Bayesian Neural Networks with Correlating Residuals

Aki Vehtari and Jouko Lampinen
Aki.Vehtari@hut.fi, Jouko.Lampinen@hut.fi
Laboratory of Computational Engineering, Helsinki University of Technology
P.O.Box 9400, FIN-02015 HUT, FINLAND

Abstract

Usually in multivariate regression problem it is assumed that residuals of outputs are independent of each other. In many applications a more realistic model would allow dependencies between the outputs. In this paper we show how a Bayesian treatment using Markov Chain Monte Carlo (MCMC) method can allow for a full covariance matrix with Multi Layer Perceptron (MLP) neural networks.

1 Introduction

In regression problems, it is generally assumed that the distribution of target data can be described by a deterministic function of inputs, together with additive Gaussian noise with a constant covariance. Different parameterizations of the covariance matrix correspond to different assumptions about the noise. Usually in the case of multivariate data, noises of different outputs are assumed independent with common or independent noise levels. In many applications it is more realistic to allow dependencies between the outputs. This can be achieved with full covariance matrix.

Use of full covariance matrix with maximum a posteriori (MAP) approach has been discussed by Williams [13]. However, MAP gives biased results (e.g., noise variance being systematically under-estimated) and requires large number of samples in order to get good results. These limitations can be overcome in a Bayesian treatment. Bayesian neural networks with independent output noises have been discussed by MacKay [8] and Neal [9, 10].

Purpose of this paper is to show how this problem can be solved using full covariance matrix with Bayesian treatment and Markov Chain Monte Carlo (MCMC) methods. We begin by briefly reviewing the Bayesian neural networks and MCMC implementation in Sections 2 and 3. In Section 4 we describe the use of full covariance matrix. In Section 5, we briefly review Bayesian Deviance Information Criterion used for model comparison. Numerical experiment illustrating the use of full covariance matrix is provided in Section 6.

2 Bayesian Neural Networks

Consider a multivariate regression problem involving the prediction of a noisy vector $\mathbf{y}$ of target variables given the value of a vector $\mathbf{x}$ of input variables.

The process of Bayesian learning is started by defining a model, and prior distribution $p(\theta)$ for the model parameters. Prior distribution expresses our initial beliefs about parameter values, before any data has been observed. After observing new data $D = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}$, prior distribution is updated to the posterior distribution using Bayes' rule

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} \propto L(\theta|D)p(\theta), \tag{1}$$

where likelihood function $L(\theta|D)$ gives the probability of the observed data as function of the unknown model parameters. In case of independent and exchangeable data points we get

$$L(\theta|D) = \prod_{i=1}^{n} p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}, \theta), \tag{2}$$

where n is the number of data points.

We consider an MLP network with weights $\mathbf{w}$. MLP takes the input vector $\mathbf{x}$ and generates an output $\mathbf{o} = f(\mathbf{x}, \mathbf{w})$ which represents the regression function. In regression problems, it is generally assumed that the distribution of target data can be described by a deterministic function of inputs, corrupted by additive Gaussian noise with a constant covariance matrix. Probability density for a target $\mathbf{y}$ is then

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \Sigma) = (2\pi)^{-d/2}|\Sigma|^{-1/2} \exp\left(-\tfrac{1}{2}(\mathbf{y}-\mathbf{o})^T\Sigma^{-1}(\mathbf{y}-\mathbf{o})\right), \tag{3}$$

where Σ is a $d \times d$ noise covariance matrix.

Using different parameterizations of the covariance matrix, we make different assumptions about the noise. Parameterizations which have been generally used are constant diagonal ($\sigma^2 I$) and diagonal (diag$(\sigma_1^2, \ldots, \sigma_d^2)$) covariance matrices. These correspond to assuming independent noise

between outputs, with common noise level or independent noise levels, respectively. Probability densities for a target $\mathbf{y}$ are given respectively by

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-1/2} \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{o})^2\right) \quad (4)$$

$$p(y_j|\mathbf{x}, \mathbf{w}, \sigma_j^2) = (2\pi\sigma_j^2)^{-1/2} \exp\left(-\frac{1}{2\sigma_j^2}(y_j - o_j)^2\right). \quad (5)$$

Note that the use of constant diagonal covariance with maximum likelihood approach corresponds to the minimization of the sum-of-squares error.

Convenient conjugate prior, which allows us easily to define our prior knowledge (or ignorance) about noise variance, is given generally by

$$p(\sigma^2) = \text{Inv-Gamma}(\sigma^2|\nu_0, \sigma_0^2) \quad (6)$$

$$\propto (\sigma^2)^{-(\nu_0/2+1)} \exp\left(-\tfrac{1}{2}\nu_0\sigma_0^2\sigma^{-2}\right), \quad (7)$$

where Inv-Gamma is inverse Gamma distribution with ν degrees of freedom[1]. In this way, prior belief about σ is equivalent to ν_0 prior measurements having common variance σ_0. Sometimes prior distributions are specified in terms of corresponding precisions $\tau = \sigma^{-2}$, with given Gamma distributions. (Inverse) Gamma distribution in Bayesian analysis has been discussed, e.g., by Box & Tiao [1] and Gelman et al. [4]. Gamma distribution in connection with MLPs has been discussed by Neal [10].

Following [10] we use four independent Gaussian prior distributions for different weight groups in MLP. Gaussians have fixed zero mean and variable variances with vague inverse Gamma distributions as hyperpriors.

To predict the new output $\mathbf{y}^{n+1}$ for new input $\mathbf{x}^{n+1}$, predictive distribution is obtained by integrating the predictions of the model with respect to the posterior distribution of the model parameters

$$p(\mathbf{y}^{(n+1)}|\mathbf{x}^{(n+1)}, D) = \int p(\mathbf{y}^{(n+1)}|\mathbf{x}^{(n+1)}, \theta)p(\theta|D)\, d\theta. \quad (8)$$

Expectations can be evaluated with respect to the posterior distribution for parameters, for example for a $\mathbf{y}^{(n+1)}$

$$\hat{\mathbf{y}}^{(n+1)} = \int f(\mathbf{x}^{(n+1)}, \mathbf{w})p(\theta|D)\, d\theta. \quad (9)$$

3 Markov Chain Monte Carlo

The posterior distributions in case of MLPs are typically very complex and above integrals are very difficult to evaluate. Neal [9, 10] has introduced Markov Chain Monte Carlo (MCMC) implementation of Bayesian learning for neural networks. In MCMC the integrations required by Bayesian approach are approximated numerically using a sample of values drawn from the posterior distribution of parameters. For example (9) is approximated by

$$\hat{\mathbf{y}}^{(n+1)} \approx \frac{1}{N}\sum_{t=1}^{N} f(\mathbf{x}^{(n+1)}, \mathbf{w}^{(t)}), \quad (10)$$

where $\mathbf{w}^{(t)}$ are samples of weights. In MCMC, samples are generated using a Markov chain that has the desired posterior distribution as its equilibrium distribution.

Neal has used the hybrid Monte Carlo (HMC) algorithm [2] for parameters and Gibbs sampling [5, 3, 7] for hyperparameters. HMC is an elaborate Metropolis-Hastings Monte Carlo method, which makes efficient use of gradient information to reduce random walk behavior. The gradient indicates in which direction one should go to find states with high probability. Gibbs sampling is perhaps the simplest MCMC method. In a single iteration, Gibbs sampling involves sampling one parameter at time from full conditional distribution given all other parameters. Use of Gibbs sampling for hyperparameters helps to minimize the amount of tuning that is needed to obtain good performance in HMC.

With conjugate priors above the full conditional distributions needed here by Gibbs sampling have the same form as the conjugate priors. In general, σ^2 specifies the variances for the independent Gaussian distributions of n lower level quantities z_i. In this situation, it can be shown that full conditional distribution $p(\sigma^2|\{z_i\})$ is inverse gamma distribution with parameters

$$\nu_n = \nu_0 + n \quad (11)$$

$$\sigma_n^2 = (\nu_0\sigma_0^2 + \sum_i z_i^2)/(\nu_0 + n) \quad (12)$$

Many software packages generate gamma random variables directly. See, e.g., [6] for the algorithms.

4 Full covariance

Neal discusses and demonstrates his methods using diagonal covariance matrix. Datasets used in [10, 11] do not have multivariate targets or they are artificial data with independent noises. We show how full covariance matrix can be used with Bayesian neural networks described above.

Using full covariance matrix Σ in (4) assumes that each output has independent noise level and noises between outputs may have linear dependencies.

[1] Note that different references have different parameterizations of the (inverse) Gamma distribution.

Convenient conjugate prior for full covariance matrix is given by inverse Wishart distribution

$$p(\Sigma) = \text{Inv-Wishart}(\Sigma|\nu_0, \Sigma_0) \quad (13)$$
$$\propto |\Sigma|^{-(\nu_0+d+1)/2} \exp\left(-\tfrac{1}{2}\nu_0 \text{tr}(\Sigma_0\Sigma^{-1})\right), \quad (14)$$

where $\text{tr}(\cdot)$ denotes the trace of a matrix argument[2][3]. Inverse Wishart is a multivariate generalization of the inverse Gamma distribution. The prior distribution could be specified in terms of the corresponding precision matrix $T = \Sigma^{-1}$, with given Wishart distribution. (Inverse) Wishart distribution generally in Bayesian analysis has been discussed, e.g., in [1, 4].

Full conditional distribution $p(\Sigma|\{\mathbf{z}_i\})$ is inverse Wishart distribution with parameters

$$\nu_n = \nu_0 + n \quad (15)$$
$$\Sigma_n = (\nu_0\Sigma_0 + \sum_i \mathbf{z}_i\mathbf{z}_i^T)/(\nu_0 + n). \quad (16)$$

When $\nu \geq d$ and ν is integer, sampling from the Wishart distribution can be easily accomplished by simulating $\alpha_1, \ldots, \alpha_\nu$, ν independent samples from a d-dimensional multivariate $\text{N}(0, \Sigma_n)$ distribution, and let $\Sigma = \frac{1}{\nu}\sum_{i=1}^{\nu} \alpha_i\alpha_i^T$ [4]. A general algorithm handling also $\nu < d$ or non-integer ν is described in [6].

5 Deviance Information Criterion

To compare different noise models we calculate mean square error (MSE) of the test data and Bayesian Deviance Information Criterion (DIC). DIC has been recently proposed by Spiegelhalter et al. [12] for comparison of arbitrarily complex Bayesian models. DIC has been shown to have several theoretical justifications, but Spiegelhalter et al. emphasize that DIC is not recommended to be used as a strict criterion for model choice.

DIC is based on comparisons on the posterior distributions of the deviance

$$D(\theta) = -2\log p(y|\theta) + 2\log f(y), \quad (17)$$

where y is observed data and θ are the lowest-level parameters directly influencing the fit. A standardizing term $f(y)$ is a function of the data alone and hence does not affect model comparison. The fit of a model is summarized by the posterior expectation of the deviance

$$\bar{D} = E_{\theta|y}[D]. \quad (18)$$

[2]Here $E(\Sigma) = \frac{\nu_0}{\nu_0-k-1}\Sigma_0$.

[3]Note that different references have different parameterizations of the Wishart distribution.

The model complexity is measured by the effective number of parameters p_D, defined as

$$p_D = E_{\theta|y}[D] - D(E_{\theta|y}[\theta]) \quad (19)$$
$$= \bar{D} - D(\bar{\theta}). \quad (20)$$

The fit and complexity are then added to form a Deviance Information Criterion

$$\text{DIC} = \bar{D} + p_D \quad (21)$$
$$= D(\bar{\theta}) + 2p_D. \quad (22)$$

Above quantities can be easily obtained from the MCMC analysis.

6 Numerical experiment

As an illustration of different covariance matrix parameterizations we consider a toy problem involving one input x and two outputs $y_1 = x^2, y_2 = 0.5 - 0.5x$. Noise with different covariances (see Fig. 1) were added to the targets. Since the estimated quantities are noisy, due to the finite data sets, we repeated every experiment 10 times and averaged the results. For each different noise covariance we generated 10 independent data sets with 8 to 20 (see Fig. 1) data points for training. For the testing we generated one data set with 1000 data points.

We used one hidden layer MLP with 10 hidden units and vague priors for weights. We used HMC with persistent momentum and heuristic choice of stepsizes as described in [10]. Hyperparameters for priors of noise covariance matrices were $\nu_0 = 1$, $\sigma_0^2 = 0.02$, $\sigma_{j,0} = 0.02$, and $\Sigma = 0.02 \times I$. 1000 HMC chains of length 50 were simulated. Hyperparameters were sampled with Gibbs sampling after each chain, except for the first 20 chains they were fixed to prior mean values. Samples were collected after each chain and latter half of samples were used for prediction and calculating DIC values.

Fig. 1 shows that when the outputs do not correlate (top row left) different covariance matrix parameterizations produce similar test MSEs. When the correlation increases (from left to right) full covariance matrix produces increasingly better results.

In all test cases where output noises correlate, DIC values (Fig. 1, bottom row) implies correctly that full covariance model is best. When output noises do not correlate (first column), full and diagonal covariance matrices are equally good. As the amount of training data increases, test MSE of all models seems to decrease to almost the same value. This shows that if we are only interested in the expectation

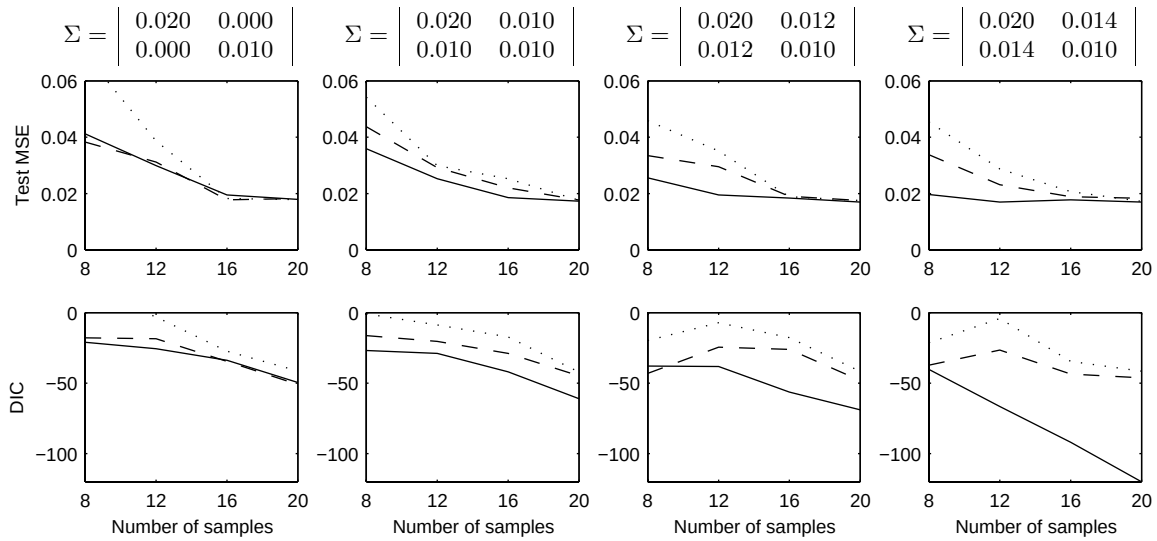

Figure 1: Averaged over 10 runs, test MSE (top row) and DIC (bottom row), for different noise covariances Σ and training data sizes with different noise covariance matrix parameterizations: *dotted)* constant diagonal, *dashed)* diagonal, *solid)* full. Note that DIC values with different number of training points are not directly comparable.

of the model output and we have enough data, we can get good results with approximative noise models. If we are interested in predicting noise covariance, we may use DIC to point out the best model. Also note that DIC can be used when separate test data is not available and we do not want to use resampling methods like cross validation to estimate test error.

7 Conclusions

We have shown how MLP neural networks with full covariance matrix can be treated in Bayesian way using MCMC methods. Using inverse Wishart prior distribution, full covariance model is easy to include in Bayesian neural network framework developed by Neal [10].

Acknowledgments

This study was partly funded by TEKES Grant 40888/97 (Project *PROMISE, Applications of Probabilistic Modeling and Search*).

References

[1] Box, G. E. P. & Tiao, G. C. (1973). *Bayesian inference in statistical analysis*. John Wiley and Sons, Inc.

[2] Duane, S., Kennedy, A. D., Pendleton, B. J. & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195:pp. 216–222.

[3] Gelfand, A. E., Hills, S. E., Racine-Poon, A. & Smith, A. F. M. (1990). Illustartion of bayesian inference in normal data models using gibbs sampling. *Journal of the American Statistical Association*, 85(412):pp. 972–985.

[4] Gelman, A., Carlin, J. B., Stern, H. S. & Rubin, D. R. (1995). *Bayesian Data Analysis*. Texts in Statistical Science. Chapman & Hall.

[5] Geman, S. & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:pp. 721–741.

[6] Gentle, J. E. (1998). *Random Number Generation and Monte Carlo Methods*. Springer-Verlag.

[7] Gilks, W., Richardson, S. & Spiegelhalter, D., eds. (1996). *Markov Chain Monte Carlo in Practice*. Chapman & Hall.

[8] MacKay, D. J. C. (1992). A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):pp. 448–472.

[9] Neal, R. M. (1992). Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Tech. Rep. CRG-TR-92-1, Dept. of Computer Science, University of Toronto.

[10] Neal, R. M. (1996). *Bayesian Learning for Neural Networks*, vol. 118 of *Lecture Notes in Statistics*. Springer-Verlag.

[11] Neal, R. M. (1998). Assessing relevance determination methods using DELVE. In C. M. Bishop, ed., *Neural Networks and Machine Learning*, vol. 168 of *NATO ASI Series F: Computer and Systems Sciences*. Springer-Verlag.

[12] Spiegelhalter, D. J., Best, N. G. & Carlin, B. P. (1998). Bayesian deviance, the effective number of parameters, and the comparison of arbitrarily complex models. Tech. Rep. 98-009, Division of Biostatistics, University of Minnesota.

[13] Williams, P. M. (1996). Using neural networks to model conditional variate densities. *Neural Computation*, 8(4):pp. 843–854.