A Bayesian Local Linear Wavelet Neural Network

Kunikazu Kobayashi, Masanao Obayashi, and Takashi Kuremoto

Yamaguchi University, 2-16-1, Tokiwadai, Ube, Yamaguchi 755-8611, Japan
{koba,m.obayas,wu}@yamaguchi-u.ac.jp
http://www.nn.csse.yamaguchi-u.ac.jp/k/

Abstract. In general, wavelet neural networks have a problem on the curse of dimensionality, i.e. the number of hidden units to be required are exponentially rose with increasing an input dimension. To solve the above problem, a wavelet neural network incorporating a local linear model has already been proposed. On their network design, however, the number of hidden units is empirically determined and fixed during learning. In the present paper, a design method based on Bayesian method is proposed for the local linear wavelet neural network. The performance of the proposed method is evaluated through computer simulation.

1 Introduction

Wavelet neural networks (WNNs) have been mainly developed on signal processing and image processing [1]. The greatest characteristic of the WNNs is based on a temporally and spacially localized basis function, called mother wavelet. The localized basis function can identify the localization with arbitrary precision because both location and resolution can be freely adjusted by translation and dilation parameters. Radial basis function (RBF) networks also use a localized function [2], but it is difficult to adjust resolution with arbitrary precision.

Y. C. Pati et al. proposed a WNN introducing a discrete affine wavelet transform and presented the solutions to three problems on general feedforward neural networks, i.e. (1) how to determine the number of hidden units, (2) how to utilize information on training data and (3) how to escape local minima [3]. Q. Zhang et al. derived a WNN from a wavelet series expansion and mathematically proved that it can approximate any continuous functions [4]. K. Kobayashi et al. proposed a network design method utilizing a wavelet spectrum on training data [5] and another design method using genetic algorithm [6].

However, WNNs still have a problem on the curse of dimensionality, i.e. the number of hidden units rises exponentially as the number of input dimensions increases. To solve the above problem, T. Wang et al. introduced a local linear model into a WNN and developed a local linear wavelet neural network (LLWNN) [7]. After that, a lot of effort were mainly devoted into the LLWNN from parameter learning aspect. T. Wang et al. proposed two methods employing gradient descent method and genetic programing for parameter learning [7,8].

M. Köppen et al. (Eds.): ICONIP 2008, Part II, LNCS 5507, pp. 147–154, 2009.

Y. Chen et al. also presented two methods using gradient descent method and PSO (particle swarm optimization) for parameter learning [9,10] Most research, however, does not focused on the network design.

On the other hand, Bayesian method based on Bayesian statistics has applied to the network design of neural networks [11]. S. Albrecht et al. formulated a RBF network by Bayesian method and employed EM (Expectation-Maximization) algorithm for parameter learning [12]. C. Andrieu et al. also formulated a RBF network and approximated the integral calculation in posteriori distribution by MCMC (Markov Chain Monte Carlo) method [13]. N. Ueda et al. proposed a split and merge EM algorithm to network design and parameter learning for a mixture of experts model [14]. M. Sato et al. presented a design method based on variational Bayes method for a normalized Gaussian network (NGnet) [15]. C. C. Holmes et al. proposed a Bayesian design method for a feedforward neural network [16].

In the present paper, a Bayesian-based method for both network design and parameter learning of LLWNN is proposed. The proposed method basically follows a framework which proposed by C. C. Holmes et al. and tries to formulate a LLWNN. The LLWNN applying the proposed method is called a BLLWNN (Bayesian local linear wavelet neural network). Through computer simulation using function approximation problem, the performance of the BLLWNN is verified.

2 Local Linear Wavelet Neural Network (LLWNN)

Figure 1 shows a three-layered feedforward local linear wavelet neural network with N inputs and one output [7]. When an input vector $\mathbf{x} = \{x_1,\ x_2, \cdots, x_N\} \in R^N$ is given, LLWNN converts it to a weighted output $\hat{f}(\mathbf{x})$ with a local linear parameter c_k. That is, the output $\hat{f}$ of LLWNN is denoted by the follwoing equation.

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{K} c_k\, \Psi_k(\mathbf{x}) = \sum_{k=1}^{K} (w_{k0} + w_{k1}x_1 + \cdots + w_{kN}x_N)\, \Psi_k(\mathbf{x}). \tag{1}$$

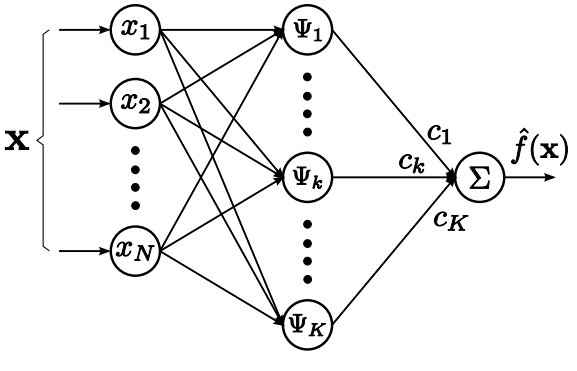

Fig. 1. Architecture of a local linear wavelet neural network

In (1), the basis function Ψ_k is defined by

$$\Psi_k(\mathbf{x}) = |\mathbf{a}_k|^{-\frac{1}{2}} \psi\left(\frac{\mathbf{x} - \mathbf{b}_k}{\mathbf{a}_k}\right), \tag{2}$$

where $\mathbf{a}_k \in R^N$ and $\mathbf{b}_k \in R^N$ refer to dilation and translation parameters, respectively. Then, the basis function ψ in (2) must satisfy the following admissible condition.

$$\int_R \frac{|\hat{\Psi}(\lambda)|^2}{\lambda} d\lambda < \infty, \tag{3}$$

where $\hat{\Psi}(\lambda)$ means Fourier transform of $\Psi(\mathbf{x})$. For a localized basis function, (3) is equivalent to $\int_R \psi(x)dx = 0$. That is, the basis function has no direct current component.

The difference between general WNN and LLWNN depends on the parameter c_k. The parameter c_k is scalar for WNN and a local linear model for LLWNN. This allows LLWNN could cover larger area in input space compared with WNN. As a result, LLWNN may resolve the curse of dimensionality because it can reduce the numbers of hidden units and free parameters. Furthermore, it is clarified that the local linear model realizes good interpolation even if the number of learning samples is small [7].

On network design of LLWNN, however, almost all the models empirically determine the number of hidden units [7,8,9,10]. Namely, the network structure is determined in advance and fixed during learning. Y. Chen et al. has tried to construct a LLWNN using eCGP (Extended Compact Genetic Programming) [17]. However, since they represented the network as a hierarchical structure, it seems that the numbers of hidden units and free parameters tend to be large [10].

In the present paper, a Bayesian method is applied to network design and parameter learning of LLWNN.

3 Bayesian Design Method for LLWNN

3.1 Bayesian Method

First of all, D, x, y and θ denote observed data $D = (x, y)$, training data, desired data and unknown parameter, respectively. The Bayesian method follows the following procedure [18].

(1) Modeling $p(\cdot|\theta)$
(2) Determine a prior distribution $p(\theta)$
(3) Observe data $p(D|\theta)$
(4) Calculate a posteriori distribution $p(\theta|D)$
(5) Estimate a predictive distribution $q(y|x, D)$

The posteriori distribution $p(\theta|D)$ is derived using the following Bayes' theorem.

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}. \tag{4}$$

Then, if the complexity of model is known, the predictive distribution $q(y|x, D)$ is derived as

$$q(y|x, D) = \int p(y|x, \theta)p(\theta|D)d\theta, \tag{5}$$

and if the the complexity of the model, m, is unknown, it is derived as

$$q(y|x, D) = \sum_m \int p(y|x, \theta, m)p(\theta, m|D)d\theta. \tag{6}$$

The Bayesian method estimates the parameters of observed data to be approximated as the posteriori distribution of parameters [11]. Its characteristics are listed as follows.

- It can utilize a priori knowledge as the prior distribution.
- It shows high generalization ability even if training data is small.
- It can evaluate the reliability of the predictive value.
- It can automatically select models.

On the other hand, the Bayesian method has a serious problem on the integral calculation in posteriori distribution. It requires any approximation methods or any numerical calculation methods. So far, four method, (1) Laplace approximation method [18], (2) mean field method [19], (3) MCMC (Markov Chain Monte Carlo) method [20] and (4) variational Bayes method [21] have been proposed.

A method to be proposed in the present paper utilizes the above MCMC method. In the present paper, as both network design and parameter learning of LLWNN are conducted and then the size of model is unknown, the reversible jump MCMC method [20] is employed

3.2 Proposed Bayesian Design Method

In this section, the proposed Bayesian method for LLWNN is described. The proposed method follows a framework by C. C. Holmes et al. [16] and formulates a LLWNN. In the present paper, the LLWNN by applying the proposed method is called a BLLWNN (Bayesian Local Linear Wavelet Neural Network).

At first, data set $D = \{(\mathbf{x}_i, y_i) \,|\, i = 1 \sim n\}$ is defined by input data $\mathbf{x}_i \in R^N$ and output data $y_i \in R$. The relation between $\mathbf{x}_i$ and y_i is represented by

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \tag{7}$$

where ϵ_i referes to a noise term. After that, the input-output characteristic $f(\cdot)$ is modeled by (1). Then, a model M_k and model space M are written as

$$M_k = \{a_1, \mathbf{b}_1, \cdots, a_k, \mathbf{b}_k\}, \tag{8}$$

$$M = \{k, M_k, W\}, \tag{9}$$

respectively, where $W = (\mathbf{w}_1, \cdots, \mathbf{w}_k)$ means a weight matrix and a vector element $\mathbf{w}_k$ is written as $\mathbf{w}_k = (w_{k0}, w_{k1}, \cdots, w_{kN})'$ ($\mathbf{w}'$ refers to a transpose of $\mathbf{w}$). Furthermore, an unknown parameter θ_k of LLWNN is written as $\theta_k = \{a_k, \mathbf{b}_k, \mathbf{w}_k\}$.

The predictive distribution is derived by averaging predictive output $\hat{f}_M(\mathbf{x})$ with the posteriori distributions $p(W|M_k, D)$ and $p(M_k|D)$.

$$p(y|\mathbf{x}, D) = \sum_k \iint \hat{f}_M(\mathbf{x}) p(W|M_k, D) p(M_k|D) dM_k dW, \tag{10}$$

where $p(W|M_k, D)$ and $p(M_k|D)$ are written as

$$p(W|M_k, D) = \frac{p(D|W, M_k)p(W)}{p(D)}, \tag{11}$$

$$p(M_k|D) = \frac{p(D|M_k)p(M_k)}{p(D)}. \tag{12}$$

In (7), if the noise term ϵ_i follows a normal distribution with mean 0 and variation σ^2, $p(D|W, M_k)$ in (11) is derived as

$$\begin{aligned} p(D|W, M_k) &= \log \prod_{i=1}^{N} p(\epsilon_i) \\ &= \log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - f(\mathbf{x}_i, M_k, W))^2}{2\sigma^2}\right) \\ &= -\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{N}\{y_i - f(\mathbf{x}_i, M_k, W)\}^2. \end{aligned} \tag{13}$$

In the present paper, prior distributions $p(W)$ in (11) and $p(M)$ in (12) are assumed by the following normal and gamma distributions, respectively.

$$p(W) = N(W|0, \lambda^{-1}I), \tag{14}$$

$$p(M) = Ga(DF|\alpha, \beta), \tag{15}$$

where DF means a degrees of freedom of model and is defined by

$$DF = tr\left(\Psi(\Psi'\Psi + \lambda I)^{-1}\Psi'\right), \tag{16}$$

where $tr(S)$ refers to a trace of matrix S. From (14) and (15),

$$\begin{aligned} p(M, W) &= Ga(DF|\alpha, \beta)N(W|0, \lambda^{-1}I) \\ &\propto DF^{\alpha-1}\exp\left(-\beta DF\right)\exp\left(-\frac{\lambda}{2}||W||^2\right). \end{aligned} \tag{17}$$

In the present paper, the integral calculation in (10) is solved using the reversible jump MCMC method, which is a type of MCMC that allows for

dimensional changes in the probability distribution being simulated [20]. Therefore, (10) is approximated by

$$p(y|\mathbf{x}, D) \approx \frac{1}{n_s - n_0} \sum_{t=n_0}^{n_s} f_{\pi_t}(\mathbf{x}), \tag{18}$$

where π_t is a Markov chain sample followed by the probability distribution $p(\pi|D)$, n_0 is the number of samples in burn-in time and n_s is the total number of samples. In the present paper, a Metropolis-Hastings algorithm [11] is used for sampling.

4 Computer Simulation

This section describes computer simulation to evaluate the performance of the proposed method.

In the simulation, a function approximation problem was used. The following two-dimensional continuous function used by T. Wang et al. [7] was employed.

$$f(x_1, x_2) = \frac{\sin(\pi x_1)\cos(\pi x_2) + 1.0}{2.0}, \tag{19}$$

where the domains of input variables x_1 and x_2 were set to $x_1,\ x_2 \in [-1.0, 1.0]$. The training data set consists of 49 input points (x_1, x_2) which generated by equally spaced on a 7×7 grid in $[-1.0, 1.0] \times [-1.0, 1.0]$ and corresponding output $f(x_1, x_2)$. Then, testing data set consists of 400 input points which generated by equally spaced on a 20×20 grid in the same domain.

The basis function, i.e. ψ in (2) is defined by

$$\psi(x) = -x \exp\left(-\frac{x^2}{2}\right). \tag{20}$$

Then, function Ψ is written as tensor product of ψ for input x_i as follows.

$$\Psi(\mathbf{x}) = \prod_{i=1}^{N} \psi(x_i). \tag{21}$$

The performance was evaluated by three factors, i.e. the number of hidden units, the number of free parameters and the root-mean-square error (RMSE) defined by

$$\mathrm{RMSE} = \sqrt{\frac{1}{N_p} \sum_{j=1}^{N_p} \left\{ f(x_j) - \hat{f}(x_j) \right\}^2}, \tag{22}$$

where $f(x_j)$ and $\hat{f}(x_j)$ are desired and predictive values for input x_j, respectively and N_p is the number of training data.

Table 1. Parameter setting

Parameter	Value
σ	1.0
λ	0.01
α	0.1
β	0.1
n_s	10000
n_0	5000

Table 2. Simulation result

Method	# of hidden units	# of free parameters	RMSE
WNN	8	40	1.65×10^{-2}
LLWNN	4	28	1.58×10^{-2}
BLLWNN	4	28	1.56×10^{-2}

The parameter setting and the simulation results are shown in Table 1 and 2, respectively. In Table 2, the results of general WNN and LLWNN [7] are also provided to compare with the proposed method, i.e. BLLWNN. Both WNN and LLWNN empirically determine the number of hidden units. The BLLWNN, however, could automatically determine the numbers of hidden units and free parameters because of the Bayesian method.

As shown in Table 2, BLLWNN is much better than WNN. Furthermore, the BLLWNN shows almost the same performance with LLWNN. Therefore, BLLWNN is superior than LLWNN and WNN because it could automatically build network structure and determine its parameters.

5 Summary

In the present paper, the Bayesian method to determine both network structure and parameter of LLWNN has been proposed. Through computer simulation using function approximation problem, the effectiveness of the proposed method is confirmed. The evaluation for higher-dimensional functions and time-series prediction are future work.

References

1. Chui, C.K.: An Introduction to Wavelets. Academic Press, London (1992)
2. Poggio, T., Girosi, F.: Networks for approximation and learning. Proc. of the IEEE 78(9), 1481–1497 (1990)
3. Pati, Y.C., Krishnaprasad, P.S.: Discrete affine wavelet transformations for analysis and synthesis of feedforward neural networks. In: Advances in Neural Information Processing Systems, vol. 3, pp. 743–749. MIT Press, Cambridge (1991)

4. Zhang, Q., Benveniste, A.: Wavelet networks. IEEE Trans. on Neural Networks 3(6), 889–898 (1992)
5. Kobayashi, K., Torioka, T., Yoshida, N.: A Wavelet Neural Network with Network Optimizing Function. Systems and Computers in Japan 26(9), 61–71 (1995)
6. Ueda, N., Kobayashi, K., Torioka, T.: A Wavelet Neural Network with Evolutionally Generated Structures. Trans. on the Institute of Electronics, Information and Communication Engineers J80-D-II(2), 652–659 (1997) (in Japanese)
7. Wang, T., Sugai, Y.: A local linear adaptive wavelet neural network. Trans. on the Institute of Electrical Engineers of Japan 122-C(2), 277–284 (2002)
8. Wang, T., Sugai, Y.: The local linear adaptive wavelet neural network with hybrid ep/gradient algorithm and its application to nonlinear dynamic system identification. Trans. on the Institute of Electrical Engineers of Japan 122-C(7), 1194–1201 (2002)
9. Chen, Y., Dong, J., Yang, B., Zhang, Y.: A local linear wavelet neural network. In: Proc. of the 5th World Congress on Intelligent Control and Automation, pp. 1954–1957 (2004)
10. Chen, Y., Yang, B., Dong, J.: Time-series prediction using a local linear wavelet neural network. Neurocomputing 69, 449–465 (2006)
11. MacKay, D.: Information Theory, Inference, and Learning Algorithms. Cambridge University Press, Cambridge (2003)
12. Albrecht, S., Busch, J., Kloppenburg, M., Metze, F., Tavan, P.: Generalized radial basis function networks for classification and novelty detection: Self-organization of optimal Bayesian decision. Neural Networks 13, 755–764 (2000)
13. Andrieu, C., de Freitas, N., Doucet, A.: Robust full Bayesian learning for radial basis networks. Neural Computation 13, 2359–2407 (2001)
14. Ueda, N., Nakano, R., Ghahramani, Z., Hinton, G.E.: Split and merge EM algorithm for improving Gaussian mixture density estimates. In: Proc. of the 1998 IEEE Signal Processing Society Workshop, pp. 274–283 (1998)
15. Sato, M.: Online model selection based on the variational Bayes. Neural Computation 13, 1649–1681 (2001)
16. Holmes, C.C., Mallick, B.K.: Bayesian radial basis functions of variable dimension. Neural Computation 10, 1217–1233 (1998)
17. Sastry, K., Goldberg, D.E.: Probabilistic Model Building and Competent Genetic Programming, pp. 205–220. Kluwer, Dordrecht (2003)
18. MacKay, D.J.C.: Bayesian interpolation. Neural Computation 4, 415–447 (1992)
19. Peterson, C., Anderson, J.R.: A Mean Field Theory Learning Algorithm for Neural Networks. Complex Systems 1, 995–1019 (1987)
20. Green, P.J.: Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrika 82, 711–732 (1995)
21. Attias, H.: Inferring parameter and structure of latent variable models by variational Bayes. In: Proc. of the 15th Conf. on Uncertainty in Artificial Intelligence, pp. 21–30 (1999)