ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

Function Approximation by Localized Basis Function Neural Networks

M. Kokol, I. Grabec

Faculty of Mechanical Engineering
Aškerčeva 6, Ljubljana, Slovenia

Abstract. A transition from the localized basis function description of a probability density function to the general regression estimator and corresponding neural network model is presented in this article, and its relations to localized basis function neural network are explained. Typical parametric and nonparametric models are described more in detail. Among them, a new elliptical multivariate basis function approach is the most advanced one. In the article, its performance is compared with the radial basis function neural network using various two-dimensional examples.

1. Introduction

One of the most important applications of neural networks is approximation of functions. Systems dealing with this problem are called *the mapping neural networks*. The most well known are multilayer perceptrons learning by back-propagation, counter-propagation network, and localized basis function neural network (LBFNN) [1].

The aim of this article is to explain theoretically that LBFNN-s have simple statistical background from which various recall rules can be obtained by standard procedures as follows.

Let us dente an independent variable by $\boldsymbol{x} \in \mathcal{X} \subset \Re^m$ and a dependent one by $\boldsymbol{y} \in \mathcal{Y} \subset \Re^n$. The regression of a variable $\boldsymbol{y}$ on a given variable $\boldsymbol{x}$ is the estimator

$$\widehat{\boldsymbol{y}}(\boldsymbol{x}) = E[\boldsymbol{y}|\boldsymbol{x}] \tag{1}$$

where the symbol $E[\,|\,]$ represents the conditional average operator. Let us suppose that the joint probability density function $p(\boldsymbol{x}, \boldsymbol{y})$ over the measurement space is known. The regression estimator (1) is then expressed as

$$\begin{aligned} \widehat{\boldsymbol{y}}(\boldsymbol{x}) = E[\boldsymbol{y}|\boldsymbol{x}] &= \int \boldsymbol{y}\, p(\boldsymbol{y}|\boldsymbol{x})\, d^n y \\ &= \int \boldsymbol{y}\, \frac{p(\boldsymbol{x}, \boldsymbol{y})}{\int p(\boldsymbol{x}, \boldsymbol{y})\, d^n y}\, d^n y. \end{aligned} \tag{2}$$

ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

The main problem is to estimate the joint probability density function by which the regression estimator of dependent variable $\boldsymbol{x}$ over the independent one $\boldsymbol{y}$ is determined.

2. LBFNN as regression estimator

A fundamental step in the development of LBFNN is the concatenation of measured variables $\boldsymbol{x}$ and $\boldsymbol{y}$

$$\boldsymbol{z} = \boldsymbol{x} \oplus \boldsymbol{y} = \left\| \begin{matrix} \boldsymbol{x} \\ \boldsymbol{y} \end{matrix} \right\|$$

yielding the joint sample space $\mathcal{Z} = \mathcal{X} \oplus \mathcal{Y}$. Elements of that space are vectors $\boldsymbol{z}$ having $(n + m)$ dimensions. In further treatment the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are considered as independent and dependent variables, respectively.

The joint probability density function $p(\boldsymbol{z}) = p(\boldsymbol{x}, \boldsymbol{y})$ is usually not known in practice and must be estimated from measurements. Its estimation is the main task of learning the LBFNN.

By the repetition of measurement we obtain a set of pairs $\{\boldsymbol{z}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i); i = 1, 2, ...N\}$ which represent independent samples from the sample space $\mathcal{Z}$ with the under-laying probability density function $p(\boldsymbol{z})$. Among various methods for estimation the probability density function from finite sample set we select the following description:

$$p(\boldsymbol{z}) = \sum_i f_i(\text{center,width}) \tag{3}$$

in which $f_i()$ denotes the localized functions. Their characteristics are usually described by a center and a width, while their analytical expression should be chosen in accordance with the properties of the joint measurement space $\mathcal{Z}$. A very broad class of distributions can be well represented by multivariate Gaussian basis functions. An essential advantage of LBFNN over other paradigms stems from the fact that after the measurement of joint data the separation into the dependent and independent variable can be selected arbitrarily.

In the following sub-sections, we present several examples in which we apply some of the most frequently-used functions $f_i()$. The corresponding regression estimator (2) is calculated and two typical examples of nonlinear functions are presented.

2.1. Non-parametric LBFNN

The probability density function estimate (3) is called non-parametric, when each measurement sample $\boldsymbol{z}_i$ is a center vector for localized function $f_i()$. Therefore, there are as many "bumps" in the estimate (3) as given measured samples.

ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

2.1.1. Non-parametrical radial case

A radial non-parametric probability density function estimator stems from the Parzen window approach [2]. Using radial multivariate Gaussian windows it is expressed as [1]

$$p(\boldsymbol{z},\sigma) = \frac{C}{N}\sum_{i=1}^{N}\frac{1}{\sigma^{(n+m)}}\exp\left(-\frac{(\boldsymbol{z}-\boldsymbol{z}_i)^T(\boldsymbol{z}-\boldsymbol{z}_i)}{2\,\sigma^2}\right). \tag{4}$$

Inserting $p(\boldsymbol{z})$ into Eq.(2), we obtain a recall rule for the nonparametric RBFNN

$$\widehat{\boldsymbol{y}}(\boldsymbol{x}) = \sum_{i=1}^{n}\boldsymbol{y}_i\frac{\exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}_i)^T(\boldsymbol{x}-\boldsymbol{x}_i)}{2\,\sigma^2}\right)}{\sum_{j=1}^{n}\exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}_j)^T(\boldsymbol{x}-\boldsymbol{x}_j)}{2\,\sigma^2}\right)}$$

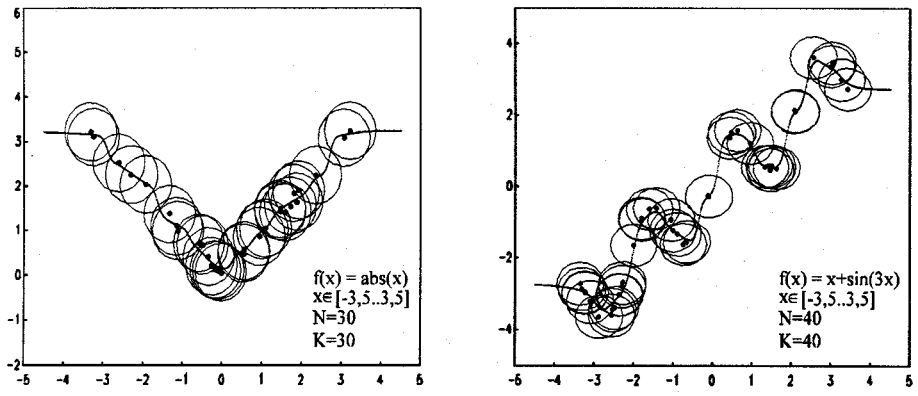

Figure 1: Approximation with the non-parametric RBFNN.

It has been already used in many applications [3, 4].

The non-parametric RBFNN has an essential advantage regarding parametric networks. It does not need to be adapted, sufficient is only the measurement of data and specification of width σ. By the measurement samples the centers of the localized basis functions are determined, while the width σ provides for a smooth representation of continuous probability density function by finite number of sample data. It can be selected globally for all basis functions. However, we can also mention deficiencies of the LBFNN: there have to be stored many prototype samples, the estimation variability is high, and extrapolation is impossible.

2.2. Parametric LBFNN

In order to reduce memory requirement of the non-parametric network a parameterization is introduced. In this case a small number $K \ll N$ of prop-

[1] Constant C in all examples is $C = 1/(2\pi)^{(n+m)/2}$.

ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

erly selected localized basis functions can be distributed over the measurement space $\mathcal{Z}$ so that they judiciously describe the under-laying probability density function. A localized basis function $f_j()$ is then described by adaptable center vector $\boldsymbol{Q}_j$ and width σ_j in the radial case or by a width matrix $\boldsymbol{\Sigma}_j$ in the elliptical case. The centers can be positioned by an appropriate clustering procedure, but an efficient adaptation algorithm for the widths is still an open problem.

2.2.1. Parametrical radial case

Let the probability density function estimate (3) be represented in the radial form

$$p(\boldsymbol{z}, \boldsymbol{Q}, \sigma) = \frac{C}{K} \sum_{j=1}^{K} \frac{1}{\sigma_j^{(n+m)}} \exp\left(-\frac{(\boldsymbol{z}-\boldsymbol{Q}_j)^T(\boldsymbol{z}-\boldsymbol{Q}_j)}{2\sigma_j^2}\right). \tag{5}$$

Inserting this estimate into (2), the parametric RBFNN recall rule is obtained. With this aim, we have to split center vectors $\boldsymbol{Q}_j$ in the independent and dependent part

$$\boldsymbol{Q}_j = \begin{Vmatrix} \boldsymbol{Q}_{(x)j} \\ \boldsymbol{Q}_{(y)j} \end{Vmatrix}$$

It follows then

$$\widehat{\boldsymbol{y}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{Q}_{(y)k} \frac{\exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{Q}_{(x)k})^T(\boldsymbol{x}-\boldsymbol{Q}_{(x)k})}{2\sigma_k^2}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{Q}_{(x)j})^T(\boldsymbol{x}-\boldsymbol{Q}_{(x)j})}{2\sigma_j^2}\right)}$$

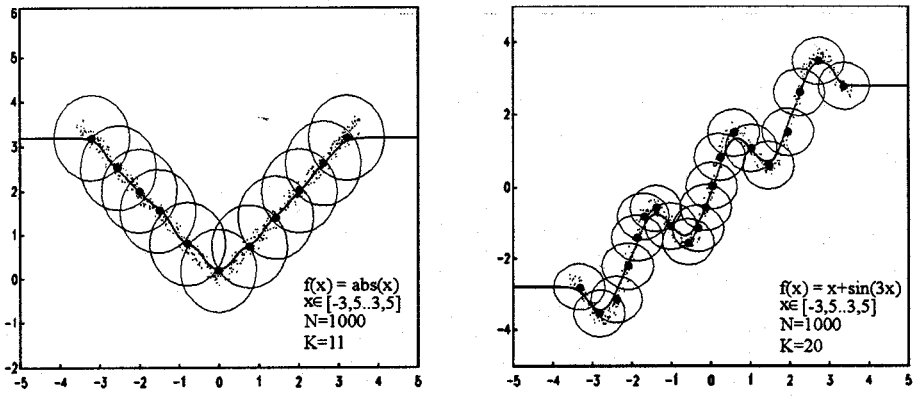

Figure 2: Approximation with the parametric RBFNN.

This type of networks has been introduced by [1], but a good discussion can be found in [6]. In fact, it is very often used recently and a lot of articles are dealing with it. Various learning schemes are applied to parametric RBFNN, from unsupervised, hybrid to completely supervised [5, 6].

ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

2.2.2. Parametric elliptical case

Instead of using the radial Gaussians as in the previous case, we utilized general multivariate Gaussians to represent the probability density function parametrically.

$$p(\boldsymbol{z}, \boldsymbol{Q}, \underline{\boldsymbol{\Sigma}}) = \frac{C}{K} \sum_{j=1}^{K} \frac{1}{\sqrt{|\underline{\boldsymbol{\Sigma}}_j|}} \exp\left(-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{Q}_j)^T \underline{\boldsymbol{\Sigma}}_j^{-1} (\boldsymbol{z} - \boldsymbol{Q}_j)\right) \qquad (6)$$

The vector $\boldsymbol{Q}_j$ and matrix $\underline{\boldsymbol{\Sigma}}_j$ describe the center vector and the receptive field of the j-th neuron, respectively. The sign $|\ \ |$ denotes the determinant of the matrix. To express regression estimator in a concise form, center vectors and width matrices are split into independent and dependent part

$$\boldsymbol{Q}_j = \begin{Vmatrix} \boldsymbol{Q}_{(x)j} \\ \boldsymbol{Q}_{(y)j} \end{Vmatrix} \qquad \underline{\boldsymbol{\Sigma}}_j = \begin{Vmatrix} \underline{\boldsymbol{\Sigma}}_{(xx)j} & \underline{\boldsymbol{\Sigma}}_{(xy)j} \\ \underline{\boldsymbol{\Sigma}}_{(yx)j} & \underline{\boldsymbol{\Sigma}}_{(yy)j} \end{Vmatrix}$$

In order to calculate regression estimator, the probability density function (6) must be inserted into the equation (2). After cumbersome algebraic manipulation we obtain recall rule for parametric EBFNN

$$\widehat{\boldsymbol{y}}(\boldsymbol{x}) = \sum_{k=1}^{K} \left(\boldsymbol{Q}_{(y)k} + \underline{\boldsymbol{\Sigma}}_{(yx)k} \underline{\boldsymbol{\Sigma}}_{(xx)k}^{-1} (\boldsymbol{x} - \boldsymbol{Q}_{(x)k})\right) \cdot \frac{|\underline{\boldsymbol{\Sigma}}_{(xx)k}|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{Q}_{(x)k})^T \underline{\boldsymbol{\Sigma}}_{(xx)k}^{-1} (\boldsymbol{x} - \boldsymbol{Q}_{(x)k})\right)}{\sum_{j=1}^{K} |\underline{\boldsymbol{\Sigma}}_{(xx)j}|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{Q}_{(x)j})^T \underline{\boldsymbol{\Sigma}}_{(xx)j}^{-1} (\boldsymbol{x} - \boldsymbol{Q}_{(x)j})\right)}$$

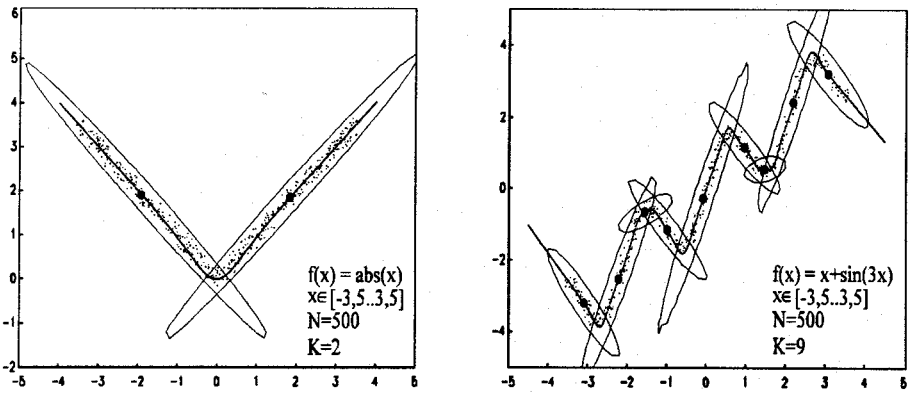

Figure 3: Approximation with the parametric EBFNN.

We have derived learning algorithms for all parameters in the EBFNN, [7, 8], but details are beyond the scope of this article. It should be only said that learning algorithms are based on an information criteria.

ESANN'1995 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 19-20-21 April 1995, D-Facto public., ISBN 2-9600049-3-0, pp. 309-314

3. Conclusions

A theoretical background of the LBFNN-s is based upon regularization technique of minimizing the estimation error [9], and is quite general and complete but complicated mathematically. On the other hand, our approach offers an alternative way to understanding the LBFNN. It stems from the localized representation of the probability distribution and can be more simply interpreted. Some of the known localized basis function estimators have been derived from the very basic statistical concept. Furthermore, a new localized basis function regression estimator has been obtained using general multivariate Gaussian representation of a probability density function.

Introduction of elliptical receptive fields of neurons in the LBFNN improves significantly the mapping properties of that network. The improvements are the most obvious in the extrapolation mapping domain what is evident from comparison of Fig.2 and Fig.3.

References

[1] Moody J., Darken C.: *Fast Learning in Networks of Locally-Tuned Processing Units.* Neural Computation, **1**, (281-294), 1989.

[2] Scot W.D.: *Multivariate Density Estiation - Theory, Practice and Visualization.* John Wiley & Sons, New York, 1992.

[3] Grabec I.: *Automatic Modeling of Physical Phenomena: Application to Ultrasonic Data.* Journal of Applied Physics, **69**, (6233-6244), 1991.

[4] Specht D.F.: *A General Regression Neural Network.* IEEE Trans. on Neural Networks, **NN-2**, (568-576), 1991.

[5] Grabec I.: *Self-Organization of Neurons Described by Maximum Entropy Principle.* Biological Cybernetics, **63**, (403-409), 1990.

[6] Stokbro K., Umberger D.K., Hertz J.A.: *Exploiting Neurons with Localized Receptive Fields to Learn Chaos.* Complex Systems **4**, (603-622), 1990.

[7] Kokol M.: *Modeling of Physical Phenomena by Regression Neural Network.* M.Sc. Thesis, University of Ljubljana - Department of Physics, Ljubljana, 1993. (In Slovene)

[8] Kokol M., Grabec I.: *Training of Elliptical Basis Function Neural Network.* Proceedings of World Congress on Neural Networks, San Diego, California, (Vol. II, 379-384), 1994.

[9] Pogio T., Girosi F.: *Networks for Approximation and Learning.* Proceedings of the IEEE, **78**, (1481-1497), 1990.