1190.0
	Distance from source (km)	191	91.9	94.4	2.0	438.0
	Urban area (%)	191	1.0	1.2	0.0	10.6
	Forest area (%)	191	49.4	28.7	0.1	100.0
Dependent	Species richness	191	8.7	4.9	1.0	22.0

7.5.4 Model Training

An MLP model was trained with four inputs (independent) variables and one output (dependent) variable. One hidden layer with three neurons was used. Thus, the model had a 4-3-1 structure. Sum of square errors (SSEs)—differences between the desired target value and the computed model output values—abruptly decreased after 100 iterations, followed by oscillations (Fig. 7.5). Subsequently, the model stabilized at 490 iterations of the learning process. Consequently, the training was terminated at 490 iterations.

7.5.5 Results from the Example

Fish species richness was appropriately predicted with their four environmental factors through the learning process of MLP. The regression determination coefficients (R^2) were 0.70 and 0.62 for the training dataset and the testing dataset, respectively (Fig. 7.6). The

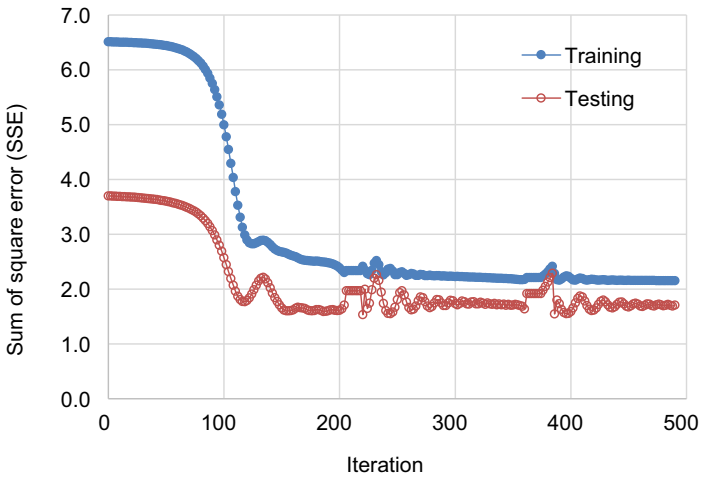

FIGURE 7.5 Changes in sum of square errors during the learning process of the model.

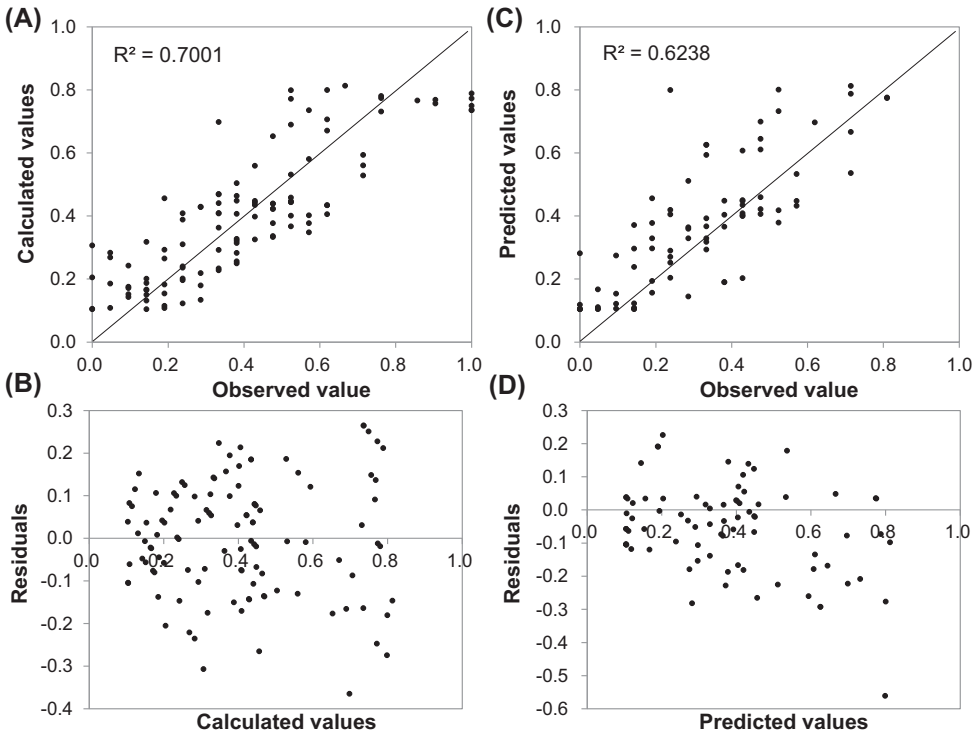

FIGURE 7.6 Relationships between (A,C) observed (desired) outputs and (B,D) values calculated by the model and their residuals. (A,B) Training dataset and (C,D) testing dataset.

residuals—the differences between desired output and calculated values—in the model were distributed evenly around zero.

7.5.6 Contribution of Variables

Following prediction of species richness with environmental factors, the problem of evaluation of the contribution of each environmental factors now had to be solved. Although an MLP is considered as a blackbox in the learning system, the contribution of its input variables can be evaluated with several proposed methods. In this paper, we use a partial derivatives method as an example.

The partial derivative values of model response mainly responded negatively to altitude, indicating that species richness is negatively influenced by altitude (Fig. 7.7A). In contrast, the partial derivatives were positive as a function of distance from source (Fig. 7.7B), reflecting an increase in fish species richness with increasing stream size. In general, fish species richness is higher downstream than upstream. This is well recognized in fish ecology.

The partial derivatives are positive at less than 2% of the urban area. The derivatives are also positive at up to 40% of forest area but are negative at a higher proportion of the forest area (Figs. 7.7C and D). These results are congruent with those for altitude and distance from

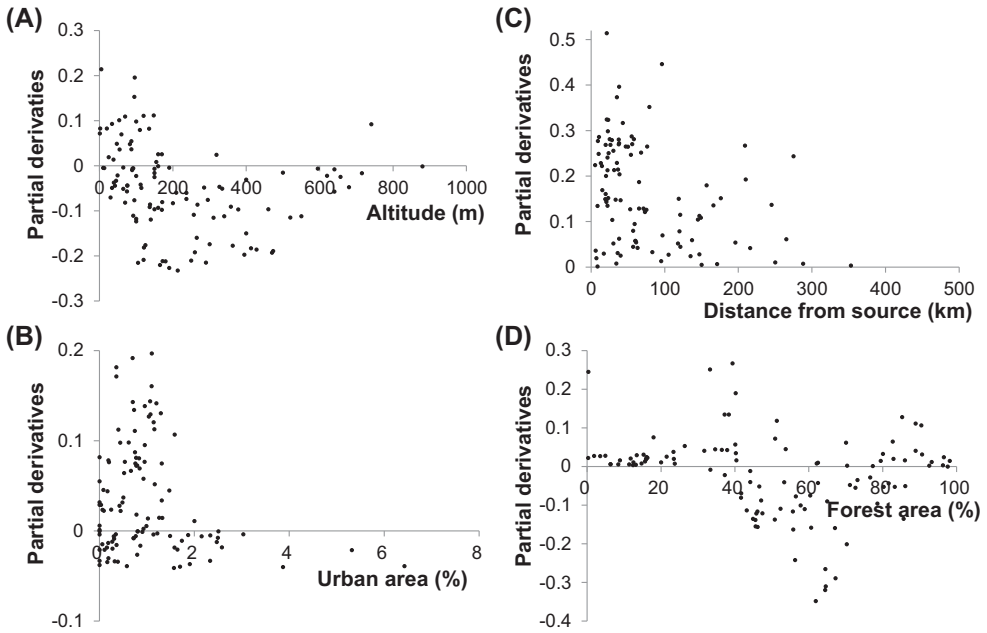

FIGURE 7.7 Partial derivatives of the MLP model response with respect to each independent variable.

source. Thus, they reflect the fact that fish species richness is lower at a high proportion of the forest area with high altitude. Altitude has a negative correlation with distance from source, but a positive correlation with proportion of forest area.

The relative contribution of each input variable was evaluated on the basis of the partial derivatives of the model responses. Among four input variables, distance from source is

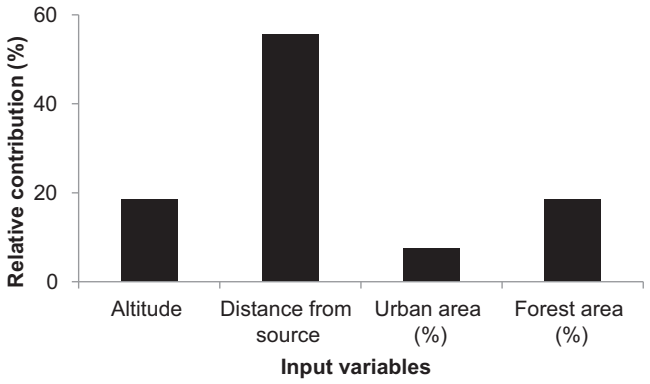

FIGURE 7.8 Relative contribution of input variables based on the partial derivatives of the model responses.

the most important in the determination of fish species richness in the streams studied, presenting a contribution of more than 55%. Altitude and forest areas show similar contribution, with each at 18%, and urban areas have the lowest contribution with less than 8% contribution (Fig. 7.8).

References

Aoki, I., Komatsu, T., 1997. Analysis and prediction of the fluctuation of the sardine abundance using a neural network. Oceanologica Acta 20, 81–88.

Aussem, A., Hill, D., 1999. Wedding connectionist and algorithmic modelling towards forecasting *Caulerpa taxifolia* development in the north-western Mediterranean sea. Ecological Modelling 120, 225–236.

Balls, G.-R., Palmer Brown, D., Sanders, G.-E., 1996. Investigating microclimatic influences on ozone injury in clover (*Trifolium subterraneum*) using artificial neural networks. New Phytologist 132, 271–280.

Baran, P., Lek, S., Delacoste, M., Belaud, A., 1996. Stochastic models that predict trout population density or biomass on a mesohabitat scale. Hydrobiologia 337, 1–9.

Chon, T.-S., Kwak, I.-S., Park, Y.-S., Kim, T.-H., Kim, Y., 2001. Patterning and short-term predictions of benthic macroinvertebrate community dynamics by using a recurrent artificial neural network. Ecological Modelling 146 (1–3), 181–193.

Chon, T.-S., Park, Y.-S., Kim, J.-M., Lee, B.-Y., Chung, Y.-J., Kim, Y., 2000. Use of an artificial neural network to predict population dynamics of the forest-pest pine needle gall midge (Diptera: Cecidomyiida). Environmental Entomology 29 (6), 1208–1215.

Colasanti, R.L., 1991. Discussions of the possible use of neural network algorithms in ecological modelling. Binary 3, 13–15.

Dimopoulos, I., Chronopoulos, J., Chronopoulou Sereli, A., Lek, S., 1999. Neural network models to study relationships between lead concentration in grasses and permanent urban descriptors in Athens city (Greece). Ecological Modelling 120, 157–165.

Dimopoulos, Y., Bourret, P., Lek, S., 1995. Use of some sensitivity criteria for choosing networks with good generalization ability. Neural Processing Letters 2, 1–4.

Efron, B., Tibshirani, R.J., 1995. Cross-Validation and the Bootstrap: Estimating the Error Rate of a Prediction Rule. Department of Statistics, Standford University, Standford. Technical Report 176.

Efron, B., 1979. Bootstrap methods: another look at the jackknife. Annals of Statistics 7 (1), 1–26.

Elizondo, D.A., McClendon, R.W., Hoongenboom, G., 1994. Neural network models for predicting flowering and physiological maturity of soybean. Transactions of the ASAE 37 (3), 981–988.

Fischer, A., 2015. How to determine the unique contributions of input-variables to the nonlinear regression function of a multilayer perceptron. Ecological Modelling 309–310, 60–63.

Friedman, J.H., 1997. On bias, variance, 0/1-loss and the curse-of-dimensionality. Data Mining and Knowledge Discovery 1, 55–77.

Garson, G.D., 1991. Interpreting neural network connection weights. Artificial Intelligence Expert 6, 47–51.

Geman, S., Bienenstock, E., Doursat, R., 1992. Neural networks and the bias/valance dilemma. Neural Computation 4, 1–58.

Gevrey, M., Dimopoulos, I., Lek, S., 2003. Review and comparison of methods to study the contribution of variables in Artificial Neural Network models. Ecological Modelling 160, 249–264.

Gevrey, M., Rimet, F., Park, Y.S., Giraudel, J.L., Ector, L., Lek, S., 2004. Water quality assessment using diatom assemblages and advanced modelling techniques. Freshwater Biology 49, 208–220.

Giam, X., Olden, J.D., 2015. A new R^2-based metric to shed greater insight on variable importance in artificial neural networks. Ecological Modelling 313, 307–313.

Goh, A.T.C., 1995. Back-propagation neural networks for modelling complex systems. Artificial Intelligence in Engineering 9, 143–151.

Gross, L., Thiria, S., Frouin, R., 1999. Applying artificial neural network methodology to ocean color remote sensing. Ecological Modelling 120, 237–246.

Haykin, S., 1994. Neural Networks: A Comprehensive Foundation. Prentice-Hall, Upper Saddle River, NJ.

Keiner, L.E., Yan, X.-H., 1998. A neural network model for estimating sea surface chlorophyll and sediments from Thematic Mapper imagery. Remote Sensing of Environment 66, 153–165.

Kimes, D.S., Holben, B.N., Nickeson, J.E., McKee, W.A., 1996. Extracting forest age in a Pacific Northwest forest from thematic mapped and topographic data. Remote Sensing of Environment 56, 133–140.

Koda, M., 1993. Sensitivity analysis of stochastic dynamical systems. International Journal of Systems Science 23 (12), 2187–2195.

Kohavi, R., Wolpert, D.H., 1996. Bias plus variance decomposition for zero-one loss functions. In: Saitta, L. (Ed.), Machine Learning: Proceedings of the Thirteenth International Conference. Morgan Kaufmann, Bari, Italy, pp. 275–283.

Kohavi, R., 1995. A study of the cross-validation and bootstrap for accuracy estimation and model selection. In: Proceeding of the 14th International Joint Conference on Artificial Intelligence (IJCAI), pp. 1137–1143. Montréal, Canada.

Komatsu, T., Aoki, I., Mitani, I., Ishii, T., 1994. Prediction of the catch of Japanese sardine larvae in Sagami Bay using a neural network. Fisheries Science 60 (4), 385–391.

Legendre, P., Legendre, L., 2012. Numerical Ecology, 3rd English edition. Developments in Environmental Modelling, Vol. 24. Elsevier Science BV, Amsterdam.

Lek, S., Guégan, J.F. (Eds.), 2000. Artificial Neuronal Networks: Application to Ecology and Evolution. Springer, Berlin.

Lek, S., Belaud, A., Baran, P., Dimopoulos, I., Delacoste, M., 1996. Role of some environmental variables in trout abundance models using neural networks. Aquatic Living Resources 9, 23–29.

Lek, S., Belaud, A., Dimopoulos, I., Lauga, J., Moreau, J., 1995. Improved estimation, using neural networks, of the food consumption of fish populations. Marine and Freshwater Research 46, 1229–1236.

Lek, S., Scardi, M., Verdonschot, P., Descy, J., Park, Y.-S. (Eds.), 2005. Modelling Community Structure in Freshwater Ecosystems. Springer, Berlin.

Lek, S., Park, Y.-S., 2008. Artificial Neural Networks. In: Jorgensen, S.E., Fath, B. (Eds.), Encyclopedia of Ecology. Elsevier, Amsterdam, pp. 237–245.

Lin, C.-T., Lee, C.S.G., 1996. Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems. Prentice-Hall, NJ, Upper Saddle River.

Lippmann, R.P., April 1987. An Introduction to Computing With Neural Nets. IEEE ASSP, pp. 4–22.

Lusk, J.J., Guthery, F.S., DeMaso, S.J., 2001. Northern bobwhite (*Colinus virginianus*) abundance in relation to yearly weather and long-term climate patterns. Ecological Modelling 146, 3–15.

Maier, H.R., Dandy, G.C., 1996. The use of artificial neural networks for the prediction of water quality parameters. Water Resources Research 32, 1013–1022.

Matthews, W.J., 1998. Patterns in Freshwater Fish Ecology. Chapman and Hall, New York, p. 756.

McCulloch, W.S., Pitts, W., 1943. A logical calculus of the ideas imminent in nervous activity. Bulletin of Mathematical Biology 5, 115–133.

McCune, B., Grace, J.B., 2002. Analysis of Ecological Communities. MjM Software Design, Gleneden Beach, Oregon.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E., 1953. Equation of state calculation by fast computing machines. The Journal of Chemical Physics 21, 1087–1091.

Nour, M.H., Smith, D.W., El-Din, M.G., Prepas, E.E., 2006. Neural networks modelling of streamflow, phosphorus, and suspended solids: application to the Canadian Boreal forest. Water Science and Technology 53 (10), 91–99.

Olden, J.D., Joy, M.K., Death, R.G., 2004. An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. Ecological Modelling 178, 389–397.

Park, Y.-S., Céréghino, R., Compin, A., Lek, S., 2003. Applications of artificial neural networks for patterning and predicting aquatic insect species richness in running waters. Ecological Modelling 160, 265–280.

Park, Y.-S., Chon, T.-S., Kwak, I.S., Kim, J.-K., Jorgensen, S.E., 2001b. Implementation of artificial neural networks in patterning and prediction of exergy in response to temporal dynamics of benthic macroinvertebrate communities in streams. Ecological Modelling 146, 143–157.

Park, Y.-S., Chung, Y.-J., 2006. Hazard rating of pine trees from a forest insect pest using artificial neural networks. Forest Ecology and Management 222 (1–3), 222–233.

Park, Y.-S., Grenouillet, G., Esperance, B., Lek, S., 2006. Stream fish assemblages and basin land cover in a river network. Science of the Total Environment 365, 140–153.

Pearson, R.G., Dawson, T.P., Berry, P.M., Harrison, P.A., 2002. SPECIES: a spatial evaluation of climate impact on the envelope of species. Ecological Modelling 154, 289–300.

Recknagel, F., 2006. Ecological Informatics: Understanding Ecology by Biologically-Inspired Computation. Springer, Berlin.

Recknagel, F., French, M., Harkonen, P., Yabunaka, K.-I., 1997. Artificial neural network approach for modelling and prediction of algal blooms. Ecological Modelling 96, 11–28.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by backpropagation error. Nature 323, 533–536.

Saltelli, A., Chan, K., Scott, M. (Eds.), 2000. Sensitivity Analysis. John Wiley & Sons, New York.

Saltelli, A., Scott, M., 1997. The role of sensitivity analysis in the corroboration of models and its links to model structural and parametric uncertainty. Special issue Reliability Engineering and System Safety 57 (1).

Scardi, M., Harding, L.W., 1999. Developing an empirical model of phytoplankton primary production: a neural network case study. Ecological Modelling 120, 213–223.

Schmid, B.H., Koskiaho, J., 2006. Artificial neural network modeling of dissolved oxygen in a wetland pond: the case of Hovi, Finland. Journal of Hydrologic Engineering 11, 188–192.

Stankovski, V., Debeljak, M., Bratko, I., Adamic, M., 1998. Modelling the population dynamics of red deer (*Cervus elaphus* L.) with regard to forest development. Ecological Modelling 108, 143–153.

Tan, S.S., Smeins, F.E., 1996. Predicting grassland community changes with an artificial neural network model. Ecological Modelling 84, 91–97.

Tison, J., Park, Y.-S., Coste, M., Wasson, J.G., Ector, L., Rimet, F., Delmas, F., 2005. Typology of diatom communities and the influence of hydro-ecoregions: a study at the French hydrosystem scale. Water Research 39, 3177–3188.

Utans, J., Moody, J.E., 1991. Selecting neural network architectures via the prediction risk: application to corporate bond rating prediction. In: Proceedings of the First International Conference on Artificial Intelligence Applications on Wall Street. IEEE Computer Society Press, Los Alamitos, CA.

Wagner, R., Dapper, T., Schmidt, H.H., 2000. The influence of environmental variables on the abundance of aquatic insects: a comparison of ordination and artificial neural networks. Hydrobiologia 422–423, 143–152.

Wasserman, P.D., 1989. Neural Computing: Theory and Practice. Van Nostrand Reinhold, New York.

Zurada, J.M., 1992. Introduction to Artificial Neural Systems. West Publishing Company, New York.