

# A Growing and Pruning Method for Radial Basis Function Networks

M. Bortman, M. Aladjem

Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev

P.O.Box 653, Beer-Sheva, 84105, Israel.

**Abstract** – A recently published learning algorithm GGAP for *radial basis function* (RBF) neural networks is studied and modified. GGAP is a growing and pruning algorithm, which means that a created network unit that consistently makes little contribution to the network's performance can be removed during the training. GGAP states a formula for computing the significance of the network units, which requires a  $d$ -fold numerical integration for arbitrary probability density function  $p(\mathbf{x})$  of the input data  $\mathbf{x}$  ( $\mathbf{x} \in \mathbf{R}^d$ ). In this work the GGAP formula is approximated using a *Gaussian mixture model* (GMM) for  $p(\mathbf{x})$  and an analytical solution of the approximated unit significance is derived. This makes it possible to employ the modified GGAP for input data having complex and high dimensional  $p(\mathbf{x})$ , which was not possible in the original GGAP. The results of an extensive experimental study show that the modified algorithm outperforms the original GGAP achieving both a lower prediction error and reduced complexity of the trained network.

**Index Terms** – Radial basis function neural networks, sequential function approximation, growing and pruning algorithms, extended Kalman filter, Gaussian mixture model.

## I. INTRODUCTION

*Radial basis function* (RBF) neural networks are well established as able to approximate complex nonlinear mappings. Recently many algorithms have been proposed for training these networks - support vector machines [1] - [3], relevance vector machines [4], [5], orthogonal least squares algorithms [6] - [8], recursive least-square-based algorithms [9], [10] and others. In some practical applications new

training data becomes available from time to time and in these cases sequential (incremental) learning algorithms [11] - [27] are generally preferred over batch algorithms because they do not require retraining for the new data.

In this work a recent *generalized growing and pruning* (GGAP) algorithm [18] for sequential learning RBF networks is studied and modified. GGAP is a *resource - allocating network* (RAN) algorithm, which means that a created network unit that consistently make little contribution to the network's performance can be removed during the training. The original contribution in the GGAP with respect to other RAN algorithms [19] - [22] is a proposed formula for computing the significance of the network units, which results in a significant reduction in the number of the units. This formula involves a  $d$ -fold integration of an expression using the probability density function  $p(\mathbf{x})$  of the input data  $\mathbf{x}$  ( $\mathbf{x} \in \mathbf{R}^d$ ). For an arbitrary  $p(\mathbf{x})$  this requires a numerical integration, which in practice is only possible for small input dimension ( $d \leq 5$ )<sup>1</sup>. In order to overcome the problem of  $d$ -fold integration the GGAP separately computes the unit significance for each attribute (dimension). However this is only accurate for independent attributes of the input data and leads to a fall off in performance when used on complex  $p(\mathbf{x})$ . In this work the unit significance is approximated using a *Gaussian mixture model* (GMM) for  $p(\mathbf{x})$ . This enables us to approximate any  $p(\mathbf{x})$  to arbitrary accuracy [23, page 111]. In addition using the GMM we derive an analytical solution of the unit significance. This allows us to apply the GGAP on data sets having a large dimension and complex  $p(\mathbf{x})$ , which was not possible in the original GGAP [18].

The paper is organized as follows. In Section II we give a brief overview of GGAP [18] and derive an expression for the threshold of the GGAP significance criterion, which overcomes a confusing interpretation of the threshold stated in [18]. In Section III.A a GMM approximation of the unit significance is proposed and a closed - form expression of the approximated significance is derived followed by an explanation

---

<sup>1</sup> <http://www.nag.co.uk/>

of the modified GGAP algorithm in Section III.B. The results of an extensive experimental study presented in Section IV show that the modified GGAP (Section III.B) outperforms the original GGAP [18], achieving both a lower prediction error and a trained network with a reduced complexity.

## II. GGAP ALGORITHM [18]

GGAP [18] is an RBF approximation algorithm in which the training points  $(\mathbf{x}_n, y_n)$  are presented sequentially (one - by - one) and disregarded after being used<sup>2</sup>. Here  $y_n$  is the desired output (target) of the network for a  $d$  - variable input vector  $\mathbf{x}_n$  ( $\mathbf{x}_n \in \mathbf{R}^d$ ).

Assume that an RBF network with  $S$  units has been obtained at the  $n$ -stage of the training. For an arbitrary input vector  $\mathbf{x} = (x_1, x_2, \dots, x_d)^T$  the network output  $f^{(n)}(\mathbf{x})$  is a linear combination of the RBF responses

$$f^{(n)}(\mathbf{x}) = \sum_{j=1}^S \omega_j \phi_j(\mathbf{x}) + \omega_0. \quad (1)$$

Here  $\phi_j(\mathbf{x}) = \exp\left(-\|\mathbf{x} - \boldsymbol{\mu}_j\|^2 / \sigma_j^2\right)$  is an RBF,  $\|\cdot\|$  denotes  $L^2$  - norm,  $\boldsymbol{\mu}_j$  ( $\boldsymbol{\mu}_j \in \mathbf{R}^d$ ) is a prototype vector and  $\sigma_j^2$  is a positive parameter defining the spread of the RBF. If the  $J$ th unit is removed then the output of the network with the remaining  $S-1$  units is  $f_*^{(n)}(\mathbf{x}) = \sum_{j=1}^{J-1} \omega_j \phi_j(\mathbf{x}) + \sum_{j=J+1}^S \omega_j \phi_j(\mathbf{x}) + \omega_0$ , which results in an error  $E(J, \mathbf{x}) = |f^{(n)}(\mathbf{x}) - f_*^{(n)}(\mathbf{x})| = |\omega_J| \phi_J(\mathbf{x})$ . In [18] the *significance of the  $J$ th unit* is defined as a  $L^2$  - norm<sup>3</sup> of  $E(J, \mathbf{x})$ , weighted by the input density function  $p(\mathbf{x})$  i.e.

$$E_{sig}(J) = |\omega_J| \left( \int_{\mathbf{R}^d} \exp\left(-2\|\mathbf{x} - \boldsymbol{\mu}_J\|^2 / \sigma_J^2\right) p(\mathbf{x}) d\mathbf{x} \right)^{1/2}. \quad (2)$$

The GGAP performs the following sequential updating of the network:

---

<sup>2</sup> The GGAP algorithm can be used sequentially only when the input data distribution  $p(\mathbf{x})$  is known a priori. Unfortunately, for most real-world problems  $p(\mathbf{x})$  is unknown [23] and must be approximated in advance by using some pre-history data set [18], [24] - [27].

<sup>3</sup> It should be noticed that in [18] the significance of the units has been derived for  $L^q$  - norm, for any value of  $q$ . Here we consider the  $L^2$  - norm in order to be consistent with previous RAN algorithms [19] - [22]. The modified algorithm (Section III) can be extended to  $L^q$  - norm straightforwardly.

- A unit is initiated if *significance* and *growth criteria* are fulfilled:

The *significance criterion*, as originally proposed in [18] is

$$|e_n| \left( \int_{\mathbf{R}^d} \exp\left(-2\|\mathbf{x} - \mathbf{x}_n\|^2 / \left(\lambda^2 \|\mathbf{x}_n - \boldsymbol{\mu}_{nr}\|^2\right)\right) p(\mathbf{x}) d\mathbf{x} \right)^{1/2} > E_{\min}^{GGAP}, \quad (3)$$

and the *growth criterion* (conventional RAN criterion [19, Section II])

$$\|\mathbf{x}_n - \boldsymbol{\mu}_{nr}\| > \varepsilon_n. \quad (4)$$

In (3)  $e_n = y_n - f^{(n)}(\mathbf{x}_n)$  is the prediction error of the approximation,  $\boldsymbol{\mu}_{nr} = \arg \min_{\boldsymbol{\mu}_j} \|\mathbf{x}_n - \boldsymbol{\mu}_j\|$  is the nearest prototype vector to  $\mathbf{x}_n$ ,  $\lambda$  is a user-supplied parameter which determines the overlap of the RBFs and  $E_{\min}^{GGAP}$  is a significance threshold. In (4)  $\varepsilon_n$  denotes the scale of resolution which decays exponentially during network training. When a new unit is added, the parameters associated with it are calculated by the original RAN algorithm [19, Section II]:  $\omega_{s+1} = e_n$ ,  $\boldsymbol{\mu}_{s+1} = \mathbf{x}_n$  and  $\sigma_{s+1} = \lambda \|\mathbf{x}_n - \boldsymbol{\mu}_{nr}\|$ . The criterion (3) ensures that the significance (2) of the newly added unit is greater than a threshold  $E_{\min}^{GGAP}$ , and criterion (4) - that the new unit is sufficiently far from the existing prototype vectors.

- GGAP does not add a unit when one or both criteria (3), (4) are not fulfilled. In this case the RBF of the unit having prototype vector  $\boldsymbol{\mu}_{nr}$  is updated by an *extended Kalman filter* (EKF) iteration [20, expr. (6.2)]. After this if (3) is not fulfilled the latter unit is removed.

The GGAP [18] replaces the classical RAN *prediction error criterion*  $|e_n| = |y_n - f(\mathbf{x}_n)| > e_{\min}$  with (3). In classical RAN [19] the threshold  $e_{\min}$  defines the *desired approximation accuracy* of the network. In GGAP papers [18, Section IV], [24, Section III], [25, Section 3], [26, Section 3], [27, Section II], a confusing setting  $E_{\min}^{GGAP} = e_{\min}$  was stated. Observing the expression under the integral in (3) we conclude that it strongly depends on the range of input data  $\mathbf{x}$ . Denoting the averaged value of the integral (the expectation over  $\mathbf{x}_n$ ) by  $\eta$  and using the significance criterion (3) which implies  $|e_n| > e_{\min}$  [20, page 961], we find that  $E_{\min}^{GGAP}$  is a scaled version of  $e_{\min}$

$$E_{\min}^{GGAP} = \eta e_{\min}. \quad (5)$$

In Section IV we define empirically a suitable range  $0.03 \leq \eta \leq 0.1$  for normalized data sets having input attributes  $0 \leq x_k \leq 1$ ,  $k = 1, 2, \dots, d$  and target  $0 \leq y \leq 1$ .

The computation of  $E_{sig}(j)$  (2) is an open problem in GGAP algorithms. In [24], [25]  $E_{sig}(j)$  (2) is calculated analytically for uniformly distributed  $p(\mathbf{x})$ . In [18, Section II]  $E_{sig}(j)$  is computed separately for  $x_k$ ,  $k = 1, 2, \dots, d$ . Unfortunately in [27] it is shown that these simplifications lead to degradation of the performance for complex  $p(\mathbf{x})$ . In next section a modification of GGAP is proposed, which overcomes these restrictions on  $p(\mathbf{x})$ .

### III. A MODIFIED GGAP ALGORITHM

As mentioned above a critical part of GGAP algorithm [18] is the computation of  $E_{sig}(j)$ . Here a modification of the GGAP (named GGAP-GMM algorithm) is proposed based on a GMM approximation of  $E_{sig}(j)$  derived in Section III.A. In Section III.B the GGAP-GMM algorithm is presented.

#### A. GMM Approximation of $E_{sig}(j)$

This approximation requires a GMM modeling of input density  $p(\mathbf{x})$

$$\hat{p}(\mathbf{x}) = \sum_{i=1}^M \alpha_i N(\mathbf{x}; \mathbf{m}_i, \Sigma_i), \quad (6)$$

where  $N(\mathbf{x}; \mathbf{m}_i, \Sigma_i)$  is a  $d$ -variate Gaussian density function with mean vector  $\mathbf{m}_i$  and covariance matrix  $\Sigma_i$ , and  $\alpha_i$  are mixing coefficients which are nonnegative and sum to one. As in other GGAP algorithms [18], [24] - [27], before starting the sequential training we require  $N_{pre-history}$  input data points. The parameters ( $\alpha_i, \mathbf{m}_i, \Sigma_i$  and  $M$ ) of the GMM are computed using these points.

In the following explanation an analytical computation of  $d$ -fold integral (2) for  $p(\mathbf{x})$  modeled by GMM (6) is derived. For this purpose the first term under the integral in (2) is presented in the form  $\exp\left(-2\|\mathbf{x} - \boldsymbol{\mu}_j\|^2 / \sigma_j^2\right) = (\pi\sigma_j^2/2)^{d/2} N(\mathbf{x}; \boldsymbol{\mu}_j, \mathbf{I}\sigma_j/2)$  where  $\mathbf{I}$

is a  $d \times d$  identity matrix. Substituting the latter expression into (2) and replacing  $p(\mathbf{x})$  by  $\hat{p}(\mathbf{x})$  (6) we obtain the following approximation for  $E_{sig}(j)$ :

$$\hat{E}_{sig}(j) = |\omega_j| \left[ \left( \pi \sigma_j^2 / 2 \right)^{d/2} \sum_{i=1}^M \alpha_i \int_{\mathbf{R}^d} N(\mathbf{x}; \boldsymbol{\mu}_j, \mathbf{I} \sigma_j / 2) N(\mathbf{x}; \mathbf{m}_i, \Sigma_i) d\mathbf{x} \right]^{1/2}. \quad (7)$$

The above integrals have a closed-form solution [28, page 101]

$$\int_{\mathbf{R}^d} N(\mathbf{x}; \boldsymbol{\mu}_j, \mathbf{I} \sigma_j / 2) N(\mathbf{x}; \mathbf{m}_i, \Sigma_i) d\mathbf{x} = N(\boldsymbol{\mu}_j - \mathbf{m}_i; \mathbf{0}, \mathbf{I} \sigma_j / 2 + \Sigma_i). \quad (8)$$

Finally substituting (8) into (7) we obtain  $\hat{E}_{sig}(j)$  by direct matrix calculations

$$\hat{E}_{sig}(j) = |\omega_j| \left( \left( \pi \sigma_j^2 / 2 \right)^{d/2} \mathbf{N}_j^T \mathbf{A} \right)^{1/2}, \quad (9)$$

where  $\mathbf{A} = (\alpha_1, \dots, \alpha_M)^T$  and

$$\mathbf{N}_j = \left( N(\boldsymbol{\mu}_j - \mathbf{m}_1; \mathbf{0}, \mathbf{I} \sigma_j / 2 + \Sigma_1), N(\boldsymbol{\mu}_j - \mathbf{m}_2; \mathbf{0}, \mathbf{I} \sigma_j / 2 + \Sigma_2), \dots, N(\boldsymbol{\mu}_j - \mathbf{m}_M; \mathbf{0}, \mathbf{I} \sigma_j / 2 + \Sigma_M) \right)^T$$

### B. GGAP Algorithm Using $\hat{E}_{sig}(j)$ (9)

Table 1 presents the GGAP-GMM algorithm. All steps, apart from those which are labeled with an asterisk, are identical to those used in the original GGAP algorithm [18, Section III].

TABLE 1. GGAP-GMM algorithm

(*1) Compute a GMM $\hat{p}(\mathbf{x})$ (6), based on $N_{pre-history}$ observations.
Set the desired approximation accuracy $e_{min}$ .
Set initial parameters of the network: number S of the network units and values of $\omega_j, \boldsymbol{\mu}_j, \sigma_j$ of the units. <sup>4</sup>
(*2) Compute $E_{min}^{GGAP} = \eta e_{min}$ , $\eta$ is a user supplied parameter.
<b>For</b> each observation $(\mathbf{x}_n, y_n)$ presented to the network, do:
1. Calculate the parameters of the significance (3) and growth (4) criteria $\varepsilon_n = \max \left\{ \varepsilon_{max} \cdot \gamma^n, \varepsilon_{min} \right\}, (0 < \gamma < 1)$ $e_n = y_n - f^{(n)}(\mathbf{x}_n)$ for $\varepsilon_{min}, \varepsilon_{max}, \gamma$ user supplied parameters and $f^{(n)}(\mathbf{x}_n)$ the current network having S number of units.
2. Compute parameters for potential new (S+1)th unit: $\omega_{S+1} = e_n, \quad \boldsymbol{\mu}_{S+1} = \mathbf{x}_n, \quad \sigma_{S+1} = \lambda \ \mathbf{x}_n - \boldsymbol{\mu}_{nr}\ .$

<sup>4</sup> Different methods can be used for the initialization of the network. In the experiments (Section IV) we used [20, p. 963].

<p>*3. Compute <math>\hat{E}_{sig}(S+1)</math> (9).</p>
<p>4. Update the network</p> <p style="padding-left: 2em;"><b>If</b> <math>\ \mathbf{x}_n - \boldsymbol{\mu}_{nr}\  &gt; \varepsilon_n</math> (<i>growth criterion</i> (4)) and <math>\hat{E}_{sig}(S+1) &gt; E_{min}^{GGAP}</math> (<i>GGAP significance criterion</i>) are fulfilled add new unit (<math>S=S+1</math>).</p> <p style="padding-left: 2em;"><b>Else</b></p> <p style="padding-left: 4em;">Adjust the parameters <math>[\omega_{nr}, \boldsymbol{\mu}_{nr}^T, \sigma_{nr}]</math> of the nearest unit <math>nr</math> to <math>\mathbf{x}_n</math> by EKF iteration [20, expr. (6.2)].</p> <p style="padding-left: 4em;">Compute <math>\hat{E}_{sig}(nr)</math> using (9).</p> <p style="padding-left: 2em;"><b>If</b> <math>\hat{E}_{sig}(nr) &lt; E_{min}^{GGAP}</math> (<i>GGAP significance criterion</i> is not fulfilled)</p> <p style="padding-left: 4em;">Remove the <math>nr</math>-th unit.</p> <p style="padding-left: 2em;"><b>Endif</b></p> <p style="padding-left: 2em;"><b>Endif</b></p>
<b>EndFor</b>

GGAP-GMM algorithm (Table 1) differs from the original GGAP [18, Section III] in the following important points:

(\*1) GGAP-GMM models input density  $p(\mathbf{x})$  by GMM (6), which can represent complex  $p(\mathbf{x})$  and overcomes the restrictions on  $p(\mathbf{x})$  required in previous GGAP algorithms [18], [24] - [27]. In recent years significant progress in GMM density estimation methods has been achieved [23, Chapters 9-12], [29] - [33]. In the experiments (Section IV) the method [30] is used, which finds the number  $M$  of the GMM components and the values of the parameters  $(\alpha_i, \mathbf{m}_i, \boldsymbol{\Sigma}_i)$  simultaneously.

(\*2) We compute the threshold for the significance criterion by the expression  $E_{min}^{GGAP} = \eta e_{min}$  derived in Section II. Here  $e_{min}$  has the clear meaning of desired approximation accuracy and  $\eta$  is in the range  $0.03 \leq \eta \leq 0.1$  for data sets having input attributes  $0 \leq x_k \leq 1$ ,  $k = 1, 2, \dots, d$  and target  $0 \leq y \leq 1$  (see Section IV.A).

\*3  $\hat{E}_{sig}(S+1)$  is computed by direct matrix calculation using a closed-form solution of the integrals (8), which makes GGAP more accurate and quick.

## IV. EXPERIMENTAL STUDY

In this section an extensive experimental study of the GGAP on artificial and real world data sets is presented. These data sets have been carefully chosen in order to represent a wide class of situations covering various dimensionalities of the observations and smoothnesses of the underlying target functions.

Following the scenario of other experimental studies of RAN algorithms [18, Section IV], [21, Section 3.1], [22], [24, Section IV], [26, Section 4], [27, Section III], we normalize the attributes of the data sets into the range  $[0, 1]$  and the values of the GGAP user-supplied parameters have been set using the guidelines in [18, Section IV]:  $\varepsilon_{\max} = 1.15$ ,  $\varepsilon_{\min} = 0.04$ ,  $\gamma = 0.999$  and  $\lambda = 0.1$ . In GGAP-GMM algorithm  $0.03 \leq \eta \leq 0.1$  is used, which is obtained empirically in Section IV.A. In Section IV.B the results from a comparative study of GGAP-GMM (Section III) and the original GGAP [18] are reported.

### A. Empirical Setting the Range of GGAP - GMM Parameter $\eta$

In this section we describe how we found that the range  $0.03 \leq \eta \leq 0.1$  is suitable for data sets having input attributes  $0 \leq x_k \leq 1$ ,  $k = 1, 2, \dots, d$  and target  $0 \leq y \leq 1$ .

We designed seven synthetic data sets  $A - G$  covering a wide class of situations with different smoothnesses of the underlying target functions and different *minimal root mean square errors* (MRMSEs) of the targets. In Appendix A a detailed description of the data sets is given. In summary, they have the following characteristics:

- Data set  $A$  has a smooth underlying target function and  $\text{MRMSE} = 0.0782$ .
- Data sets  $B - F$  have smooth underlying target function (which is the same as those for  $A$ ) and the following MRMSEs: 0.03, 0.05, 0.0782, 0.09, 0.11.
- Data set  $G$  has a highly variable underlying target function and  $\text{MRMSE} = 0.0711$  (which is almost the same as those for  $A$ ).

The data sets  $A - G$  comprise 15000 data points and have been divided randomly fifteen times into  $N_{train} = 10000$  training points  $(\mathbf{x}_n^{train}, y_n^{train})$  and  $N_{test} = 5000$  test points  $(\mathbf{x}_n^{test}, y_n^{test})$ .

GGAP-GMM (Section III) was run on these data sets. The known analytical expressions of the input densities  $p_A(\mathbf{x})$  and  $p_G(\mathbf{x})$  for the data sets (see Appendix A) are exploited as  $\hat{p}(\mathbf{x})$  for GGAP-GMM (the best setting for the algorithm). The desired approximation accuracy  $e_{min}$  was chosen to be equal to the MRMSEs of the targets (the best setting for the EKF estimators [34, Chapter 1]). The values of  $\eta$  have been varied and the resulting test root mean square error (RMSE)

$RMSE_{test} = \sqrt{\sum_{n=1}^{N_{test}} \|f(\mathbf{x}_n^{test}) - y_n^{test}\|^2 / N_{test}}$  is computed, where  $f(\mathbf{x}_n^{test})$  is the output of the trained network.

The obtained results are presented in Tables 2 and 3. The first columns of the tables comprise the names of the data sets and the parameters  $\hat{p}(\mathbf{x})$ ,  $e_{min}$ ,  $\eta$  of GGAP-GMM. In the last columns the means and standard deviation of the  $RMSE_{test}$ , and the number of units for the trained networks over the fifteen replications of GGAP-GMM are reported. The suitable values of  $\eta$  are indicated in bold in the tables. These values result in a small number of units (low complexity of the trained network) and a small differences  $e_{min} - RMSE_{test}$  (low prediction error of the network).

Table 2 comprises the results of a study aimed to find suitable values of  $\eta$  for data sets  $A$  and  $G$  having almost the same MRMSEs and different smoothness of the underlying target functions. Observing the lines in bold we conclude that the range  $\eta = 0.05-0.1$  is suitable for  $A$  (having a smooth underlying target function), while a highly variable underlying target function of  $G$  needs smaller values for  $\eta$  ( $\eta = 0.03-0.05$ ).

TABLE 2. Test RMSEs and number of units for the trained networks on data sets  $A$  and  $G$  having different smoothness of underlying target function using various values for  $\eta$ .

Data Set	Parameters of the GGAP-GMM algorithm (Section III)			RMSE <sub>test</sub>		Number of Units	
	$\hat{p}(\mathbf{x})$	$e_{\min} = MRMSE$	$\eta$	Mean	Std	Mean	Std
<b>A</b> (smooth underlying target function)	$p_A(\mathbf{x})$	0.0782	0.9	0.1824	0.0117	1.6667	0.488
			0.5	0.0947	0.0011	2	0.6547
			0.2	0.0919	0.0018	5	0.7559
			<b>0.1</b>	<b>0.0888</b>	<b>0.0013</b>	<b>10.6</b>	<b>1.35</b>
			<b>0.05</b>	<b>0.0873</b>	<b>0.0009</b>	<b>18.8</b>	<b>2.62</b>
			0.03	0.0885	0.0015	30.73	3.67
<b>G</b> (highly variable underlying target function)	$p_G(\mathbf{x})$	0.0711	0.2	0.089	0.0019	4.6667	0.9795
			0.1	0.0863	0.0011	7.7333	0.9612
			<b>0.05</b>	<b>0.0844</b>	<b>0.001</b>	<b>16.4</b>	<b>2.35</b>
			<b>0.03</b>	<b>0.0842</b>	<b>0.0011</b>	<b>28.6667</b>	<b>4.0999</b>
			0.01	0.0883	0.0011	93.8	4.2122

Table 3 presents the results of a study the sensitivity of  $\eta$  to various MRMSEs. Here we use data sets  $B - F$ , having smooth underlying target function (the same as those for  $A$ ) and different MRMSEs. In this experiment  $\eta=0.1$  is used, which has been found to be suitable for  $A$  in previous study (see Table 2). The obtained results show that by keeping  $\eta=0.1$  a similar number of units and low prediction errors ( $e_{\min} - RMSE_{test}$ ) of the trained networks are obtained for  $B - F$ . This indicates that the appropriate value of  $\eta$  is not sensitive to the MRMSEs and depends only on the smoothness of the underlying target function.

TABLE 3. Test RMSEs and number of units for the trained networks on  $B - F$  having a smooth underlying target function and different MRMSEs.

Data Set	Parameters of the GGAP-GMM algorithm (Section III)			RMSE <sub>test</sub>		Number of Units	
	$\hat{p}(\mathbf{x})$	$\eta$	$e_{\min} = MRMSE$	Mean	Std	Mean	Std
<b>B</b>	$p_A(\mathbf{x})$	<b>0.1</b>	<b>0.03</b>	<b>0.0432</b>	<b>0.0012</b>	<b>13.8</b>	<b>1.97</b>
<b>C</b>			<b>0.05</b>	<b>0.0608</b>	<b>0.0009</b>	<b>11.2</b>	<b>1.3202</b>
<b>D</b>			<b>0.0782</b>	<b>0.0868</b>	<b>0.0019</b>	<b>9.7333</b>	<b>1.1629</b>
<b>E</b>			<b>0.09</b>	<b>0.0983</b>	<b>0.0011</b>	<b>10.0667</b>	<b>1.3345</b>
<b>F</b>			<b>0.11</b>	<b>0.1163</b>	<b>0.001</b>	<b>9.6667</b>	<b>1.1127</b>

### B. Comparative Study of GGAP-GMM and GGAP Algorithm [8]

In this section the results from a comparative study of GGAP-GMM algorithm

(Section III) and the original GGAP algorithm [18] using real world data sets<sup>5</sup> are reported. Table 4 shows the characteristics of the data sets:  $N$  denotes the available number of observations,  $N_{train}$  - the number of training observations,  $N_{test}$  - the number of test observations and  $d$  - the number of attributes (dimension of the observations). In the last column the target feature name is presented. As previously the available observations have been divided randomly fifteen times into training and test observations.

TABLE 4. Characteristics of data sets

Data Set Name	N	$N_{train}$	$N_{test}$	d	Target Feature
California Housing	20640	8000	12640	9	House price
Abalone	4177	3000	1177	8	Number of rings
Weather Ankara	1609	1100	509	10	Mean temperature
House-16H	22784	15000	7784	17	House price
Auto-mpg	398	320	78	8	Fuel consumption (miles per gallon)

Table 5 summarizes the parameters of GGAP-GMM and GGAP [18] algorithms and the results obtained over the fifteen replications of the algorithms. The values of the parameters in Table 5 are set empirically by a trial and error procedure.

TABLE 5. Test RMSEs and number of units for trained networks on real-world data sets

Data set	Algorithm	Parameters	RMSE <sub>test</sub>		Number of Units	
			Mean	Std	Mean	Std
California Housing	GGAP	$E_{min}^{GGAP}=0.0001, *N_{pre-history}=20640$	0.1477	0.0031	25.2	2.4832
	GGAP-GMM	$e_{min}=0.1, \eta=0.1, N_{pre-history}=100$	0.1424	0.0027	18.8	3.3582
Abalone	GGAP	$E_{min}^{GGAP}=0.00008, *N_{pre-history}=4177$	0.0910	0.0024	8.2	1.32
	GGAP-GMM	$e_{min}=0.08, \eta=0.1, N_{pre-history}=100$	0.0850	0.0027	5.13	0.83
Weather Ankara	GGAP	$E_{min}^{GGAP}=0.00002, *N_{pre-history}=1609$	0.0610	0.0097	9.5333	1.8459
	GGAP-GMM	$e_{min}=0.06, \eta=0.03, N_{pre-history}=100$	0.0611	0.0075	5.2	1.8593
House-16H	GGAP	$E_{min}^{GGAP}=0.00008, *N_{pre-history}=22784$	0.0973	0.002	17.4	1.5024
	GGAP-GMM	$e_{min}=0.09, \eta=0.09, N_{pre-history}=100$	0.0965	0.0026	10.4667	1.6591
Auto-mpg	GGAP	$E_{min}^{GGAP}=0.00007, *N_{pre-history}=398$	0.1162	0.0156	2.8	0.7746
	GGAP-GMM	$e_{min}=0.11, \eta=0.06, N_{pre-history}=100$	0.1167	0.0134	3.6	0.7368

\*The original GGAP [18, Section IV] uses the available observations for fitting the univariate densities of the attributes. In the modified algorithm (Section III) the GMM  $\hat{p}(\mathbf{x})$  (6), having diagonal component covariance matrices, is fitted by method [30] for 100 training points.

<sup>5</sup> <http://funapp.cs.bilkent.edu.tr/DataSets/>  
[http://www.niaad.liacc.up.pt/~ltorgo/Regression/cal\\_housing.html](http://www.niaad.liacc.up.pt/~ltorgo/Regression/cal_housing.html)

Observing the results in Table 5 it is concluded that GGAP-GMM (Section III) outperforms the original GGAP algorithm [18], achieving both lower  $RMSE_{test}$  and a smaller number of units for most data sets.

## VI. CONCLUSIONS

A modification of a recent growing and pruning learning algorithm GGAP [18] for *radial basis function* (RBF) neural networks is proposed. GGAP states a formula for computing the significance  $E_{sig}(j)$  (2) of the network units, which involves a d-fold integration of an expression using the probability density function  $p(\mathbf{x})$  of the input data  $\mathbf{x}$  ( $\mathbf{x} \in \mathbf{R}^d$ ). The computation of  $E_{sig}(j)$  (2) is an open problem in GGAP algorithms [18], [24] - [27]. In Section III a GMM approximation of  $E_{sig}(j)$  is derived and a modified algorithm (named GGAP-GMM) is proposed which differs from the original GGAP in the following important points:

1. GGAP-GMM models input density  $p(\mathbf{x})$  by GMM (6), which can represent complex  $p(\mathbf{x})$  and overcomes the restrictions on the input density assumed in GGAP algorithms [18], [24] - [27].
2. GGAP-GMM computes the significance threshold by the expression  $E_{min}^{GGAP} = \eta e_{min}$  derived in Section II. This overcomes a confusing setting  $E_{min}^{GGAP} = e_{min}$  used in GGAP algorithms [18], [24] - [27].
3. GGAP-GMM computes the network unit significance by direct matrix calculations using a closed-form solution of the integrals (8). This increases the accuracy and decreases the computational complexity of GGAP-GMM (Section III) in comparison with the original GGAP.

The results of the experimental study (Section IV) show that GGAP-GMM (Section III) outperforms the original GGAP [18] achieving both a lower prediction error and the trained network with a reduced complexity (number of units).

## APPENDIX A

### SYNTETIC DATA SETS A - G

These data sets cover a wide class of situations with different smoothness of the underlying target function and different MRMSEs of the targets. We designed the data sets using the following mean vectors and covariance matrices [35]:

$$\begin{aligned}
 \tilde{\mathbf{m}}^{(1)} &= [5.835, 8.525, 6.615, 7.065, 7.865, 4.435]^T, \\
 \tilde{\mathbf{m}}^{(2)} &= [5.980, 3.975, 9.020, 14.685, 10.640, 4.175]^T, \\
 \tilde{\mathbf{m}}^{(3)} &= [5.850, 4.365, 6.340, 4.675, 6.260, 4.440]^T,
 \end{aligned}
 \quad
 \tilde{\Sigma}^{(1)} = \begin{bmatrix}
 7.138 & 1.192 & 2.726 & 1.116 & 0.678 & 2.097 \\
 & 2.269 & 1.367 & 0.146 & 0.201 & -0.308 \\
 & & 5.727 & 1.280 & 0.933 & 2.107 \\
 & & & \tilde{\Sigma}_{\mathbf{xx}}^{(1)} & 2.941 & 1.949 \\
 & & & & 1.577 & 2.197 \\
 \hline
 & & & & & \tilde{\Sigma}_{\mathbf{yx}}^{(1)} \\
 & & & & & 6.606
 \end{bmatrix},
 \tag{10}$$

$$\tilde{\Sigma}^{(2)} = \begin{bmatrix}
 2.500 & 2.834 & -0.665 & -2.621 & 0.248 & 1.738 \\
 & 4.704 & -0.629 & -2.913 & 0.576 & 2.471 \\
 & & 19.000 & 0.896 & 8.622 & -0.254 \\
 & & & \tilde{\Sigma}_{\mathbf{xx}}^{(2)} & 5.856 & 1.357 \\
 & & & & 1.357 & -2.915 \\
 & & & & 20.800 & -0.622 \\
 \hline
 & & & & & \tilde{\Sigma}_{\mathbf{yx}}^{(2)} \\
 & & & & & 3.214
 \end{bmatrix},
 \quad
 \tilde{\Sigma}^{(3)} = \begin{bmatrix}
 6.117 & 2.525 & 1.321 & 1.501 & 1.274 & 2.191 \\
 & 4.432 & 2.481 & 2.179 & 1.080 & 1.784 \\
 & & 2.134 & 2.325 & 1.017 & 1.030 \\
 & & & \tilde{\Sigma}_{\mathbf{xx}}^{(3)} & 4.099 & 2.019 \\
 & & & & 4.099 & 1.803 \\
 & & & & 1.872 & 2.081 \\
 \hline
 & & & & & \tilde{\Sigma}_{\mathbf{yx}}^{(3)} \\
 & & & & & 3.806
 \end{bmatrix}.$$

Each of the sets A - G comprises 15000 data points  $(\mathbf{x}_n, y_n)$  having attributes into the range  $[0, 1]$ .

1) *Data set A having smooth underlying target function  $E_A[y|\mathbf{x}]$  (13) and  $MRMSE=0.0782$ .*

This data set is generated as follows. First points  $(\tilde{x}_1^{(n)}, \tilde{x}_2^{(n)}, \tilde{x}_3^{(n)}, \tilde{x}_4^{(n)}, \tilde{x}_5^{(n)}, \tilde{y}_n)^T$  for  $n = 1, 2, \dots, 15000$  are drawn from GMM density

$$\tilde{p}_A(\mathbf{x}, y) = \sum_{i=1}^3 \alpha_i N(\mathbf{x}, y; \tilde{\mathbf{m}}^{(i)}, \tilde{\Sigma}^{(i)}) \tag{11}$$

with  $\alpha_1 = \alpha_2 = 0.3$ ,  $\alpha_3 = 0.4$  and  $\tilde{\mathbf{m}}^{(i)}, \tilde{\Sigma}^{(i)}$  (10). The values of  $\tilde{x}_1^{(n)}, \tilde{x}_2^{(n)}, \tilde{x}_3^{(n)}, \tilde{x}_4^{(n)}, \tilde{x}_5^{(n)}, \tilde{y}_n$  are normalized to the range  $[0, 1]$  resulting in points  $(\mathbf{x}_n^A, y_n^A)$  drawn from the density

$p_A(\mathbf{x}, y) = \sum_{i=1}^3 \alpha_i N(\mathbf{x}, y; \mathbf{m}^{(i)}, \Sigma^{(i)})$  where

$$\begin{aligned}
 \mathbf{m}^{(i)} &= \tilde{\mathbf{H}}^{-1} (\tilde{\mathbf{m}}^{(i)} - \tilde{\mathbf{K}}^{(\min)}), \quad \Sigma^{(i)} = \tilde{\mathbf{H}}^{-1} \tilde{\Sigma}^{(i)} \tilde{\mathbf{H}}^{-1}, \quad \tilde{\mathbf{K}}^{(\min)} = (\tilde{x}_1^{(\min)}, \dots, \tilde{x}_5^{(\min)}, \tilde{y}^{(\min)})^T, \\
 \tilde{\mathbf{H}} &= \text{diag}(\tilde{x}_1^{(\max)} - \tilde{x}_1^{(\min)}, \dots, \tilde{x}_5^{(\max)} - \tilde{x}_5^{(\min)}, \tilde{y}^{(\max)} - \tilde{y}^{(\min)})
 \end{aligned}
 \tag{12}$$

and  $\tilde{x}_k^{(\min)}$ ,  $\tilde{x}_k^{(\max)}$ ,  $\tilde{y}^{(\min)}$ ,  $\tilde{y}^{(\max)}$  are the minimal and maximal values of the original data points. For  $p_A(\mathbf{x}, y)$  the conditional expectation and variance of the target  $y$  for given input  $\mathbf{x}$  are

$$E_A[y|\mathbf{x}] = (1/p_A(\mathbf{x})) \sum_{i=1}^3 \alpha_i N(\mathbf{x}; \mathbf{m}_x^{(i)}, \Sigma_{xx}^{(i)}) \left[ m_y^{(i)} + \Sigma_{yx}^{(i)} (\Sigma_{xx}^{(i)})^{-1} (\mathbf{x} - \mathbf{m}_x^{(i)}) \right] \quad (13)$$

and

$$\text{Var}_A(y|\mathbf{x}) = (1/p_A(\mathbf{x})) \sum_{i=1}^3 \alpha_i N(\mathbf{x}; \mathbf{m}_x^{(i)}, \Sigma_{xx}^{(i)}) \left[ \Sigma_{yy}^{(i)} - \Sigma_{yx}^{(i)} (\Sigma_{xx}^{(i)})^{-1} \Sigma_{xy}^{(i)} \right], \quad (14)$$

where  $\mathbf{m}_x^{(i)}$ ,  $m_y^{(i)}$  and  $\Sigma_{xx}^{(i)}$ ,  $\Sigma_{yy}^{(i)}$  are the marginal means and covariances of  $\mathbf{x}$ ,  $y$ ;  $\Sigma_{yx}^{(i)}$  a vector comprising the cross-covariances of  $\mathbf{x}$  and  $y$ , and  $p_A(\mathbf{x}) = \sum_{i=1}^3 \alpha_i N(\mathbf{x}; \mathbf{m}_x^{(i)}, \Sigma_{xx}^{(i)})$ .

The least RMSE estimator of  $y$  given  $\mathbf{x}$  is  $E_A[y|\mathbf{x}]$  (13) resulting in

$$\text{MRMSE}_A = \sqrt{\int_{\mathbf{R}^d} \text{Var}_A[y|\mathbf{x}] p_A(\mathbf{x}) d\mathbf{x}} = \sqrt{\sum_{i=1}^3 \alpha_i \left[ \Sigma_{yy}^{(i)} - \Sigma_{yx}^{(i)} (\Sigma_{xx}^{(i)})^{-1} \Sigma_{xy}^{(i)} \right]}, \quad (15)$$

which is 0.0782 for the values defined in (10).

2) *Data sets B - F having smooth underlying target function  $E_A[y|\mathbf{x}]$  (13) and the following MRMSEs: 0.03, 0.05, 0.0782, 0.09, 0.11.*

These data sets are modification of A. Here the target values  $\tilde{y}_n^B = E_A[y|\mathbf{x}] + v_n^B, \dots, \tilde{y}_n^F = E_A[y|\mathbf{x}] + v_n^F$  are computed using the underlying target function  $E_A[y|\mathbf{x}]$  (13) and adding zero mean Gaussian noise having different variances. The variances of  $v_n^B, \dots, v_n^F$  are set to values which imply MRMSEs 0.03, 0.05, 0.0782, 0.09, 0.11 for the targets of B - F.

3) *Data set G having highly variable underlying target function  $f_G(\mathbf{x})$  (18) and MRMSE=0.0711 (which is almost the same as those for A).*

Here we draw vectors  $\tilde{\mathbf{x}}_n^G$ ,  $n=1, 2, \dots, 15000$  from a nine component GMM density

$$\tilde{p}_G(\mathbf{x}) = \sum_{i=1}^9 \beta_i N(\mathbf{x}; \tilde{\mathbf{m}}_x^{(i)}, \tilde{\Sigma}_{xx}^{(i)}), \quad (16)$$

where  $\tilde{\mathbf{m}}_x^{(1)}, \tilde{\mathbf{m}}_x^{(2)}, \tilde{\mathbf{m}}_x^{(3)}$  and  $\tilde{\Sigma}_{xx}^{(1)}, \tilde{\Sigma}_{xx}^{(2)}, \tilde{\Sigma}_{xx}^{(3)}$  are the marginal means and covariance matrices of (10),  $\tilde{\mathbf{m}}_x^{(3+i)} = \tilde{\mathbf{m}}_x^{(i)} + 3$ ,  $\tilde{\Sigma}_{xx}^{(3+i)} = \tilde{\Sigma}_{xx}^{(i)}$ ,  $\tilde{\mathbf{m}}_x^{(6+i)} = \tilde{\mathbf{m}}_x^{(i)} + 6$ ,  $\tilde{\Sigma}_{xx}^{(6+i)} = \tilde{\Sigma}_{xx}^{(i)}$ , for

$i = 1, 2, 3$ , and  $\beta_1 = \beta_2 = \beta_4 = \beta_5 = \beta_7 = \beta_8 = 0.1$  and  $\beta_3 = \beta_6 = \beta_9 = 0.133$ . Then, as previously, the attributes of the vectors  $\tilde{\mathbf{x}}_n^G$  are normalized to the range  $[0, 1]$  resulting in input vector  $\mathbf{x}_n^G$  having a density  $p_G(\mathbf{x}) = \sum_{i=1}^9 \beta_i N(\mathbf{x}; \mathbf{m}_x^{G(i)}, \Sigma_{\mathbf{xx}}^{G(i)})$ . Finally we define a target

$$\tilde{y}_n^G = f_G(\mathbf{x}) + \nu_n \quad (17)$$

with a highly variable underlying target function

$$f_G(\mathbf{x}) = \sum_{i=1}^9 (-1)^{i-1} \beta_i N(\mathbf{x}; \mathbf{m}_x^{G(i)}, \Sigma_{\mathbf{xx}}^{G(i)}). \quad (18)$$

The variance of the zero mean Gaussian noise  $\nu_n$  was set to a value which implies  $\text{MRMSE} = 0.0711$  (for normalized targets).

## Acknowledgments

The authors wish to thank Editor-in-Chief Prof. Marios M. Polycarpou, the associate editor and the reviewers for their critical reading of the manuscript and helpful comments. This work was supported in part by the Paul Ivanier Center for Robotics and Production Management, Ben-Gurion University of the Negev, Israel.

## References

- [1] C. Cortes and V. N. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273-297, 1995.
- [2] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, "The Entire Regularization Path for the Support Vector Machine", *The Journal of Machine Learning Research*, Vol. 5, pp.1391-1415, 2004.
- [3] B. Li, X. Li, Z. Zhao, "Novel algorithm for constructing support vector machine regression ensemble", *Journal of Systems Engineering and Electronics*, Vol. 17, pp. 541-545, 2006.
- [4] M. E. Tipping, "The Relevance Vector Machine", In S. A. Solla, T. K. Leen, and

- K.-R. Muller, editors, *Advances in Neural Information Processing Systems*, Vol. 12, pp. 652-658, MIT Press, 2000.
- [5] C. M. Bishop, M. E. Tipping, "Variational relevance vector machines", In C. Boutilier and M. Goldszmidt, editors, *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pp. 46-53, Morgan Kaufmann, 2001.
- [6] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modeling using orthogonal forward regression with press statistic and regularization", *IEEE Trans. Syst. Man. Cybern.- Part B: Cybernetics*, Vol. 34, pp. 898-911, 2004.
- [7] S. Chen, "Local regularization assisted orthogonal least squares regression", *Neurocomputing*, Vol. 69, pp. 559-585, 2006.
- [8] M. J. L. Orr, "Regularization in the selection of radial basis function centers", *Neural Computation*, Vol. 7, pp. 606-623, 1995.
- [9] C. S. Leung, G. H. Young, J. Sum, W. Kan, "On the regularization of Forgetting Recursive Least Square", *IEEE Trans. on Neural Networks*, Vol. 10, No. 6, pp. 1482-1486, 1999.
- [10] C. S. Leung, K. W. Wong, P. F. Sum, L. W. Chan, "A pruning method for the recursive least squared algorithm", *Neural Networks*, Vol. 14, pp. 147-174, 2001.
- [11] G. A. Carpenter, S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine", *Computer Vision, Graphics, and Image Processing*, Vol. 37, 54-115, 1987.
- [12] G. A. Carpenter, S. Grossberg, "ART 2: Self-organization of Stable Category Recognition Codes for Analog Input Patterns", *Applied Optics*, Vol.26, pp. 4919-4930, 1987.
- [13] S. Grossberg, "A theory of human memory: Self-organization and performance of sensory-motor codes, maps and plans", In R. Rosen and F. Snell editors, *Progress in theoretical biology*, Vol. 5, pp. 233-374, Academic, 1978.
- [14] C. Constantinopoulos, A. Likas, "An Incremental Training Method for the Probabilistic RBF Network", *IEEE Trans. on Neural Networks*, Vol. 17, No. 4, pp. 966-974, 2006.

- [15] X. Hong, “A Fast Identification Algorithm for Box-Cox Transformation Based Radial Basis Function Neural Network”, *IEEE Trans. on Neural Networks*, Vol. 17, No. 4, pp. 1064-1069, 2006.
- [16] S. Ferrari, M. Jensenius, “A constrained optimization approach to preserving prior knowledge during incremental training”, *IEEE Trans. on Neural Networks*, Vol. 19, No. 6, pp. 996-1009, 2008.
- [17] S. Ozawa, S. Pang, N. Kasabov, “Incremental Learning of Chunk data for online pattern classification systems”, *IEEE Trans. on Neural Networks*, Vol. 19, No. 6, pp. 1-14, 2008.
- [18] G.-B. Huang, P. Saratchandran, N. Sundararajan, “A Generalized growing and pruning RBF (GGAP - RBF) neural network for function approximation”, *IEEE Trans. on Neural Networks*, Vol. 16, No. 1, pp. 57 – 67, 2005.
- [19] J. Platt, “A resource – allocating network for function interpolation”, *Neural Computation*, Vol. 3, pp. 213 – 225, 1991.
- [20] V. Kadiramanathan, M. Niranjan, “ A function estimation approach to sequential learning with neural networks”, *Neural Computation*, Vol. 5, pp. 954 – 975, 1993.
- [21] N. Sundararajan, P. Saratchandran, L. Yingwei, *Radial Basis Function Neural Networks with Sequential Learning: MRAN and Its Applications*, World Scientific, Singapore, 1999.
- [22] L. Yingwei, N. Sundararajan, P. Saratchandran, “A sequential learning scheme for function approximation using minimal radial basis function (RBF) neural networks”, *Neural Computation*, Vol. 9, pp. 461 – 478, 1997.
- [23] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
- [24] G.-B. Huang, P. Saratchandran, N. Sundararajan, “An Efficient Sequential Learning Algorithm for Growing and Pruning RBF (GAP-RBF) Networks”, *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 34, No 6, pp. 2284 – 2292, 2004.

- [25] G.-B. Huang, P. Saratchandran, N. Sundararajan, “A Recursive Growing and Pruning RBF (GAP-RBF) Algorithm for Function Approximations”, *The 4-th International Conference on Control and Animation, Montreal*, 2003.
- [26] R. Zhang, G.-B. Huang, N. Sundararajan, P. Saratchandran, “Improved GAP-RBF network for classification problems”, *Neurocomputing* 70, pp. 3011 – 3018, 2007.
- [27] R. Zhang, N. Sundararajan, G.-B. Huang, P. Saratchandran, “An efficient sequential RBF network for bio-medical classification problems”, *Proceedings of the International Joint Conference on Neural Networks*, Budapest, pp. 2477-2482, 2004.
- [28] M.P. Wand and M.C. Jones, *Kernel Smoothing*, Charman & Hall/CRC, 1995.
- [29] M. Aladjem, "Projection Pursuit Mixture Density Estimation", *IEEE Trans. on Signal Processing*, Vol. 53, No. 11, pp. 4376-4383, 2005.
- [30] M. A. T. Figueiredo and A. Jain, “Unsupervised learning of finite mixture models”, *IEEE Trans.on Pattern Anal. Mach. Intell.*, Vol. 24, pp. 381-396, 2002.
- [31] D. Bohning, W. Seidel, M. Alfo, B. Garel, V. Patilea, G. Walther, “Advances in Mixture Models”, *Computational Statistics &Data Analysis*, Vol. 51, pp. 5205 – 5210, 2007.
- [32] C. Constantinopoulos and A. Likas, “Unsupervised Learning of Gaussian Mixtures Based on Variational Component Splitting”, *IEEE Trans. on Neural Networks*, Vol. 18, No. 3, pp. 745 - 755, 2007.
- [33] G.J. McLachlan, D. Peel, R.W. Bean, “Modelling high-dimensional data by mixtures of factor analyzers”, *Computational Statistics & Data Analysis*, Vol. 41, pp. 379 – 388, 2003.
- [34] S. Haykin, *Kalman Filtering and Neural Networks*, John Wiley & Sons, 2001.
- [35] T. Marill and D. M. Green, “On the effectiveness of receptors in recognition systems”, *IEEE Trans. on Information Theory*, Vol. 9, No. 1, pp. 11-17, 1963.