TWRBF – Transductive RBF Neural Network with Weighted Data Normalization

Qun Song and Nikola Kasabov

Knowledge Engineering & Discovery Research Institute
Auckland University of Technology
Private Bag 92006, Auckland 1020, New Zealand
{qsong,nkasabov}@aut.ac.nz

Abstract. This paper introduces a novel RBF model – *Transductive Radial Basis Function Neural Network* with *Weighted Data Normalization* (TWRBF). In transductive systems a local model is developed for every new input vector, based on some closest to this vector data from the training data set. The *Weighted Data Normalization* method (WDN) optimizes the data normalization range individually for each input variable of the system. A gradient descent algorithm is used for training the TWRBF model. The TWRBF is illustrated on two case study prediction/identification problems. The first one is a prediction problem of the Mackey-Glass time series and the second one is a real medical decision support problem of estimating the level of renal functions in patients. The proposed TWRBF method not only gives a good accuracy for an individual, "personalized" model, but depicts the most significant variables for this model.

1 Introduction: Transductive Modeling and Weighted Data Normalization

Most of learning models and systems in artificial intelligence, developed and implemented so far, are based on *inductive* methods, where a model (a function) is derived from data representing the problem space and this model is further applied on new data. The model is usually created without taking into account any information about a particular new data vector (test data). An error is measured to estimate how well the new data fits into the model. The inductive learning and inference approach is useful when a global model of the problem is needed even in its very approximate form. In contrast to the inductive learning and inference methods, *transductive* methods estimate the value of a potential model (function) only in a single point of the space (the new data vector) utilizing additional information related to this point. This approach seems to be more appropriate for clinical and medical applications of learning systems, where the focus is not on the model, but on the individual patient. Each individual data vector may need an individual, local model that best fits the new data, rather then a global model, in which the new data is matched without taking into account any specific information about this data.

A transductive method is concerned with the estimation of a function in single point of the space only [13]. For every new input vector $\boldsymbol{x}_i$ that needs to be processed for a prognostic/classification task, the N_i nearest neighbours, which form a data subset D_i, are derived from an existing data set D or/and generated from an existing

N.R. Pal et al. (Eds.): ICONIP 2004, LNCS 3316, pp. 633–640, 2004.

model M. A new model M_i is dynamically created from these samples to approximate the function in the point $\boldsymbol{x}_i$ – Fig. 1 and Fig. 2. The system is then used to calculate the output value $\boldsymbol{y}_i$ for this input vector $\boldsymbol{x}_i$.

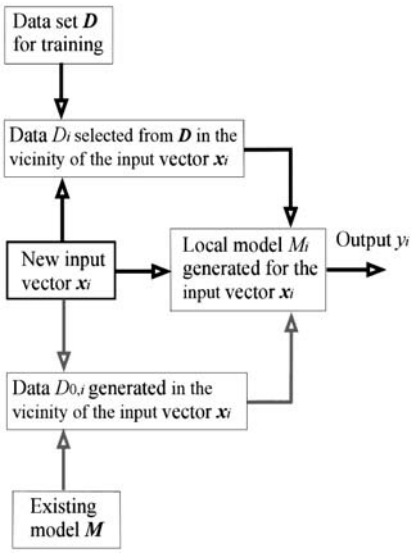
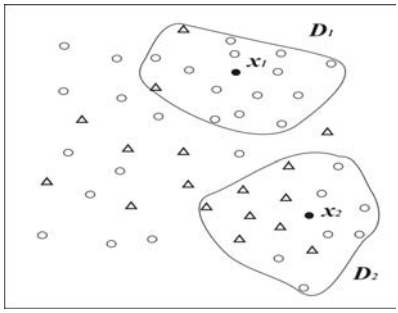


Fig. 1. A block diagram of a *transductive* model. An individual model M_i is trained for every new input vector $\boldsymbol{x}_i$ with data use of samples D_i selected from a data set D, and data samples $D_{0,i}$ generated from an existing model (formula) M (if such a model is existing). Data samples in both D_i and $D_{0,i}$ are similar to the new vector $\boldsymbol{x}_i$ according to defined similarity criteria



● – a new data vector; ○ – a sample from D; Δ – a sample from M

Fig. 2. The new data vector is in the centre of a transductive model (here illustrated with two of them – $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$), and is surrounded by a fixed number of nearest data samples selected from the training data D and possibly generated from an existing model M

In many neural network models and applications, raw (not normalized) data is used. This is appropriate when all the input variables are measured in the same units. Normalization, or standardization, is reasonable when the variables are in different

units, or when the variance between them is substantial. However, a general normalization means that every variable is normalized in the same range, e.g. [0, 1] with the assumption that they all have the same importance for the output of the system.

For many practical problems, variables have different importance and make different contribution to the output. Therefore, it is necessary to find an optimal normalization and assign proper importance factors to the variables. Such a method can also be used for feature selection, or for reducing the size of the input vectors through keeping the most important ones [12]. This is especially applicable to a special class of models – the clustering based neural networks or fuzzy systems (or also: distance-based; prototype-based) such as RBF [6, 10] and ECOS [3, 4]. In such systems, distance between neurons or fuzzy rule nodes and input vectors are usually measured in *Euclidean distance*, so that variables with a higher upper bound of the range will have more influence on the learning process and on the output value, and vice versa.

The paper is organized as follows: Sect. 2 presents the structure and algorithm of the TWRBF model. Sect. 3 and 4 illustrate the approach on two case study problems. Conclusions are drawn in Sect. 5.

2 Transductive RBF Neural Networks with Weighted Data Normalization: Structure and Learning Algorithm

The origin of the radial basis function (RBF) models is in the approximation theory and in the regularization theory [10]. They deal with the general approximation problem common to supervised learning neural networks. Standard RBF networks have three layers of neurons: input layer - that represents input variables; hidden layer – representing centers of clusters of data in the input space; and output layer – that represents the output variables. Although RBF networks usually need more neurons in their hidden layer than standard feed-forward back-propagation networks, such as MLP, they train much faster than MLP networks and can be used to extract meaningful rules based on clusters of data. RBF models work very well when there is enough training data.

A TWRBF network has an additional layer – the WDN layer for weighted data normalization. *Gaussian* kernel functions are used as activation functions for each neuron in the hidden layer. The TWRBF structure is shown in Fig. 3.

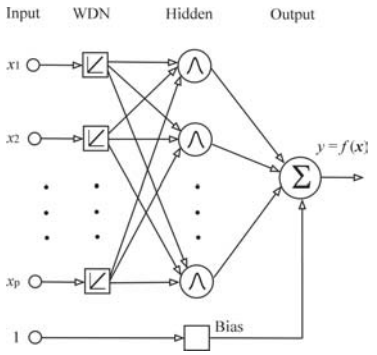


Fig. 3. The Structure of the proposed TWRBF Network

For each new data vector $\boldsymbol{x}_q$, the TWRBF learning algorithm performs the following steps:

1. Normalize the training data set (the values are between 0 and 1) with the initial input variable weights.
2. Search in the training data set in the input space with weighted normalized *Euclidean* distance, defined as Eq.1, to find N_q training examples that are closest to $\boldsymbol{x}_q$. The value for N_q can be pre-defined based on experience, or – optimized through the application of an optimization procedure. Here we assume the former approach.

$$\|\boldsymbol{x} - \boldsymbol{y}\| = \frac{1}{P}\left[\sum_{j=1}^{P} w_j^2 |x_j - y_j|^2\right]^{\frac{1}{2}} \tag{1}$$

   where: $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{R}^P$ and w_j are weights.
3. Calculate the distances d_i, $i = 1, 2, \ldots, N_q$, between each of these data samples and $\boldsymbol{x}_q$. Calculate the vector weights $v_i = 1 - (d_i - \min(\boldsymbol{d}))$, $i = 1, 2, \ldots, N_q$, $\min(\boldsymbol{d})$ is the minimum value in the distance vector $\boldsymbol{d} = [d_1, d_2, \ldots, d_{Nq}]$.
4. Use the *ECM* clustering algorithm [5, 11] (or some other clustering methods) to cluster and partition the input sub-space that consists of N_q selected training samples.
5. Create *Gaussian* kernel functions and set their initial parameter values according to the *ECM* clustering procedure results; for each cluster, the cluster centre is taken as the centre of a *Gaussian* function and the cluster radius is taken as the width.
6. Apply the gradient descent method to optimize the parameters of the system in the local model M_q that include the individual variable normalization weights (the upper bound of the normalization intervals, the lower bound being 0) (see Eq. 2-9 below).
7. Search in the training data set to find N_q samples (the same to Step 2), if the same samples are found as the last search the algorithm turns to Step 8, otherwise, Step 3.
8. Calculate the output value y_q for the input vector x_q applying the trained model.
9. End of the procedure.

The parameter optimisation procedure is described below:

Consider the system having P inputs, one output, and M neurons in the hidden layer, the output value of the system can be calculated on input vector $\boldsymbol{x}_i = [x_1, x_2, \ldots, x_P]$ as follows:

$$y = f(\boldsymbol{x}_i) = b_0 + b_1 R_1(\boldsymbol{x}_i) + b_2 R_2(\boldsymbol{x}_i), \ldots, + b_M R_M(\boldsymbol{x}_i) \tag{2}$$

$$\text{here, } R_l(\boldsymbol{x}_i) = \prod_{j=1}^{P} \exp\left[-\frac{w_j^2 (x_{ij} - m_{lj})^2}{2\sigma_{lj}^2}\right] \tag{3}$$

m_{lj} and σ_{lj} are parameters of *Gaussian* functions, w_j are weights of input variables.

In the TWRBF learning algorithm, the following indexes are used:

- Training Data pairs: $i = 1, 2, \ldots, N$;
- Input Variables: $j = 1, 2, \ldots, P$;
- Neurons in the hidden layer: $l = 1, 2, \ldots, M$;
- Learning Iterations: $k = 1, 2, \ldots$

The TWRBF learning algorithm minimizes the following objective function (an error function):

$$E = \frac{1}{2}\sum_{i=1}^{N} v_i[f(\boldsymbol{x}_i) - t_i]^2 \qquad (v_i \text{ are defined in Step 3}) \tag{4}$$

The gradient descent algorithm (back-propagation algorithm) is used on the training data pairs, $[\boldsymbol{x}_i, t_i]$, to obtain the recursions for updating the parameters $\boldsymbol{b}$, $\boldsymbol{m}$, $\boldsymbol{\sigma}$ and $\boldsymbol{w}$ such that E of Eq. 4 is minimized:

$$b_0(k+1) = b_0(k) - \eta_b \sum_{i=1}^{N} v_i[f(\boldsymbol{x}_i) - t_i] \tag{5}$$

$$b_l(k+1) = b_l(k) - \eta_b \sum_{i=1}^{N} \{v_i R_l(\boldsymbol{x}_i)[f(\boldsymbol{x}_i) - t_i]\} \tag{6}$$

$$m_{lj}(k+1) = m_{lj}(k) - \eta_m \sum_{i=1}^{N} \left\{ v_i R_l(\boldsymbol{x}_i) b_l [f(\boldsymbol{x}_i) - t_i] \frac{w_j^2 (x_{ij} - m_{lj})}{\sigma_{lj}{}^2} \right\} \tag{7}$$

$$\sigma_{lj}(k+1) = \sigma_{lj}(k) - \eta_\sigma \sum_{i=1}^{N} \left\{ v_i R_l(\boldsymbol{x}_i) b_l [f(\boldsymbol{x}_i) - t_i] \frac{w_j^2 (x_{ij} - m_{lj})^2}{\sigma_{lj}^3} \right\} \tag{8}$$

$$w_j(k+1) = w_j(k) - \eta_w \sum_{l=1}^{M} \sum_{i=1}^{N} \left\{ v_i R_l(\boldsymbol{x}_i) b_l [f(\boldsymbol{x}_i) - t_i] \frac{w_j (m_{lj} - x_{ij})^2}{\sigma_{lj}{}^2} \right\} \tag{9}$$

here, η_b, η_m, η_σ and η_w are learning rates for updating the parameters $\boldsymbol{b}$, $\boldsymbol{m}$, $\boldsymbol{\sigma}$ and $\boldsymbol{w}$ respectively.

3 Case Study Example of Applying the TWRBF for a Time-Series Prediction

In this paper, the TWRBF network is applied to a time series prediction. The TWRBF learning is demonstrated on the Mackey-Glass (MG) time series prediction task [8]. The MG time series is generated with a time-delay differential equation as follows:

$$\frac{dx(t)}{dt} = \frac{0.2\, x(t-\tau)}{1 + x^{10}(t-\tau)} \tag{10}$$

To obtain this time series values at integer points, the *fourth-order Runge-Kutta method* was used to find the numerical solution to the above MG equation. Here we used the following parameter values: a time step of 0.1, $x(0) = 1.2$, $\tau = 17$ and $x(t) = 0$ for $t < 0$. From the input vector $[x(t-18)\ \ x(t-12)\ \ x(t-6)\ \ x(t)]$, the task is to predict the value $x(t+85)$. In the experiments, 3000 data points, from t = 201 to 3200, were extracted as training data; 500 data points, from t = 5001 to 5500 were taken as testing data. For each of these testing data sample a local transductive model is created and tested on this data.

Table 1. Experimental results on MG data

Model	Neurons or rules	Training iterations	Testing NDEI	WDN - Input variable w1	w2	w3	w4
MLP	60	500	0.022	1	1	1	1
ANFIS	81	500	0.025	1	1	1	1
DENFIS	58	100	0.017	1	1	1	1
RBF	128	200	0.031	1	1	1	1
TWRBF	6.2 (**average**)	200	0.012	0.95	1	0.74	0.89

To compare the performance of the TWRBF method we conducted the following experiments. We applied some well-known inductive models, such as MLP [9], RBF [9], ANFIS [1, 2] and DENFIS [5], on the same data. The experimental results are listed in Table 1 that includes the number of neurons in the hidden layer (for MLP and RBF) or fuzzy rules (for ANFIS and DENFIS), training iterations and testing NDEI – *non-dimensional error index* which is defined as the *root mean square error* (RMSE) divided by the standard deviation of the target series, the WDN upper bound for each input variable. It is seen that variable x2 is at average the most significant when this method is applied.

The proposed method, TWRBF, evolves individual, "personalized" models and performs better at average than the other inductive, global models. This is a result of the fine tuning of each local, individual model in TWRBF for each simulated example. The finely tuned local models achieve a better local generalization.

4 Case Study Example of Applying the TWRBF for a Medical Decision Support Problem

A real data set from a medical institution is used here for experimental analysis. The data set has 447 samples, collected at hospitals in New Zealand and Australia. Each of the records includes six variables (inputs): age, gender, serum creatinine, serum albumin, race and blood urea nitrogen concentrations, and one output - the Glomerular Filtration Rate (GFR) value. All experimental results reported here are based on 10-cross validation experiments with the same model and parameters and the results are averaged. In each experiment 70% of the whole data set is randomly selected as training data and another 30% as testing data.

For comparison, several well-known methods are applied on the same problem, such as the MDRD function [7], MLP and RBF neural network [9], adaptive neural fuzzy inference system (ANFIS) [1] and dynamic evolving neural fuzzy inference system (DENFIS) [5] along with the proposed TWRBF and results are given in Table 2. The results include the number of fuzzy rules (for ANFIS and DENFIS), or neurons in the hidden layer (for RBF and MLP), the testing RMSE (root mean square error), the testing MAE (mean absolute error) and the average weights of the input variables representing their average significance for the GFR prediction. It is seen that Sreatenine and Age are the most important variables at average.

Table 2. Experimental results on GFR data

Model	Neurons or rules	Testing RMSE	Testing MAE	Weights of input variables					
				Age w1	Sex w2	Scr w3	Surea w4	Race w5	Salb w6
MDRD	—	7.74	5.88	1	1	1	1	1	1
MLP	12	8.44	5.75	1	1	1	1	1	1
ANFIS	36	7.49	5.48	1	1	1	1	1	1
DENFIS	27	7.29	5.29	1	1	1	1	1	1
RBF	32	7.22	5.41	1	1	1	1	1	1
TWRBF	5.1 (average)	7.10	5.15	0.92	0.63	1	0.78	0.37	0.45

The proposed TWRBF method not only gives a good accuracy at average, but provides a good accuracy for an individual, "personalized" model, and also depicts the most significant variables for this model as it is illustrated on fig.3 on a single, randomly selected sample from the GFR data. The importance of variables for a particular patient may indicate potential specific problems and a personalized treatment.

Table 3. A TWRBF personalised model of a single patient (one sample from the GFR data)

Values for the Input variables of a selected patient	Age 58.9	Sex Female	Scr 0.28	Surea 28.4	Race White	Salb 38
Weights of input variables (TWRBF) for this patient	0.84	0.62	1	0.83	0.21	0.41
Results	GFR (desired) 18.0		MDRD 14.9		TWRBF 16.2	

5 Conclusions

This paper presents a transductive RBF neural network with weighted data normalization method – TWRBF. The TWRBF results in a better local generalization over new data as it develops an individual model for each data vector that takes into account the new input vector location in the space, and it is an adaptive model, in the sense that input-output pairs of data can be added to the data set continuously and immediately made available for creating transductive models. This type of modelling can be called "personalised", and it is promising for medical decision support systems. As the TWRBF creates a unique sub-model for each data sample, it usually needs more performing time than inductive models, especially in the case of training and simulating on large data sets.

Further directions for research include: (1) TWRBF system parameter optimization such as optimal number of nearest neighbors; and (2) applications of the TWRBF method for other decision support systems, such as: cardio-vascular risk prognosis; biological processes modeling and prediction based on gene expression micro-array data.

Acknowledgement

The research presented in the paper is funded by the New Zealand Foundation for Research, Science and Technology under grant NERF/AUTX02-01. We thank Dr

Mark Marshall and Ms Maggie Ma from the Middlemore hospital in Auckland for their help with the clinical data.

References

1. Fuzzy System Toolbox, MATLAB Inc., Ver.2 (2002)
2. Jang, R., "ANFIS: adaptive network-based fuzzy inference system", *IEEE Trans. on Syst.,Man, and Cybernetics*, vol. 23, No.3, (1993) 665 – 685 1993.
3. Kasabov, N., "Evolving fuzzy neural networks for on-line supervised/unsupervised, knowledge–based learning," *IEEE Trans. SMC* – part B, Cybernetics, vol.31, No.6 (2001) 902-918
4. Kasabov, N., Evolving connectionist systems: Methods and Applications in Bioinformatics, Brain study and intelligent machines, Springer Verlag, London, New York, Heidelberg, (2002)
5. Kasabov, N. and Song, Q., "DENFIS: Dynamic, evolving neural-fuzzy inference systems and its application for time-series prediction," *IEEE Trans. on Fuzzy Systems*, vol. 10, (2002) 144 – 154
6. Lee, Y., "Handwritten digit recognition using K nearest-neighbor, radial basis function, and back-propagation neural networks," *Neural Computation,* vol.3 No.3 (1991) 440 – 449
7. Levey, A. S., Bosch, J. P., Lewis, J. B., Greene, T., Rogers, N., Roth, D., for the Modification of Diet in Renal Disease Study Group, " A More Accurate Method To Estimate Glomerular Filtration Rate from Serum Creatinine: A New Prediction Equation", *Annals of Internal Medicine*, vol. 130 (1999) 461 – 470
8. Mackey, M. C., and Glass, L., "Oscillation and Chaos in Physiological Control systems", *Science*, vol. 197 (1977) 287 – 289
9. Neural Network Toolbox user's guide, The MATH WORKS Inc., Ver.4 (2002)
10. Poggio, F., "Regularization theory, radial basis functions and networks" In: *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*. NATO ASI Series, No.136 (1994) 83 – 104
11. Song, Q. and Kasabov, N., "ECM - A Novel On-line, Evolving Clustering Method and Its Applications", *Proceedings of the Fifth Biannual Conference on Artificial Neural Networks and Expert Systems (ANNES2001)*, (2001) 87 – 92
12. Song, Q. and Kasabov, N., "Weighted Data Normalizations and feature Selection for Evolving Connectionist Systems Proceedings", *Proc. of The Eighth Australian and New Zealand Intelligence Information Systems Conference* (ANZIIS2003), (2003) 285 – 290
13. V. Vapnik, Statistical Learning Theory, John Wiley & Sons, Inc. (1998)