Available online at www.sciencedirect.com
SCIENCE DIRECT®

Neural Networks 18 (2005) 117–122

Neural
Networks

www.elsevier.com/locate/neunet

Neural Networks Letter

Intelligent initialization of resource allocating RBF networks

Manolis Wallace[a,b,*], Nicolas Tsapatsoulis[b], Stefanos Kollias[b]

[a]Department of Computer Science, University of Indianapolis, Athens Campus, Athens, Greece
[b]Department of Computer Science, School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece

Received 3 August 2004; accepted 22 November 2004

Abstract

In any neural network system, proper parameter initialization reduces training time and effort, and generally leads to compact modeling of the process under examination, i.e. less complex network structures and better generalization. However, in cases of multi-dimensional data, parameter initialization is both difficult and time consuming. In the proposed scheme a novel, multi-dimensional, unsupervised clustering method is used to properly initialize neural network architectures, focusing on resource allocating networks (RAN); both the hidden and output layer parameters are determined by the output of the clustering process, without the need for any user interference. The main contribution of this work is that the proposed approach leads to network structures that are compact, efficient and achieve best classification results, without the need for manual selection of suitable initial network parameters. The efficiency of the proposed method has been tested on several classes of publicly available data, such as iris, Wisconsin and ionosphere data.

Keywords: Wisconsin; Ionosphere; Resource allocating networks

1. Introduction

Resource Allocating Network (RAN) architectures (Platt, 1991), were found to be suitable for online modeling of non-stationary processes. In this sequential learning method the network initially contains no hidden nodes. On incoming training examples, based on two criteria, either the RAN is grown, or the existing network parameters are adjusted using a least mean square gradient descent. The first criterion is based on the prediction error while the second is the novelty criterion. In the cases where hidden neurons are modeled via RBFs, the novelty criterion states that the distance between the observation and the winning RBF neuron should be greater than a threshold. If both criteria are satisfied, then the data is memorized and a new hidden node is added to the network.

Starting without any hidden nodes is highly inefficient, since outliers in the training data may create unnecessary nodes and, therefore, increase both learning effort and convergence time and deteriorate generalization performance. Furthermore, when neural networks are used for rule extraction and especially for the sub-symbolic phase (Apolloni et al., 2000) then proper initialization is absolutely necessary. Unsupervised clustering of the training data provides the means of a successful initialization of RAN architectures that initially contain RBF-type hidden nodes. Clusters can be represented through their mean vector and, either an overall spread (vector spread) or a vector of spreads, corresponding to the spread of elements in each input dimension. Clearly, such kind of parameters can be directly transferred to RBF nodes.

In this paper we consider resource allocating radial basis function (RBF) network architectures that generally consist of three layers: the input layer, containing n neurons, through which an input vector $\underline{x} \in \mathbb{R}^n$ is fed to a hidden layer containing $q(t)$ RBF-type hidden neurons (at iteration t), and an output layer, containing p sigmoid neurons (Lee & Street, 2001). Learning is incorporated into the network using the gradient descent method, while a squared error criterion is

* Corresponding author. Address: Department of Computer Science, School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece.

E-mail addresses: wallace@image.ntua.gr (M. Wallace), ntsap@image.ntua.gr (N. Tsapatsoulis), stefanos@cs.ntua.gr (S. Kollias).

URLs: http://image.ntua.gr/~wallace (M. Wallace), http://image.ntua.gr/~ntsap (N. Tsapatsoulis), http://image.ntua.gr/stefanos (S. Kollias).

0893-6080/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2004.11.005

used for network training. The squared error $e(t)$ at iteration t is computed in the standard way:

$$e(t) = \frac{1}{2}\sum_{k=1}^{p}(d_k(t) - y_k(t))^2 \tag{1}$$

where $d_k(t)$ is the desired output and $y_k(t)$ is the output of neuron k given by:

$$y_k(t) = \frac{1 - e^{2z_k}}{1 + e^{2z_k}},\ z_k = (\underline{w_k})^{\mathrm{T}} \cdot \underline{\phi(t)} \tag{2}$$

where $\underline{w_k} = [w_{k1}, w_{k2}, \ldots, w_{\mathrm{kq}(t)}]^{\mathrm{T}}$ are the weights connecting the RBF hidden neurons with the output neurons and $\underline{\phi(t)}$ is the output of the hidden layer. Each hidden neuron represents a single RBF and computes a kernel function of $\underline{x}$ according to the following equation:

$$\phi_j(t) = \phi_j(x(t)) = \exp\left(-\frac{1}{2}\sum_{i=1}^{n}\left(\frac{x_i(t) - \mu_{ji}}{\sigma_{ji}}\right)^2\right) \tag{3}$$

where $\underline{\mu_j} = [\mu_{j1}, \mu_{j1}, \ldots, \mu_{\mathrm{jn}}]$ and $\underline{\sigma_j} = [\sigma_{j1}, \sigma_{j1}, \ldots, \sigma_{\mathrm{jn}}]$ are the center and spread of the jth hidden neuron, respectively. Training data are supplied to the network in the form of pairs $\underline{x(t)}, \underline{d(t)}$ of input and target vectors. During training, if a new input does not significantly activate any hidden neuron and the prediction error is significantly large, a new neuron is created. If the new input activates at least one of the hidden neurons, or the prediction error is small, the network parameters are updated. The network is initialized by setting the value of $q(0)$, i.e. the count of hidden neurons, $\underline{\mu_j}, \underline{\sigma_j}, j = 1\ldots q(0)$, i.e. the hidden layer parameters, and $\underline{w_k}, k = 1\ldots p$, i.e. the weights connecting the hidden to the output neurons.

In Section 2 we present a methodology for properly initializing networks through an unsupervised clustering procedure. Experimental results are also presented, illustrating the theoretical developments.

2. The proposed clustering and network initialization approach

In order to train the network to converge quickly, without generating unnecessary hidden neurons, the initial parameter values need to be as close as possible to their optimal counterparts; such values may be estimated by clustering available data samples: clusters will correspond to high density areas in the input space, thus providing the information needed for the formation and initialization of the corresponding hidden neurons. As the count of meaningful clusters in the training data, and equivalently the optimal number of hidden layer neurons, is typically not known beforehand, partitioning methods are inapplicable; an agglomerative clustering approach needs to be utilized (Theodoridis & Koutroumbas, 1998).

The generic agglomerative approach utilizes a metric that quantifies the distance between clusters of data samples; this metric is produced based on a metric that quantifies the distance between individual data samples. Having a unique metric to measure distances between data samples is not compatible with the notion of input area density in real life data, where different spreads have to be considered for each one of the n input features. Moreover, when the input space has more than one dimension, an aggregating distance function, such as Euclidean distance, is typically used as the distance metric, which is not always meaningful; cases exist, in which the ‘context’ can change the metric to be used. In such cases, a selection of distance function among samples needs to be performed, prior to calculating the distance among clusters (Wallace & Kollias, 2004).

In this paper, we extend the classic agglomerative clustering algorithm in order to incorporate soft feature selection in the inter cluster distance estimation process, thus providing an output that is more effective for initializing the network. To achieve this we tackle feature selection based on the following principle: while we expect data samples of a given set to have random distances from one another according to most features, we expect them to have small distances according to the features that relate them. In the following, we rely on this difference in distribution of distance values in order to identify the context, i.e. the features that most probably relate a set of data samples.

More formally, let c_1 and c_2 be two clusters of data samples. Let also $r_i, i = 1\ldots F$ be a distance metric defined in space $\mathbb{R}^{S_i} \subseteq \mathbb{R}^S$, F the count of distinct metrics that may be defined among a pair of clusters, S the count of features for the data samples and S_i the count of features considered by the ith sample-to-sample distance metric. A distance metric between the two clusters, when considering the ith sample-to-sample distance metric, is given by

$$f_i(c_1, c_2) = \sqrt[\kappa]{\frac{\sum_{a\in c_1, b\in c_2}(r_i(\underline{a_i}, \underline{b_i}))^{\kappa}}{|c_1|\cdot|c_2|}} \tag{4}$$

where $\underline{a_i}, \underline{b_i}$ are the positions of data samples a and b in feature space $\mathbb{R}^{S_i}$ and $|c_1|$, $|c_2|$ are the cardinalities of clusters c_1 and c_2 respectively and $\kappa \in \mathbb{R}$ is a constant. Adjusting the value of κ Eq. (4) can be transformed into most of the classic agglomerative clustering metrics (Yager, 2000). For example, for $\kappa \to -\infty$ Eq. (4) approaches the min operator, for $\kappa \to +\infty$ Eq. (4) approaches the max operator, for $\kappa = 1$ Eq. (4) estimates the mean value, for $\kappa = 2$ Eq. (4) becomes a Euclidian distance based metric and so on.

The context is selection of features that should be considered when calculating an overall distance value; we define it as a vector $\underline{ctx} \in \mathbb{R}_+^F$ with $\sum_{i=1}^{F} ctx_i = 1$. Then, the overall distance between clusters c_1 and c_2 is calculated as

$$f^*(c_1, c_2) = \sum_{i=1}^{F}(ctx_i(c_1, c_2))^{\lambda}\cdot f_i(c_1, c_2) \tag{5}$$

where $\lambda \in \mathbb{R}$ is a constant and ctx_i is the degree to which f_i is included in the context. The optimal context can be calculated as the context that produces the best (smallest) overall distance f^*:

□ When $\lambda=1$ the feature $i_* \in \{1 \ldots F\}$ for which $f_{i_*}(c_1, c_2) = \min_{i=1}^{F} f_i(c_1, c_2)$ is the only one included in the context. If more than one feature satisfies the above, they are all included to the same degree.

□ When $\lambda \neq 1$ and $\exists i_* \in \{1 \ldots F\}$: $f_i(c_1,c_2)=0$, then i is the only feature included in the context. Again, if multiple such features exist, they are equally included.

□ In the non-trivial cases the optimal context is provided by the following theorem.

Theorem. *If* $\lambda \neq 1$ *and* $f_i(c_1,c_2) \neq 0 \, \forall i \in \{1 \ldots F\}$, *then the optimal context is given by*:

$$ctx_i(c_1,c_2) = ctx_F(c_1,c_2) \cdot \left(\frac{f_F(c_1,c_2)}{f_i(c_1,c_2)}\right)^{\frac{1}{\lambda-1}}, \quad i = 1 \ldots F-1 \tag{6}$$

$$ctx_F(c_1,c_2) = \frac{1}{\sum_{i=1}^{F} \left(\frac{f_F(c_1,c_2)}{f_i(c_1,c_2)}\right)^{\frac{1}{\lambda-1}}} \tag{7}$$

Proof. We start by substituting $ctx_F(c_1,c_2)$ for

$$ctx_F(c_1,c_2) = 1 - \sum_{i=1}^{F-1} ctx_i(c_1,c_2) \tag{8}$$

in Eq. (5), thus turning the minimization of $f^*(c_1,c_2)$ into an $(F-1)$-parameter unconstrained optimization problem. Demanding that

$$\frac{\partial f^*(c_1,c_2)}{\partial ctx_i(c_1,c_2)} = 0, \quad i = 1 \ldots F-1,$$

we have

$$\frac{\partial (ctx_F(c_1,c_2))^{\lambda} \cdot f_F(c_1,c_2)}{\partial ctx_i(c_1,c_2)} + \frac{\partial (ctx_i(c_1,c_2))^{\lambda} \cdot f_i(c_1,c_2)}{\partial ctx_i(c_1,c_2)} = 0, \quad i = 1 \ldots F-1 \tag{9}$$

From Eq. (8) it is easy to calculate that

$$\frac{\partial (ctx_F(c_1,c_2))^{\lambda} \cdot f_F(c_1,c_2)}{\partial ctx_i(c_1,c_2)} = -\lambda \cdot (ctx_F(c_1,c_2))^{\lambda-1} \cdot f_F(c_1,c_2), \quad i = 1 \ldots F-1 \tag{10}$$

applying which Eq. (9) is transformed into

$$(ctx_F(c_1,c_2))^{\lambda-1} \cdot f_F(c_1,c_2) = (ctx_i(c_1,c_2))^{\lambda-1} \cdot f_i(c_1,c_2), \quad i = 1 \ldots F-1 \tag{11}$$

The proof of Eq. (6) starting from Eq. (11) is straightforward through simple term rearrangement, while Eq. (7) is proven by considering Eq. (8). □

The fact that the above theorem provides an analytical solution to the optimization problem means that the proposed approach can be incorporated in the hierarchical agglomerative process without augmenting its computational complexity.

In order for distances to be used during clustering it is imperative that they are transformed as to become directly comparable to each other, even when different contexts are used for different pairs of clusters. Therefore, the following metric is finally used:

$$f(c_1,c_2) = \frac{f_*(c_1,c_2)}{\mathrm{adj}(c_1,c_2)} \tag{12}$$

where

$$\mathrm{adj}(c_1,c_2) = \sum_{i \in \mathbb{N}_F} (ctx_i(c_1,c_2))^{\lambda} \tag{13}$$

As far as the termination criterion is concerned, a threshold on the growth rate of $f(c_1, c_2)$ is used. Once the clustering algorithm terminates, the count of detected clusters is used to determine the count $q(0)$ of initial hidden neurons. The centers $\mu_{j\_}$ of the hidden RBF neurons are obtained directly from the centers of the created clusters, while the spreads $\sigma_{j\_}$ are set based on the clusters' standard deviations in each feature. Weights $w_{k\_}$ are determined by considering the way data samples of detected clusters are mapped to output classes. Specifically, if *per%* of the samples of cluster j belong to class k, then the corresponding hidden neuron is linked to the class's output neurons with a weight of $w_{\mathrm{kj}} = per/100$.

3. Experimental results

In order to prove the efficiency of the proposed approach, in this section we present experimental results of its application to the iris, breast cancer and ionosphere datasets, all publicly available at UCI Repository of machine learning databases.

The iris data set contains 150 elements characterized by 4 features and belonging to three classes; two of these classes are not linearly separable from each other. For testing purposes, a part of the set was used as training data (20% of the data, i.e. 30 randomly selected elements, were used for clustering and training of the resulting RAN, while all 150 elements were used for testing). Four different experiments were carried out for the iris data set:

1. The network was initialized based on the results of the proposed clustering, and was not trained.
2. The network was initialized with three random hidden neurons and then trained.

Table 1
MSE as a function of epochs for the iris data

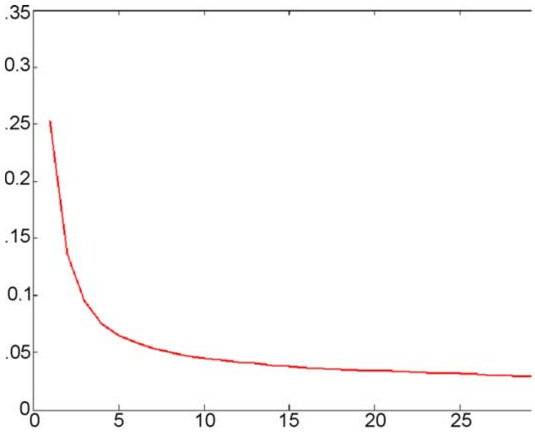

Figure 1. Random initialization

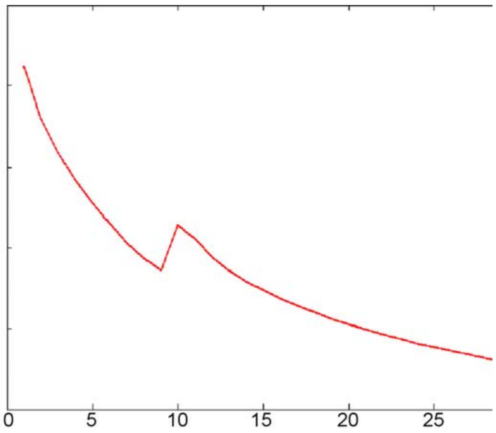

Figure 2. Bayesian initialization

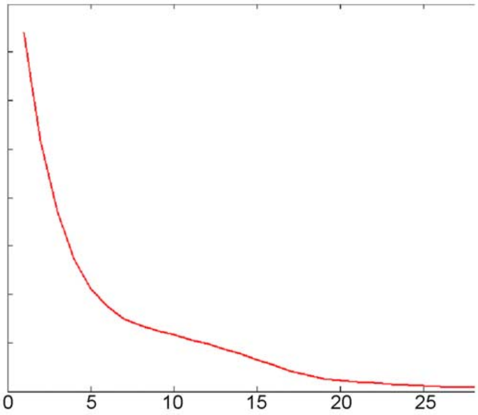

Figure 3. Proposed initialization

3. Labels were used to partition training data in three clusters, corresponding to the three existing classes, and the network was first initialized based on these clusters and then trained.
4. The approach described in this paper was used both for clustering of available data and network initialization, before training the network.

As can be seen in Table 1, where the mean square error is presented as a function of the number of epochs, random initialization of the network quickly reaches an upper bound of performance, as far as the MSE is concerned, which is probably caused by a local minimum that the network is unable to overcome. Properly initialized approaches, on the other hand, progress much better, indicating that proper initialization is imperative. In Table 1, where the MSE on the training set is considered, the Bayesian approach seems to outperform the proposed approach, as has a smaller MSE in both the initial epochs and after the termination of the training. Still, the difference between the two, although starting from considerable values, rapidly approaches zero after few epochs of training. More importantly, as can be seen in Table 2, where results from application on the whole data set are presented, the proposed approach generates a more efficient neural classifier, as it achieves better classification rate on the test data, while at the same time having a smaller number of hidden layer neurons. The reason is that the proposed unsupervised clustering technique is more efficient in detecting the patterns that underlie in the data, thus providing better network initialization information.

In Table 3 we present a few of the best results reported in the literature for the iris data set. (Countless more can be found, as the iris data set is a data set that almost all classification papers that refer to the UCI repository use as a first simple example.) We can see that one of the two

Table 2
Classification rates and counts of hidden nodes for the iris data

	No training	Random init	Bayesian init.	Proposed approach
Number of rules	3	6	5	3
Classification rate (%)	87.3	96	97.3	98

Table 3
Comparative study on the iris data

Method	No of neurons/rules	Classification rate (%)
Proposed approach	3	98
Pertselakis et al. (2003)	3	98
Paul and Kumar (2002)	5	100

These results are reported in Paul and Kumar (2002) and Pertselakis et al. (2003).

presented approaches, when using 5 rules, reaches 100% precision. This is actually accomplished by numerous works in the literature (with all other approaches using more than 5 rules) which is the main reason that results presented on the iris data set are typically not considered sufficient to assess the performance of a classification methodology. The main problem with the iris data set is that due to the fact that it contains too few data samples, it is not easy to partition it into training and test data. Re-substitution is used, i.e. all the data are used for training and then the same data are used for testing purposes. In the experimental application of our methodology we have used only 20% of the available data and using 3 clusters we have equalled the performance of the best known classifier that uses only 3 rules; the latter achieves this performance through re-substitution.

The proposed approach has also been applied on the Wisconsin breast cancer database, which contains 699 samples, characterized by 10 attributes, all assuming integer values in (Apolloni et al., 2000; Wallace & Kollias, 2004). 65.5% of the data samples belong to the benign class and 34.5% to the malignant class. 16 samples are incomplete (an attribute is missing) and have been excluded from the database for the application of our algorithm. For testing purposes, a part of the set was used as training data (50 randomly selected elements, i.e. 7.32% of the data, were used for clustering and training of the resulting RAN, while the remaining 633 elements were used for testing). Table 4 presents a comparative study between the proposed approach and other results reported in the literature for the Wisconsin breast cancer database. The classifier generated using the proposed approach achieves a classification rate similar to the best performing other method (Nauk & Kruse, 1997), while displaying a considerably more compact modeling (2 neurons compared to 7 or more).

Table 4
Comparative study on the Wisconsin data

Method	No of neurons/rules	Classification rate (%)
Nauk and Kruse (1997)	7	96.7
Proposed approach	2	96.6
K-NN	200 samples	96.34
Bagui, Bagui, Pal, and Pal (2003)	200 samples	96.17
Halgamuge and Glesner (1994)	7	96
Kasabov (1996)	9	95.3
Kasabov and Woodford (1999)	17	95.3

These results are reported in Bagui et al. (2003), Halgamuge and Glesner (1994), Kasabov (1996), Kasabov and Woodford (1999), and Nauk and Kruse (1997).

Table 5
Comparative study on the ionosphere data

Method	Classification rate (%)
Proposed approach	96
C4.5	94.9
RIAC	94.6
GALE	94
SVM	93.2

These results are reported in Barry, Holmes, and Llora (2004), Bennett and Blue (1997), and Hamilton et al. (1996).

Similar results are observed in other data sets as well. Especially for the ionosphere data set, which is clearly a high dimensional data set (it is characterized by 34 features), simpler initialization approaches, such as the Bayesian approach followed for the iris data set, or the selection of random initial parameter values, have proven to be totally ineffective; using them the hidden layer is rapidly populated with large numbers of neurons and this overfitting leads to poor performance on test data. The proposed initialization leads to the generation of a very efficient classifier of 10 hidden neurons. Comparative results are presented in Table 5. In order to produce the results for the proposed methodology 250 points have been utilized for training and 101 data points for testing. Results of other methods reported in other works in the literature and cited here have been acquired using either leave-one-out methodology (350 training, 1 test) or ten-fold cross validation (316 training, 35 test). Similarly to the other data sets, we can see that the proposed methodology can lead to the generation of networks with high efficiency demanding lesser training effort.

4. Conclusions

In this paper, we apply a novel agglomerative clustering algorithm for the initialization of a RAN in order to properly initialize the network parameters and especially the RBF neurons of the hidden layer. Proper initialization serves two purposes: (a) reduces the learning effort, (b) keeps the number of hidden neurons low, thus being capable of achieving good generalization performance. The classification performance of the proposed network turns out to be excellent, while using a compact problem representation. When initiated with three hidden neurons, based on the results of the proposed clustering approach, it outperforms the majority of the soft-computing schemes that were tested on the iris classification problem. It performs similarly in other data sets as well, especially as the dimensionality of the input space increases.

The experimental results provided in this paper clearly establish that the proposed methodology is very promising. Currently, we are working towards the application of the RAN structure and the initialization methodology presented in this paper for the extraction and tracking of semantic user preferences, in the framework of intelligent information and multimedia retrieval.

Acknowledgements

This work has been partially funded by the HERACLETUS national grant for basic research. This work has been partially funded by the EU IST-1999-20502 FAETHON project. This work has been partially funded by the ORESTEIA IST-2000-26091/TBD project.

References

Apolloni, B., Orovas, C., Taylor, J., Fellenz, W., Gielen, S., & Westerdijk, M. (2000). A general framework for symbol and rule extraction in neural networks. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*.

Bagui, S. C., Bagui, S., Pal, K., & Pal, N. R. (2003). Breast cancer detection using rank nearest neighbor classification rules. *Pattern Recognition*, *36*, 25.

Barry, A. M., Holmes, J., Llora, X. (2004). Data mining using learning classifier systems. In: L. Bull (Ed.), *Applications of learning classifier systems. Springer-Verlag LNAI Series*.

Bennett K. P., Blue J. (1997). *A support vector machine approach to decision trees*. R.P.I Math Report No. 97-100, Rensselaer Polytechnic Institute, Troy, NY.

Halgamuge, S., & Glesner, M. (1994). Neural Networks in designing fuzzy systems for real world applications. *Fuzzy Sets and Systems*, *65*, 1–12.

Hamilton H. J., Shan N., Cercone N. (1996). *RIAC: a rule induction algorithm based on approximate classification*, Technical report CS 96-06, Regina University.

Kasabov, N. (1996). Learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems. *Fuzzy Sets and Systems*, *82*, 135–149.

Kasabov, N., & Woodford, B. (1999). Rule insertion and rule extraction from evolving fuzzy neural networks: Algorithms and applications for building adaptive, intelligent, expert systems. *Proceedings of FUZZ-IEEE'99* .

Lee, K., & Street, W. N. (2001). Intelligent image analysis using adaptive resource allocating networks. *Proceedings of the IEEE International Workshop on NN for Signal*.

Nauk, D., & Kruse, R. (1997). A neuro-fuzzy method to learn fuzzy classification rules from data. *Fuzzy sets and Systems*, *8*, 277–288.

Paul, S., & Kumar, S. (2002). Subsethood-product fuzzy neural inference system (SuPFuNIS). *IEEE Transactions on Neural Networks*, *13*(3), 578–599.

Pertselakis M., Tsapatsoulis N., Kollias S., Stafylopatis A. (2003). An Adaptive resource allocating neural fuzzy inference system. In: *Proceedings of IEEE Intelligent Systems Application to Power Systems (ISAP'03), Limnos*.

Platt, J. (1991). A resource-allocating network for function interpolation. *Neural Computing*, *3*, 213–225.

Theodoridis, S., & Koutroumbas, K. (1998). *Pattern Recognition*. New York: Academic Press.

UCI Repository of machine learning databases. Irvine, CA. University of California, Department of Information and Computer Science. [http://www.ics.uci.edu/(mlearn/MLRepository.html]

Wallace, M., & Kollias, S. (2004). Robust, generalized, quick and efficient agglomerative clustering. *Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS), Porto, Portugal*.

Yager, R. R. (2000). Intelligent control of the hierarchical agglomerative clustering process. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, *30*(6), 835–845.