Input Selection for Radial Basis Function Networks by Constrained Optimization

Jarkko Tikka

Helsinki University of Technology, Laboratory of Computer and Information Science, P.O. Box 5400, FI-02015 HUT, Finland
tikka@mail.cis.hut.fi, http://www.cis.hut.fi/tikka

Abstract. Input selection in the nonlinear function approximation is important and difficult problem. Neural networks provide good generalization in many cases, but their interpretability is usually limited. However, the contributions of input variables in the prediction of output would be valuable information in many real world applications. In this work, an input selection algorithm for Radial basis function networks is proposed. The selection of input variables is achieved using a constrained cost function, in which each input dimension is weighted. The constraints are imposed on the values of weights. The proposed algorithm solves a log-barrier reformulation of the original optimization problem. The input selection algorithm was applied to both simulated and benchmark data and obtained results were compelling.

1 Introduction

Radial basis function (RBF) networks have been widely utilized in regression problems. The advantages of RBF networks are that the training of networks is relatively fast and they are capable on universal approximation with non-restrictive assumptions [1]. Fastness of the training is a consequence of simple structure of the RBF networks. They have only one hidden layer, in which each node corresponds to a basis function and a mapping from the hidden layer to the output layer is linear. The activation of hidden node is evaluated by the distance between an input vector and a center of the basis function. The usual choice for the basis function is the radially symmetric Gaussian function.

Many approaches are proposed to optimize the number of basis functions and widths of the Gaussian functions. In [2], the widths are fixed to be same and centers of the basis functions are selected from the input vectors using a regularized forward selection. Unsupervised clustering techniques, such as k-means and Gaussian mixture models trained with the EM algorithm, are another alternative to determine the centers of basis functions [3,4]. After the centers are selected, the widths of the Gaussian functions are found, for example, using the p-nearest neighbor rule [3] or weighting the standard deviation of the data in each cluster [5]. Nevertheless, these studies do not consider the input selection at all.

The disadvantage of RBF networks is their black-box characteristics. Basically, the network includes all the input variables and, in addition, importances of the inputs are not clear at all. However, interpretation or understanding of the underlying process can be increased by selecting the input variables. In addition to interpretability, the rejection of non-informative inputs can improve the generalization capability of the network [6].

Several approaches exist to perform input selection [6]. In the filter approach, the input selection procedure is independent from the final non-linear model. The inputs can be selected, for example, based on mutual information [7] or linear models [8,9], and the final non-linear model is trained using the selected subset of inputs. The wrapper methodology takes into account the nonlinear model in the input selection [10]. Time consuming strategy is to fix the number of input variables and search through all the possible combinations. Computationally more efficient search strategies are presented in [10]. In the embedded methods, the input selection and training of the model are carried out simultaneously [6]. For instance, in the case of classification and support vector machines, a radius margin bound is used as a cost function and weights of the inputs are optimized by gradient descent algorithm [11]. Another alternative is to built the model using all the available inputs and use the backward elimination of the least significant inputs. In [12], several different criteria for the ranking of inputs are suggested.

In this work, an input selection algorithm for RBF networks is proposed. The algorithm is based on the weighted Euclidean distance, thus each input dimension has its own weight. The sum of weights are constrained such that some of the weights tend to be zero and corresponding inputs are rejected from the final model. The optimal weights are calculated by solving a constrained optimization problem, which takes into account the non-linear dependency between the inputs and the output. The proposed algorithm belongs to the class of embedded input selection methods.

The article is organized as follows. The ordinary RBF networks are briefly presented in Sect. 2 followed by the input selection algorithm for RBF networks in Sect. 3. The results of experiments on a simulated data set and Boston Housing benchmark data set are shown in Sect. 4. Finally, conclusions are given in Sect. 5.

2 Radial Basis Function Networks

Let us assume that there are N measurements available from an output variable y_j and input variables $\boldsymbol{x}_j = [x_{j,1}, \ldots, x_{j,d}]$, $j = 1, \ldots, N$. In a regression problem, the task is to estimate the values of output y_j as accurately as possible using the inputs $\boldsymbol{x}_j$. If the dependency is presumed to be non-linear, an unknown function can be estimated using artificial neural networks. In the case of RBF networks with Gaussian basis functions the model can be written as

$$\hat{y}_j = \sum_{n=1}^{N} \alpha_n K(\boldsymbol{c}_n, \boldsymbol{x}_j) + \alpha_0, \text{ where } K(\boldsymbol{c}_n, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{c}_n - \boldsymbol{x}_j\|^2}{\sigma_n^2}\right) \quad . \tag{1}$$

Usually, the training of RBF networks consists of three stages. First, the centers of Gaussian basis functions $\boldsymbol{c}_n$ are placed using some unsupervised learning algorithm. Second, the widths of basis functions σ_n are computed. Third, the parameters α_0 and $\alpha_n, n = 1, \ldots, N$ are estimated by minimizing the mean squared error (MSE). However, in this work the basis function is placed on each training data point $\boldsymbol{x}_n, n = 1, \ldots, N$ and the widths of Gaussian basis functions have common value $\sigma_n = \sigma$. The parameters α_0 and $\alpha_n, n = 1, \ldots, N$ are estimated by minimizing the regularized cost function

$$J = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2 + \gamma \sum_{n=1}^{N} \alpha_n^2 \ , \tag{2}$$

where the second term controls the smoothness of nonlinear mapping. For the given values of σ and γ the parameters α_0 and α_n are found by solving the system of linear equations.

3 Input selection for RBF Networks

The disadvantage of model (1) is that it includes all the available input variables. In addition, it is nearly impossible to distinguish irrelevant and relevant inputs from each other. In [13], the problem is circumvented by using a Mahalanobis-like distance in place of the Euclidean distance. The distance is evaluated using a genetic algorithm in order to minimize the error criterion of the network. Nevertheless, the approach does not necessarily select inputs.

In this work, a weighted Euclidean distance is used

$$d_w(\boldsymbol{c}_n, \boldsymbol{x}_j) = \sqrt{\sum_{i=1}^{d} w_i (c_{n,i} - x_{j,i})^2}, \quad w_i \geq 0, \quad i = 1, \ldots, d \ . \tag{3}$$

The constraints $w_i \geq 0$ guarantee that the distance $d_w(\boldsymbol{c}_n, \boldsymbol{x}_j)$ is real-valued and nonnegative. The output of RBF network with distance (3) is

$$\hat{y}_j(\boldsymbol{w}) = \sum_{n=1}^{N} \alpha_n K_w(\boldsymbol{c}_n, \boldsymbol{x}_j) + \alpha_0 \text{ and } K_w(\boldsymbol{c}_n, \boldsymbol{x}_j) = \exp\left(-d_w(\boldsymbol{c}_n, \boldsymbol{x}_j)^2\right) \ . \tag{4}$$

Same weights w_i are used in all the basis functions. The basis functions are ellipsoidal, whose principal axes are parallel to the coordinate axes. The basis function is located to each training data point as in (1).

The goal is to estimate the weights w_i by minimizing the MSE between the observations y_j and the outputs of network $\hat{y}_j(\boldsymbol{w})$. However, if $\boldsymbol{w} \to \boldsymbol{0}$ the output of network is constant α_0, which equals to the mean of the observations y_j. On the other hand, if $\boldsymbol{w} \to \boldsymbol{\infty}$ the output of network interpolates exactly the observations y_j. In order to achieve a smooth mapping there has to be a constraint on the values of weights w_i. This leads to consider the following optimization problem

$$\begin{aligned}
&\underset{\boldsymbol{w}}{\text{minimize}} \quad E(\boldsymbol{w}) = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j(\boldsymbol{w}))^2 + \gamma \sum_{n=1}^{N} \alpha_n^2 \\
&\text{such that} \quad \sum_{i=1}^{d} w_i \leq t \quad \text{and} \quad w_i \geq 0, \quad i = 1, \ldots, d \ .
\end{aligned} \tag{5}$$

The regularization term for the parameters α_n is also needed in this case, since the basis function is located to each data point. The constraint $\sum_{i=1}^{d} w_i < t$ shrinks the values of weights toward zero. It is known that the constraint of this type tends to set some of the coefficients w_i exactly to zero with appropriate choice of the parameter t [14,15]. This means that the corresponding input variables x_i are dropped from the model.

Problem (5) can be transformed into an unconstrained problem using the barrier function method [15]. With a logarithmic barrier function, it can be written as

$$\begin{aligned}
\underset{\boldsymbol{w}}{\text{minimize}} \quad & J(\boldsymbol{w}) = E(\boldsymbol{w}) + \mu B(\boldsymbol{w}), \quad \text{where} \quad \mu > 0 \quad \text{and} \\
& B(\boldsymbol{w}) = -\log\left(t - \sum_{i=1}^{d} w_i\right) - \frac{1}{d}\sum_{i=1}^{d}\log(w_i) \ ,
\end{aligned} \tag{6}$$

where μ is a predefined small constant. The objective function $J(\boldsymbol{w})$ is differentiable with respect to all the parameters α_0, $\alpha_n, n = 1, \ldots, N$, and $w_i, i = 1, \ldots, d$.

In this work, the unconstrained problem (6) is solved in two phases. First, the values of weights w_i are fixed and the parameters α_0 and α_n are optimized by solving the system of linear equations. Second, the obtained values of α_0 and α_n are fixed and the weights w_i are optimized. These two steps are repeated until the convergence is achieved.

The objective function $J(\boldsymbol{w})$ cannot be solved in the closed from with respect to the weights w_i. The solution is determined using Levenberg-Marquardt optimization algorithm [4,16]. The derivative of $J(\boldsymbol{w})$ with respect to w_k is

$$\nabla J(\boldsymbol{w})_k = -\frac{2}{N}\sum_{j=1}^{N} e_j \frac{\partial \hat{y}_j(\boldsymbol{w})}{\partial w_k} + \mu \frac{\partial B(\boldsymbol{w})}{\partial w_k} \ , \tag{7}$$

where $e_j = y_j - \hat{y}_j(\boldsymbol{w})$. The second partial derivative of the MSE part $E(\boldsymbol{w})$ with respect w_k and w_l is

$$\frac{\partial^2 E(\boldsymbol{w})}{\partial w_l \partial w_k} = -\frac{2}{N}\sum_{j=1}^{N} \frac{\partial e_j}{\partial w_l}\frac{\partial \hat{y}_j(\boldsymbol{w})}{\partial w_k} + e_j \frac{\partial^2 \hat{y}_j(\boldsymbol{w})}{\partial w_l \partial w_k} \ . \tag{8}$$

Algorithm 1

```
1: Set $k = 0$, $\lambda^k = 1$, $\mu = 10^{-6}$, and initialize $\boldsymbol{w}^k$
2: Evaluate the search direction $\boldsymbol{p}^k$ by solving
   $\left[\tilde{\boldsymbol{H}}(\boldsymbol{w}^k) + \lambda^k \boldsymbol{I}\right] \boldsymbol{p}^k = -\nabla J(\boldsymbol{w}^k)$
3: Determine the step length
   $\delta = \max\left\{0 \le \delta \le 1 : \sum_{i=1}^{d} w_i^k + \delta p_i^k < t, \;\; w_i^k + \delta p_i^k > 0, \;\; i = 1, \ldots, d\right\}$
4: Calculate the ratio
   $r^k = \dfrac{J(\boldsymbol{w}^k) - J(\boldsymbol{w}^k + \delta \boldsymbol{p}^k)}{J(\boldsymbol{w}^k) - \tilde{J}(\boldsymbol{w}^k + \delta \boldsymbol{p}^k)}$
5: If $r^k > 0.75$, set $\lambda^k = \lambda^k/2$
6: If $r^k < 0.25$, set $\lambda^k = 2\lambda^k$
7: If $J(\boldsymbol{w}^k + \delta \boldsymbol{p}^k) < J(\boldsymbol{w}^k)$, set $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \delta \boldsymbol{p}^k$, $\lambda^{k+1} = \lambda^k$, and $k = k + 1$
8: Go to step 2 or terminate if the stopping criterion is fulfilled
```

Following [4,16], the second term is neglected and the element (l, k) of the approximated Hessian matrix is

$$\tilde{H}(\boldsymbol{w})_{l,k} = \frac{2}{N} \sum_{j=1}^{N} \frac{\partial \hat{y}_j(\boldsymbol{w})}{\partial w_l} \frac{\partial \hat{y}_j(\boldsymbol{w})}{\partial w_k} + \mu \frac{\partial^2 B(\boldsymbol{w})}{\partial w_l \partial w_k} \; . \tag{9}$$

Only the first partial derivatives of model (4) are required in the evaluation of (7) and (9), which reduces the computational complexity.

The algorithm to optimize weights $\boldsymbol{w}$ for the given values of γ and t and the fixed values of parameters α_0 and α_n is summarized in Algorithm 1. It starts by initializing the trust region parameter λ^0 and weights w_i^0 such that the starting point is strictly feasible, i.e. $\sum w_i^0 < t$, and $w_i^0 > 0$. The algorithm continues by evaluating the search direction $\boldsymbol{p}^k$ and determining the step length δ such that the next iterate $\boldsymbol{w}^k + \delta \boldsymbol{p}^k$ stays strictly in the feasible region (steps 2 and 3). The trust region parameter λ is adjusted according to the ratio r^k, which is the ratio between the actual and the predicted decrease in the value of objective function $J(\boldsymbol{w})$. In step 7, the new iterate is accepted if it leads to reduction in the value of objective function. Steps 2-7 are repeated as long as a stopping criterion is fulfilled, which can be the maximum number of iterations or a relative change in the value of objective function.

3.1 Input Selection Algorithm for RBF Network

The two phase algorithm for solving the optimization problem is summarized in Algorithm 2. Let us assume that the regularization parameter γ and the shrinking parameter t are given. The algorithm starts by initializing the values of weights $w_i^m, m = 0$ and evaluating the kernel matrix K_{w^m}. The initialization $w_i^m = 0.9t/d$ corresponds to model (1) with the width $\sigma^2 = d/0.9t$. The next

Algorithm 2 Input selection algorithm for RBF network

```
1: Set $m = 0$, initialize the weights $w_i^m = 0.9t/d$, $i = 1, \ldots, d$, and evaluate the values of kernel matrix $K_{w^m}(\boldsymbol{x}_n, \boldsymbol{x}_j)$, $n = 1, \ldots, N$ and $j = 1, \ldots, N$
2: Use the weights $w_i^m$ to determine the values of $\alpha_0^{m+1}$ and $\alpha_n^{m+1}$, $n = 1, \ldots, N$ by minimizing
   $E(\boldsymbol{w}) = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j(\boldsymbol{w}))^2 + \gamma \sum_{n=1}^{N} \alpha_n^2$
3: Use the obtained values $\alpha_0^{m+1}$ and $\alpha_n^{m+1}$, $n = 1, \ldots, N$ to determine the weights $w_i^{m+1}$ by **Algorithm 1**
4: Update the values of kernel matrix $K_{w^{m+1}}(\boldsymbol{x}_n, \boldsymbol{x}_j)$ and evaluate the value of cost function (6) using the the parameters $\alpha_0^{m+1}$, $\alpha_n^{m+1}$, and $w_i^{m+1}$, set $m = m + 1$
5: Go to step 2 or terminate if the stopping criterion is fulfilled
```

step is to estimate the parameters α_0^{m+1} and α_n^{m+1} by minimizing the regularized error function $E(\boldsymbol{w})$ for the fixed values of weights w_i^m. New values for the weights w_i^{m+1} are computed using Algorithm 1 using in the previous step obtained values of α_0^{m+1} and α_n^{m+1} (step 3). Algorithm 1 is terminated if the relative decrease in the value of objective function during the last five iterations is less than 10^{-4}. The maximum number of iterations is 10 in Algorithm 1. Algorithm 2 continues by updating the kernel matrix $K_{w^{m+1}}$ and evaluating the value of objective function (6) using the new values of parameters α_0^{m+1}, α_n^{m+1} and the weights w_i^{m+1} (step 4). Steps 2-4 are repeated until the stopping criterion is achieved. The iteration is terminated if the relative decrement is less than 10^{-4} during the last five iterations or 200 iterations are performed. Algorithm 2 produces the sequence of decreasing values for the objective function.

4 Experiments

4.1 Simulated Data

In this experiment, the performance of Algorithm 2 is illustrated using a simulated data set. In the case of simulated data, the assessment of quality of results is straightforward, since the underlying phenomenon is completely known. The values of each of five input variables $\boldsymbol{x} = [x_1, \ldots, x_5]$ were independently drawn from the uniform distribution in the range $x_i \in [-3, 3]$. The target was one dimensional sinc function and noisy samples from that model were generated as

$$y_j = \mathrm{sinc}(x_{j,1}) + \varepsilon_j, \quad j = 1, \ldots, N \ , \tag{10}$$

where ε_j were independent samples from the normal distribution $\varepsilon_j \sim \mathrm{N}(0, 0.15^2)$. Thus, only first input was relevant. The sizes of the training and the validation sets were $N_t = 200$ and $N_v = 2000$, respectively.

Algorithm 2 is evaluated in twenty points of the regularization parameter γ and the shrinking parameter t, which were logarithmically equally spaced in the

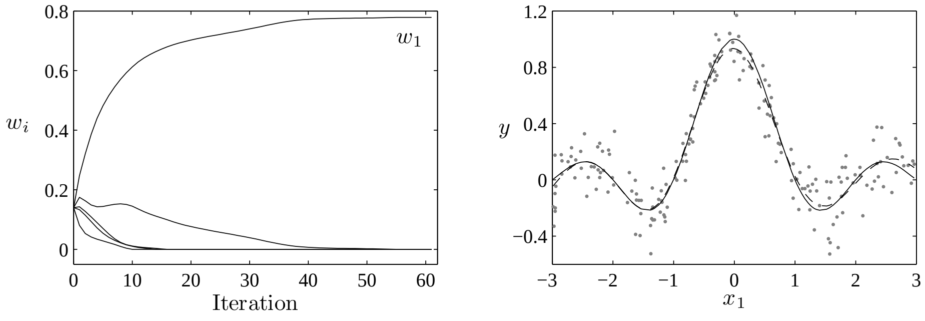

Fig. 1. Development of weights w_i as a function of iterations in Algorithm 2 for the simulated data (*left panel*) using the values of parameters γ and t, which produce the minimum validation error. On the *right panel*, the training samples $\boldsymbol{x}_1$ (*gray dots*), the unnoisy sinc function (*solid line*), and the estimated sinc function (*dashed line*).

ranges $\gamma \in [10^{-4}, 1]$ and $t \in [0.3, 4]$, respectively. In all, 400 RBF networks were trained using Algorithm 2. The validation error was the MSE for the validation set and the minimum was achieved using the parameter values $\gamma = 0.013$ and $t = 0.779$. The standard deviation of the validation errors $e_j = y_j - \hat{y}_j(\boldsymbol{w})$ was 0.133, which corresponds very well to the standard deviation of the noise.

On the left panel of Fig. 1, the evolution of the weights $w_i,\ i = 1, \ldots, 5$ during the training are presented as a function of the iterations m in Algorithm 2. The stopping criterion was fulfilled after 62 iterations. In the end, only the weight w_1 was nonzero, which corresponds to the input variable x_1. This means that Algorithm 2 detected the relevant input and discarded irrelevant ones from the model. The right panel of Fig. 1 presents the samples of the first dimension of input variables (gray dots), the target sinc function (solid line), and estimated function (dashed line). The peak of the sinc function is slightly underestimated and there is also visible overestimation with large values of x_1. Otherwise the approximation is nearly perfect.

4.2 Boston Housing Data

The purpose of this experiment is to illustrate the performance of Algorithm 2 and compare it to the ordinary RBF networks defined by (1) using a real data set. The used data are called Boston Housing data set and it can be downloaded from the UCI Machine Learning Repository[1]. The data contain 506 samples from $d = 13$ input variables $x_i,\ i = 1, \ldots, 13$ and from a single output y. The data set was randomly divided to the training set ($N_t = 400$) and to the test set ($N_{test} = 106$). All the inputs and the output were scaled to have zero mean and unit variance to make the weights comparable.

[1] http://www.ics.uci.edu/∼mlearn/MLRepository.html

Table 1. MSEs for Boston Housing data. Minimum CV error (2. column), the test error of the minimum CV error model (3.column), and the optimal test error (4. colum).

Model	CV error	Test error	Opt. test error
Ordinary RBF	0.127	0.143	0.126
Sparse RBF	0.130	0.149	0.112

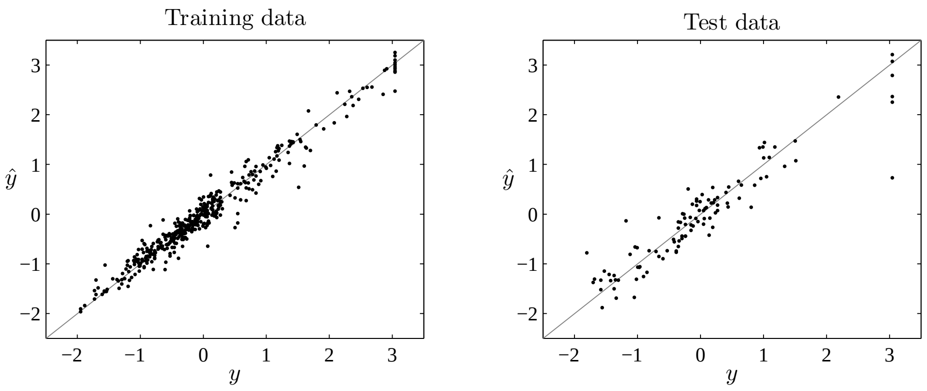

Fig. 2. The values of estimated output $\hat{y}_j(\boldsymbol{w})$ plotted versus the actual values y_j for the training data (*left*) and the test data (*right*).

The training of RBF networks was carried out using a 10-fold cross validation (CV). The ordinary RBF was evaluated using 20 values of the regularization parameter γ_o and the widths of Gaussian basis function σ^2, which were logarithmically equally spaced in the ranges $\gamma_o \in [10^{-9}, 5 \cdot 10^{-5}]$ and $\sigma^2 \in [4, 500]$. Algorithm 2 was calculated using 20 logarithmically equally spaced points of the regularization parameter γ_a and the shrinking parameter t ($\gamma_a \in [2 \cdot 10^{-6}, 0.03]$ and $t \in [0.2, 7]$).

The minimum CV errors and the corresponding test errors for the ordinary RBF network and the sparse RBF network, i.e. Algorithm 2, are shown in the second and in the third column of Table 1. The test errors were also calculated for all the combinations of parameters and the results are shown in the last column of Table 1. In practice, the both methods are equally accurate in the prediction. However, it is notable that the most accurate CV model did not produce the best possible test error in either model.

In Fig. 2, the estimated outputs versus the actual outputs are shown for the minimum CV error model for the training data (*left*) and the test data (*right*). The predictions are overall very good in the training data. In the test set, some of the highest values of the output are evidently underestimated otherwise the approximation is excellent.

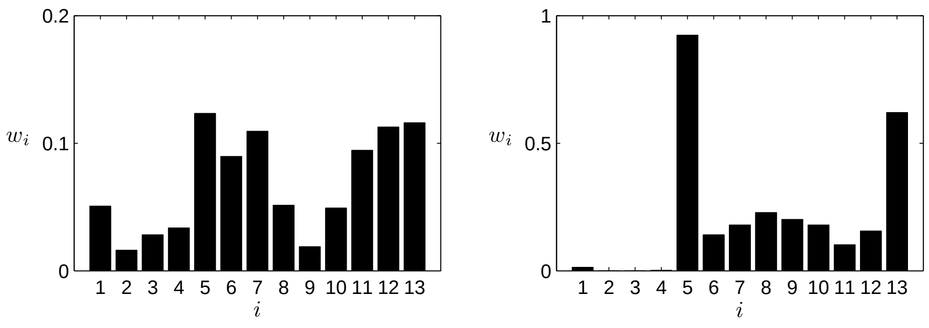

Fig. 3. The values of weights w_i in the minimum CV error model (*left*) and in the model producing the optimal test error (*right*).

On the left panel of Fig. 3, the weights w_i of the minimum CV error model are shown. All the weights are non-zero, but the values are not equal. Thus, the model highlights differences in importances of the inputs in the prediction. In [9] proposed method selects also all the input variables in the case of Boston Housing data. On the right panel of Fig. 3, the weights w_i of the optimal test error model are illustrated. The inputs x_2 and x_3 are rejected from this model. Also, the weights of the first and fourth inputs w_1 and w_4 are nearly zero. The validation error of the optimal test error model (0.146) was just slightly worse than the minimum CV error (0.130). Thus, it would probably be reasonable to make compromise between the CV error and number of selected inputs in the model selection as it is done in [8].

5 Summary and Conclusions

In this work, an input selection algorithm for the RBF networks was proposed. The algorithm solves a constrained optimization problem. The weighted Euclidean distance is used in the basis functions and the constraint structure on weights favors sparsity in terms of the input variables. The proposed algorithm solves the weights and the parameters of the output layer in two phases. Sparsity in terms of the inputs makes the models more interpretable compared to the ordinary RBF networks. The method was applied to the simulated and real world data set and the results were convincing in both cases.

In the case of large training data, it is not feasible to place the basis function in each training data point. The centers of basis functions can then be selected using some unsupervised technique. After that, the centers are kept fixed and the proposed algorithm can be applied without any modifications. However, the regularization parameter γ is not necessarily needed anymore. That would decrease computational complexity in the model selection phase, since only the value of shrinking parameter t would have to be validated. The disadvantage of unsupervised approach is that the selection of centers are based only on the input data. The resulting centers may be suboptimal solution with respect to the prediction accuracy. Probably better results would be achieved if the centers of the basis functions were optimized using the embedded approach. That is, the selection of centers of basis functions would be incorporated into the training process. Further work is also required that the weights and the parameters of output layer would be optimized simultaneously.

References

1. Park, J., Sandberg, I.W.: Approximation and Radial-Basis-Function Networks. Neural Computation **5** (1993) 305–316
2. Orr, M.J.L.: Regularization in the Selection of Radial Basis Function Centers. Neural Computation **7** (1995) 606–623
3. Moody, J., Darken, C.J.: Fast Learning in Networks of Locally-Tuned Processing Units. Neural Computation **1** (1989) 218–294
4. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
5. Benoudjit, N., Verleysen, M.: On the Kernel Widths in Radial-Basis Function Networks. Neural Processing Letters **18** (2003) 139–154
6. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Reasearh **3** (2003) 1157–1182
7. Herrera, L.J., Pomares, H., Rojas, I., Verleysen, M., Guilén, A.: Effective Input Variable Selection for Function Approximation. In Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 2006). LNCS **4131** (2006) 41–50
8. Tikka, J., Lendasse, A., Hollmén, J.: Analysis of Fast Input Selection: Application in Time Series Prediction. In Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 2006). LNCS **4132** (2006) 161–170
9. Bi, J., Bennett, K.P., Embrechts, M., Breneman, C.M., Song, M.: Dimensionality Reduction via Sparse Support Vector Machines. Journal of Machine Learning Research **3** (2003) 1229–1243
10. Kohavi, R., John, G.H.: Wrappers for Feature Selection. Artificial Intelligence **97** (1997) 273–324
11. Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., Vapnik, V.: Feature Selection for SVMs. In Advances in Neural Information Processing Systems **13** (2000)
12. Rakotomamonjy, A.: Variable Selection Using SVM-based Criteria. Journal of Machine Learning Research **3** (2003) 1357–1370
13. Valls, J.M., Aler, R., Fernández, O.: Using a Mahalanobis-Like Distance to Train Radial Basis Neural Networks. In Proceedings of the 8th International Work-Conference on Artificial Neural Networks (IWANN 2005). LNCS **3512** (2005) 257–263
14. Tibshirani, R.: Regression Shrinkage and Selection via the LASSO. Journal of the Royal Statistical Society, Series B (Methodological) **58** (1996) 267–288
15. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programming Theory and Algorithms. John Wiley & Sons, Inc. (1979)
16. Nørgaard, M., Ravn, O., Poulsen, N.K., Hansen, L.K.: Neural Networks for Modelling and Control of Dynamic Systems. Springer-Verlag (2003)