ELSEVIER

Pattern Recognition Letters 20 (1999) 1241–1248

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

An incremental-learning neural network for the classification of remote-sensing images

L. Bruzzone *, D. Fernàndez Prieto

Department of Biophysical and Electronic Engineering, University of Genoa, Via Opera Pia 11 A, 16145 Genova, Italy

Abstract

A novel classifier for the analysis of remote-sensing images is proposed. Such a classifier is based on Radial Basis Function (RBF) neural networks and relies on an incremental-learning technique. This technique allows the periodical acquisition of new information whenever a new training set becomes available, while preserving the knowledge learnt by the network on previous training sets. In addition, in each retraining phase, the network architecture is automatically updated so that new classes may be considered. These characteristics make the proposed neural classifier a promising tool for several remote-sensing applications.

Keywords: Neural networks; Remote sensing; Radial Basis Functions; Incremental learning

1. Introduction

An important problem encountered in the classification of remote-sensing data is that classical classifiers, once they have been trained on a data set related to a specific image, seldom attain acceptable classification accuracies on different images (even if the land-cover classes present in all the images are the same). Several factors contribute to this problem: differences in the atmospheric and light conditions at the image-acquisition dates, sensor non-linearities, differences in soil moisture levels, etc. From an operational point of view, such a problem represents a serious limitation in several real-world applications (e.g., development of automatic monitoring systems able to categorise extensive territorial areas periodically; development of classification systems able to efficiently recognise a predefined set of land-cover classes in remote-sensing images acquired in different areas). In this context, the design of a "robust" classification system capable to perform efficiently on different images, irrespective of the acquisition dates or even the geographical areas considered, is a major challenge for the remote-sensing community.

An important step in the development of such a system is the definition of a classifier able to acquire new knowledge when new training sets become available, while preserving the current one (this problem is known in the pattern-recognition literature as the incremental-learning problem). On the one hand, such a classifier would be able to learn from new training sets, thus increasing its capabilities to classify new images; on the other, these properties would allow the classifier to preserve the information acquired from different

* Corresponding author. Tel.: +39-010-353-2672; fax: +39-010-353-2134.

E-mail address: lore@dibe.unige.it (L. Bruzzone)

0167-8655/99/$ - see front matter
PII: S0167-8655(99)00091-4

data sets and, consequently, to improve its generalisation capabilities on unknown images (i.e., remote-sensing images for which a training set is not available). Unfortunately, to accomplish such a task, conventional classifiers have to be completely retrained on the new samples along with the old ones every time a new training set is available. This is an impractical solution as it requires the storage of all available training sets, considerable computation time and, in most cases, a complete redefinition of the architecture of the classifier considered.

Only few papers have been published that deal with the incremental-learning problem, which is still an open issue in the pattern-recognition literature, in particular, in the field of remote sensing. It is worth noting that in most attempts to address the incremental-learning problem for classification purposes (Fu et al., 1996; Park et al., 1991; Schaal and Atkeson, 1998), the number of information classes (i.e., land-cover classes in remote-sensing problems) was considered as a fixed parameter of the problem (Fu et al., 1996). This hinders the capabilities of the classifier to be adapted, whenever a new training set contains some classes different from those considered in the initial design of the classification system. In the context of remote sensing, such a constraint may be a serious drawback.

In this paper, we propose a novel classifier based on Radial Basis Function (RBF) neural networks (Powell, 1987; Broomhead and Lowe, 1988; Moody and Darken, 1989), which aims at meeting the primary requirements of a robust classifier for remote-sensing images. In particular, the proposed classifier relies on an incremental-learning technique that allows the acquisition of new knowledge, while preserving the existing one. In addition, such a technique permits an automatic adaptation of the network architecture so that new information classes may be learnt in each retraining phase.

2. The proposed neural-network classifier

2.1. Background

The proposed classifier is based on a Gaussian RBF neural network (Bishop, 1995). The choice of this neural model is justified by some of its particular properties, i.e., local learning, fast training phase, ability to recognise when an input pattern has fallen into a region of the input space without training data, and capability to provide high classification accuracies on remote-sensing images (Bishop, 1995; Bruzzone and Fernàndez Prieto, 1999).

Generally, a Gaussian RBF neural network is composed of three layers (see Fig. 1). Input neurons (as many as input features) just propagate input features to the next layer. Each neuron in the hidden layer is associated with a Gaussian kernel function, $\varphi_j(\cdot)$, characterised by a centre μ_j and a width σ_j. The output layer is composed of as many neurons as there are classes to be recognised. Each output neuron o_i computes a simple weighed summation over the responses of the hidden neurons to a given input pattern described by the feature vector $\boldsymbol{x}_n$:

$$o_i(\boldsymbol{x}_n) = \sum_{j=1}^{M} w_{ij}\varphi_j(\boldsymbol{x}_n) + w_{\text{bias},i}, \tag{1}$$

where M is the number of hidden neurons, w_{ij} represents the weight associated with the connection between the kernel function $\varphi_j(\cdot)$ and the output neuron o_i, and $w_{\text{bias},i}$ is the bias of the output neuron o_i.

The proposed classifier is based on the supervised learning strategy proposed in (Bruzzone and Fernàndez Prieto, 1999). In particular, this strategy selects the kernel-function parameters of a network in such a way that each kernel function

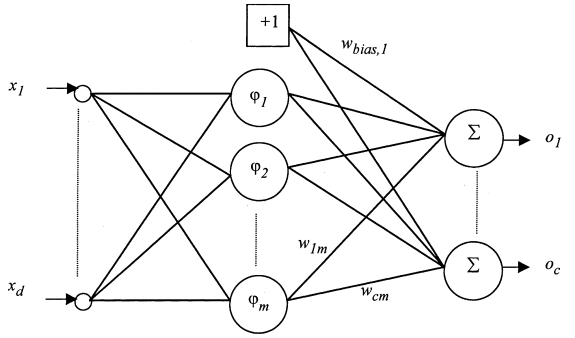

Fig. 1. Typical architecture of a classifier based on Radial Basis Function (RBF) neural networks.

can be associated with a given information class. Consequently, each information class can be represented by a different set of kernel functions distributed in the corresponding region of the input space. In this context, given the kernel function $\varphi_j(\cdot)$ associated with the information class ω_i, the corresponding parameters provide a local description of the training samples belonging to ω_i. This suggests the idea of describing the long-term memory of the network through a set of prototypes, $\boldsymbol{P} = \{\pi_1, \pi_2, \ldots, \pi_M\}$, where each prototype π_j is characterised by the triple $\{\mu_j, \sigma_j, \alpha_j\}$, α_j being the mass coefficient that denotes the number of training samples associated with the kernel function $\varphi_j(\cdot)$. The first two parameters are related to the local distribution of the corresponding class (and characterise the kernel functions of the network); the mass parameter concerns the prior probability of a class. Consequently, the status of the network (i.e., the knowledge captured in the training phase) can be completely described by the tuple $\mathrm{Net} = \{\boldsymbol{P}, \boldsymbol{W}\}$, where $\boldsymbol{W}$ represents the weight matrix and $\boldsymbol{P}$ is the above-defined set of prototypes (i.e., the long-term memory of the network).

2.2. Problem formulation and general description of the method

Let us consider a classification problem in which a training set $\boldsymbol{T}_k$ is available at each time t_k $(k = 1, 2, \ldots, m)$. Each $\boldsymbol{T}_k$ contains N_k samples related to a set $\Lambda_k = \{\lambda_1^k, \lambda_2^k, \ldots, \lambda_{q_k}^k\}$ of q_k different land-cover classes. We assume that the land-cover classes considered may change from one training set to another; therefore, a class present in $\boldsymbol{T}_k$ may be a "new" class for the classifier. Let $\Omega_k = \{\omega_1^k, \omega_2^k, \ldots, \omega_{C_k}^k\}$ be the set of C_k classes already "known" to the network at t_k. Let us also denote by $\mathrm{Net}(t_k) = \{\boldsymbol{P}_k, \boldsymbol{W}_k\}$ the network status at the time t_k, where $\boldsymbol{P}_k$ and $\boldsymbol{W}_k$ represent the set of M_k prototypes (i.e., $\boldsymbol{P}_k = \{\pi_1^k, \pi_2^k, \ldots, \pi_{M_k}^k\}$) and the set of $C_k \times (M_k + 1)$ weights, respectively, that characterise the network at t_k.

In this context, we propose a retraining technique that allows the network $\mathrm{Net}(t_k)$ to be updated (whenever a new training set $\boldsymbol{T}_{k+1}$ becomes available) so that the resulting network $\mathrm{Net}(t_{k+1})$ incorporates both the "new" information contained in the training set $\boldsymbol{T}_{k+1}$ and the "existing" knowledge of the network (provided that the information about the classes contained in $\boldsymbol{T}_{k+1}$ is not in conflict with the current knowledge of the network).

A schematic representation of the proposed incremental learning technique is shown in Fig 2. As one can see, the retraining phase of the network is carried out in two sequential steps. The first aims at updating the long-term memory of the network $\boldsymbol{P}_k$, the latter aims at recalculating the weight matrix on the basis of both the new knowledge (i.e., the training samples in $\boldsymbol{T}_{k+1}$) and the existing one (i.e., the set of prototypes $\boldsymbol{P}_k$ at the time t_k).

In the following, a detailed description of the training procedure for the classifier is provided. Such a procedure consists of two different phases: the initial training of the network (at the

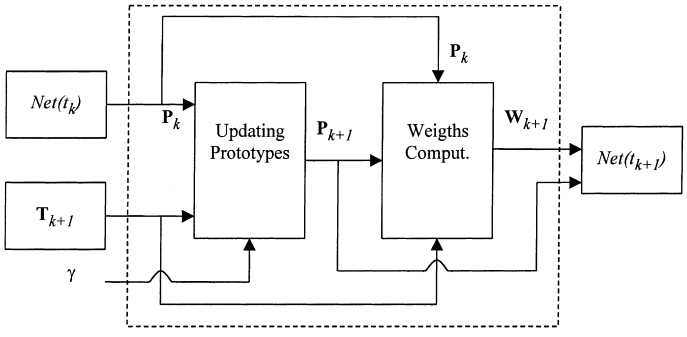

Fig. 2. Schematic representation of the proposed retraining technique.

time t_1) and retraining of the network (at the time t_k).

2.3. Initial training of the network

Let us consider the network at the time t_0, at which $\boldsymbol{W}_0 \equiv \boldsymbol{P}_0 \equiv \emptyset$, as no training set has yet been learnt by the classifier. When the training set $\boldsymbol{T}_1$ becomes available at the time t_1, the initial training of the network is carried out in accordance with the learning procedure proposed by Bruzzone and Fernàndez Prieto, 1999. Such a procedure is divided into two steps.

In the first step, the set of prototypes $\boldsymbol{P}_1$ (and hence the parameters of the kernel functions) is obtained by applying a clustering procedure separately to the training samples of each class present in $\boldsymbol{T}_1$. At the end of this process, a prototype π_j^1 (and the corresponding kernel function $\varphi_j^1(\cdot)$) is derived from each cluster $\boldsymbol{S}_j^1$. This is done by associating the prototype centre μ_j^1 with the barycentre of the cluster $\boldsymbol{S}_j^1$, and the corresponding width σ_j^1 with the standard deviation computed on the training patterns included in $\boldsymbol{S}_j^1$. In addition, the mass coefficient α_j^1 is made equal to the cardinality of $\boldsymbol{S}_j^1$.

In the second step, the set of weights $\boldsymbol{W}_1$ corresponding to the connections between the hidden nodes and the output units is selected. This is done by minimizing the following *sum-of-squares* error function (Bishop 1995; Bruzzone and Fernàndez Prieto, 1999):

$$E_1 = \frac{1}{2} \sum_{i=1}^{q_1} \sum_{n=1}^{N_1} [o_i(\boldsymbol{x}_n) - T_i(\boldsymbol{x}_n)]^2, \tag{2}$$

where $T_i(\boldsymbol{x}_n)$ is the target for the output $o_i(\boldsymbol{x}_n)$. As the *sum-of-squares* error function is a quadratic function of the weights, its minimum can be found by solving a set of linear equations in terms of a pseudo-inverse matrix (Bishop, 1995).

2.4. Retraining phase of the network

Let us now consider the network status Net(t_k) at the generic time t_k. Let us also assume that a new training set $\boldsymbol{T}_{k+1}$ is available at the time t_{k+1}. Analogously to the initial training phase, the retraining phase of the network is composed of two main steps: updating of prototypes and weights computation.

2.4.1. Updating of prototypes

In this phase, the long-term memory of the network (i.e., the set of prototypes $\boldsymbol{P}_k$) is updated by following a rather simple and fast incremental clustering procedure. The proposed procedure allows not only the updating of the existing prototypes (and hence the updating of the parameters of the existing kernel functions) but also, if necessary, the generation of new prototypes (and hence the allocation of new hidden neurons).

Let $\boldsymbol{x}_n^i$ be the nth training sample belonging to the class λ_i^{k+1} of the training set $\boldsymbol{T}_{k+1}$. Let π_{nearest} be the prototype nearest to $\boldsymbol{x}_n^i$ (i.e., the prototype of class λ_i^{k+1} whose centre μ_{nearest} is the closest to $\boldsymbol{x}_n^i$). Let us also denote by $\sigma_{\min}^k$ the minimum width associated with an existing prototype at the time t_k. Let us finally define the condition of similarity $S(\boldsymbol{x}_n^i, \pi_{\text{nearest}})$ of the training sample $\boldsymbol{x}_n^i$ to the corresponding nearest prototype π_{nearest} as:

$$S(\boldsymbol{x}_n^i, \pi_{\text{nearest}}) = \begin{cases} 1 & \text{if } \|\boldsymbol{x}_n^i - \mu_{\text{nearest}}\| < \gamma\sigma_{\text{nearest}}, \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where γ is a predefined distance parameter (which is the only input required by the algorithm).

In this context, the proposed algorithm is based on a sequential comparison of all the training samples $\boldsymbol{x}_n^i$ with the corresponding nearest prototype π_{nearest}. Such a comparison is made in terms of the above-described similarity function $S(\boldsymbol{x}_n^i, \pi_{\text{nearest}})$, which can be considered as a measure of the capability of π_{nearest} to represent the training sample $\boldsymbol{x}_n^i$. Therefore, if $\boldsymbol{x}_n^i$ is found to be "similar" to π_{nearest} (i.e., $S(\boldsymbol{x}_n^i, \pi_{\text{nearest}}) = 1$), such a prototype is updated in accordance with $\boldsymbol{x}_n^i$ in order to refine the corresponding parameters. Otherwise, if $\boldsymbol{x}_n^i$ is found to be "different" from the corresponding prototype π_{nearest} (i.e., $S(\boldsymbol{x}_n^i, \pi_{\text{nearest}}) = 0$), the training sample considered cannot be efficiently represented by the current long-term memory of the network, and hence a new prototype is generated in order to represent the new information. It is

worth noting that the generation of a new prototype involves the allocation of a new hidden neuron of the network (and consequently the definition of a new kernel function). The proposed algorithm is described in greater detail in the following.

Initialise $\boldsymbol{P}_{k+1} = \boldsymbol{P}_k$.
For all the classes $\lambda_i^{k+1} \in \boldsymbol{T}_{k+1}$ do:
  For all the training patterns $\boldsymbol{x}_n^i \in \lambda_i^{k+1}$ do:
    Search in $\boldsymbol{P}_{k+1}$ for the prototype π_{nearest} nearest to $\boldsymbol{x}_n^i$;
    If [π_{nearest} exists] and $[S(\boldsymbol{x}_n^i) = 1]$:
      then update the π_{nearest} parameters;
      else: initialise a new prototype π_{new} characterised by
        $\mu_{\text{new}} = \boldsymbol{x}_n^i,\ \sigma_{\text{new}} = \sigma_{\min},\ \alpha_{\text{new}} = 1$;
        add π_{new} to $\boldsymbol{P}_{k+1}$.

The final set of prototypes $\boldsymbol{P}_{k+1}$ represents the new long-term memory of the network and defines the set of kernel functions (and hence the number of hidden units) of the network at the time t_{k+1}. It is worth noting that the above-described algorithm supports the case in which new classes are present in $\boldsymbol{T}_{k+1}$ (generating a new prototype whenever π_{nearest} is not found for a given training sample).

2.4.2. Weights computation

The set of weights $\boldsymbol{W}_{k+1}$ is obtained by using a new error function that consists of the sum of two terms:

$$E_{k+1} = E_{k+1}^{\text{new}} + E_{k+1}^{\text{prot}}. \tag{4}$$

The first term E_{k+1}^{new} is the *sum-of-squares* error function computed on the new training set $\boldsymbol{T}_{k+1}$ according to (2). The second term E_{k+1}^{prot} is a weighed summation over the classification errors computed on the set of the old prototypes $\boldsymbol{P}_k$. It is given by:

$$E_{k+1}^{\text{prot}} = \frac{1}{2}\sum_{i=1}^{C_{k+1}}\sum_{j=1}^{M_k}\left\{\alpha_j^k\left[o_i(\mu_j^k) - T_i(\mu_j^k)\right]^2\right\}, \tag{5}$$

where C_{k+1} is the number of classes considered by the network $\text{Net}_{(t_{k+1})}$, and M_k is the number of prototypes at time t_k. The contribution of the jth prototype to the error function is weighed by the corresponding coefficient α_j^k. It is worth noting that E_{k+1}^{prot} approximates the *sum-of-squares* error function computed on the previously considered training sets, and makes it possible to preserve the existing knowledge approximately without re-examining the old training samples.

3. Experimental results

In order to assess the effectiveness of the proposed incremental learning classifier, different experiments were carried out on a multitemporal data set composed of two remote-sensing images acquired in the same geographical area at two different times. In particular, the data set consists of two multispectral images acquired by the Landsat TM sensor in April and May 1994 in an agricultural area in the basin of the Po river (northern Italy). The available ground truth was used to derive both a training and a test set for each image (see Table 1). The cereals class, which has too small prior probability, was not considered in the experiments. As one can see, some land-cover changes occurred in the study area between the above two dates. This fact allowed the classifier to be tested in rather a complex situation.

Table 1
The classes considered and the related numbers of training and test pixels: (a) April data set; (b) May data set

Land-cover class	Number of pixels	
	Training set	Test set
(a) April data set		
Wood	683	735
Wet rice field	499	322
Bare soil	1034	858
Total	2216	1915
(b) May data set		
Wood	683	735
Wet rice field	1037	900
Dry rice field	341	172
Bare soil	155	108
Total	2216	1915

The main objective of the experiments was to analyse the incremental learning capabilities of the proposed classifier. In particular, two major aspects were investigated: the capabilities of the classifier to acquire new knowledge from a new training set; the ability of the classifier to preserve the existing knowledge during incremental learning. To this end, the proposed classifier was initially trained on the April image (time t_1) and subsequently tested on both the April and May test sets. Afterwards, it was retrained on the May training set (time t_2) and subsequently tested again on both test sets. This allows the performance of the network to be assessed before and after the retraining phase.

In order to provide reference information about the performances attained on the considered data set by other non-parametric classifiers typically used in the context of remote sensing. Table 2 shows the classification accuracies exhibited by a classical RBF classifier (Moody and Darken, 1989) and a k-nearest neighbour classifier (Serpico et al., 1996). It is worth noting that such classifiers do not support incremental learning; therefore, the results given in Table 2 were obtained by training and testing the classifiers on the same image (i.e., training and test in April, and training and test in May).

Table 2
Classification accuracies provided by the classical RBF and the k-nearest neighbour (k-nn) classifiers: (a) training and test on the April data set; (b) training and test on the May data set

Land-cover class	RBF	k-nn
(a) Classification accuracy (%) (training on April, test on April)		
Wood	99.86	100.00
Wet rice field	98.45	99.06
Bare soil	89.86	92.65
Total	95.14	96.55
(b) Classification accuracy (%) (training on May, test on May)		
Wood	100.00	100.00
Wet rice field	98.88	96.22
Dry rice field	88.95	85.46
Corn	48.14	48.15
Total	95.56	93.99

Table 3 presents the class-by-class and overall classification accuracies provided by the proposed classifier at the time t_1 on both the April and May test sets. As expected (since the proposed classifier at the time t_1 behaves as a classical RBF classifier trained on the April image), the overall classification accuracy obtained on the April test set (i.e., 96.13%) is comparable to those provided by the classical classifiers mentioned above (see Table 2). On the other hand, as most of the land-cover classes in the April image are different from those present in the May image, at the time t_1 the network is not able to classify the latter image efficiently. In greater detail, also for the wet-rice-field class, which is one of the two classes present in both the April and May images, the classifier exhibits an unacceptable classification accuracy (61.0%). This is a direct result of the different statistical distributions of such a class in the two images considered (due to the different atmospheric conditions at the acquisition dates, etc.).

At this point, the network was retrained on the May training set. The distance parameter γ that tunes the condition of similarity described in Eq. (3) was fixed at 3.0. Table 4 gives the classification accuracies provided by the proposed classifier after the retraining phase (i.e., at the time t_2) on both the April and May test sets. The overall classification accuracy provided by the proposed classifier on the May image (95.98%) was found to be comparable to those obtained by the classical

Table 3
Classification accuracies obtained on the April and May test sets by the proposed classifier trained on the April image (time t_1)

	Classification accuracy (%) at time t_1 (training on April)	
Land-cover class	April test set	May test set
Wood	100.00	100.00
Wet rice field	98.14	61.00
Bare soil	92.08	–
Dry rice field	–	0.0
Corn	–	0.0
Total	96.13	67.07

Table 4
Classification accuracies obtained on the April and May test sets by the proposed classifier trained on the April image and retrained on the May one (time t_2)

	Classification accuracy (%) at time t_2 (training on April retraining on May)	
Land-cover class	April test set	May test set
Wood	94.15	100.00
Wet rice field	97.83	99.22
Bare soil	94.29	–
Dry rice field	–	89.53
Corn	–	51.86
Total	94.83	95.98

classifiers considered. This confirms the effectiveness of the presented classifier in acquiring new knowledge from a new training set. In particular, it is worth noting that the classification accuracy provided by the proposed classifier on the wet-rice-field class in May significantly improved from 61.0% at the time t_1 to 99.22% after the retraining phase. In addition, the presented classifier exhibited at t_2 an overall classification accuracy on the April image equal to 94.83%, which represents only a slight reduction (1.3%) in the classification accuracy provided at the time t_1 on the same image. This points out the capabilities of the newly developed classifier to preserve the existing knowledge, even if some of the information classes in the new training set differ from those already known to the network.

4. Conclusions

In this paper, an incremental learning classifier for the analysis of remote-sensing images has been proposed. Such a classifier is based on RBF neural networks and exhibits the following characteristics:

1. Capability of performing incremental learning (i.e., the system is able to acquire new knowledge as it becomes available).
2. Capability of approximately preserving the "old" knowledge in each retraining phase (i.e., the acquisition of new knowledge does not result in a significant loss of the existing one).
3. Self-organising architecture (i.e., when new training sets become available, the architecture of the network may be modified to meet the characteristics of the new data).
4. Adaptive structure (i.e., the number and typology of information classes are not subject to any particular limitation and, if necessary, may vary in each retraining phase).

Experimental results reported in the paper have confirmed the effectiveness of the proposed classifier.

Finally, it is worth noting that further research is needed to develop more efficient procedures for the selection and updating of the set of prototypes in each retraining phase.

For further reading, see (Bruzzone and Serpico, 1997; Bruzzone et al., 1999).

References

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Clarendon Press, Oxford.

Broomhead, D.S., Lowe, D., 1988. Multivariate functional interpolation and adaptive networks. Complex Systems 2, 321–355.

Bruzzone, L., Fernàndez Prieto, D., 1999. A technique for the selection of kernel-function parameters in RBF neural networks for classification of remote-sensing images. IEEE Transactions on Geoscience and Remote Sensing 37 (2), 1179–1184.

Bruzzone, L., Fernàndez Prieto, D., Serpico, S.B., 1999. A neural-statistical approach to multitemporal and multisource remote-sensing image classification. IEEE Transactions on Geoscience and Remote Sensing 37 (3) (in press).

Bruzzone, L., Serpico, S.B., 1997. Classification of imbalanced remote-sensing data by neural networks. Pattern Recognition Letters 18 (11–13), 1323–1328.

Fu, L., Hsu, H., Principe, J.C., 1996. Incremental backpropagation learning networks. IEEE Transactions on Neural Networks 7 (3), 757–761.

Park, D.C., El-Sharkawi, M.A., Marks II, R.J., 1991. An adaptively trained neural network. IEEE Transactions on Neural Networks 2 (3), 334–345.

Powell, M.J.D., 1987. Radial basis functions for multivariate interpolation: a review. In: Mason, J.C., Cox, M.G. (Eds.), Algorithms for Approximation, Clarendon Press, Oxford.

Moody, J., Darken, C.J., 1989. Fast learning in networks of locally tuned processing units. Neural Computation 1 (2), 281–294.

Schaal, S., Atkeson, C.G., 1998. Constructive incremental learning from only local information. Neural Computing 10 (8), 2047–2084.

Serpico, S.B., Bruzzone, L., Roli, F., 1996. An experimental comparison of neural and statistical non-parametric algorithms for supervised classification of remote-sensing images. Pattern Recognition Letters 17 (13), 1331–1341.