# Empirical Evaluation of Bayesian Sampling for Neural Classifiers

Dirk Husmeier, William D. Penny, Stephen J. Roberts
Department of Electrical and Electronic Engineering, Imperial College
London, United Kingdom

**Abstract**

Adopting a Bayesian approach and sampling the network parameters from their posterior distribution is a rather novel and promising method for improving the generalisation performance of neural network predictors. The present empirical study applies this scheme to a set of different synthetic and real-world classification problems. The paper focuses on the dependence of the prediction results on the prior distribution of the network parameters and hyperparameters, and provides a critical evaluation of the automatic relevance determination (ARD) scheme for detecting irrelevant inputs.

## 1 Introduction

Consider a $K$-fold classification problem[1], where an $m$-dimensional feature vector $\mathbf{x}_t$ is assigned to one of K classes $\{C_1, \ldots, C_K\}$ indicated by a binary label vector $y_t^k = 1$ if $\mathbf{x}_t \in C_k$, $y_t^k = 0$ if $\mathbf{x}_t \notin C_k$, $||\mathbf{y}_t|| = 1$. For a neural network (NN) with $K$ softmax units in the final layer, the network outputs $f_k(\mathbf{x}_t; \mathbf{w}) \in [0, 1]$ can be interpreted as predictors for the class-conditional probabilities, $P(y_t^k = 1|\mathbf{x}_t, \mathbf{w}) = f_k(\mathbf{x}_t; \mathbf{w})$, where $\mathbf{w}$ denotes the vector of all network weights (including biases). The likelihood for the entire training set $\mathbf{D} = \{(\mathbf{x}_t, \mathbf{y}_t)\}_{t=1}^{t=N}$ is given by

$$P(\mathbf{D}|\mathbf{w}) \quad = \quad \prod_{t=1}^{N} \prod_{k=1}^{K} f_k(\mathbf{x}_t; \mathbf{w})^{y_t^k} \tag{1}$$

and conventional network training schemes aim at finding the (local) maximum of this function or, equivalently, the (local) minimum of the energy

$$E(\mathbf{w}) \ := \ -\ln P(\mathbf{D}|\mathbf{w}) \ = \ \sum_{t=1}^{N} \sum_{k=1}^{K} y_t^k \ln f_k(\mathbf{x}_t; \mathbf{w}). \tag{2}$$

From a Bayesian perspective, this approach is equivalent to finding the mode of the posterior distribution $P(\mathbf{w}|\mathbf{D})$ under the assumption of a flat prior $P(\mathbf{w}) = const$. The problem of this approach is that, for a sparse training set, this mode is not representative of the distribution as a whole or, in the language of neural computation, that the network overfits the training data. A conventional remedy is to introduce different weight groups, labeled by $g \in \{1, \ldots, G\}$, and add a weight-decay penalty term $R(\mathbf{w}; \boldsymbol{\alpha}) = \frac{\alpha_g}{2} \sum_{i_g=1}^{W_g} w_{i_g}^2$ to the energy

[1] This paper only deals with classification problems, but the results equally apply to regression tasks.

(2). Within a Bayesian framework this can be interpreted as a (Gaussian) prior distribution of the weights, $P(\mathbf{w}|\boldsymbol{\alpha}) \propto \exp[-R(\mathbf{w};\boldsymbol{\alpha})]$. Some authors suggest optimising the so-called hyperparameters or weight-decay constants $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_G)$ by maximum likelihood [1]. A proper Bayesian approach, however, is to integrate over all the parameters and hyperparameters to obtain the class-conditional probability conditional only on the data $\mathbf{D}$, as in [2]:

$$P(y^k|\mathbf{x},\mathbf{D}) = \int P(y^k|\mathbf{w},\mathbf{x})P(\mathbf{w},\boldsymbol{\alpha}|\mathbf{D})d\mathbf{w}d\boldsymbol{\alpha} = \int f_k(\mathbf{x};\mathbf{w})P(\mathbf{w},\boldsymbol{\alpha}|\mathbf{D})d\mathbf{w}d\boldsymbol{\alpha} \quad (3)$$

This integral can be solved numerically by Markov chain Monte Carlo (MCMC), that is, by constructing a Markov chain $\{(\mathbf{w}_n, \boldsymbol{\alpha}_n)\}_n$ which converges to a sample drawn from the posterior distribution $P(\mathbf{w},\boldsymbol{\alpha}|\mathbf{D})$ and then, after a sufficiently long equilibration period, approximating the integral by a sum over this sample, $P(y^k|\mathbf{x},\mathbf{D}) \approx \frac{1}{T}\sum_{n=1}^{T} f_k(\mathbf{x};\mathbf{w}_n)$.

## 2 Prior distribution

The performance of the Bayesian scheme hinges on two critical factors. The first concerns the efficiency of the sampling process and the Monte-Carlo approximation of the integral in (1). Neal [2] has introduced a powerful scheme termed *hybrid-chain Monte Carlo* to the neural network community, which we applied in all our simulations [2]. The second factor is more fundamental and concerns the prior distribution of the parameters (weights and biases) and hyperparameters (weight-decay constants). This inclusion of subjective prior knowledge is inherent to the whole Bayesian approach and has, at least for sparse data, a significant influence on the inference scheme. Yet while in conventional statistical applications the incorporation of prior knowledge can usually be done explicitly because the parameters of interest have *per se* some physical meaning, the neural network approach is troubled with the problem that the prior distribution is defined in the 'wrong' space. Given that the network ultimately implements an input-output mapping trying to fit a limited set of observations or measurements, one would like to specify a prior distribution in the space of functions itself, rendering, for instance, smooth mappings more likely than those with a large curvature. What one needs to do, though, is to decide on a prior in the parameter and hyperparameter space, which usually defies the incorporation of prior domain-specific knowledge. The aim of this study, therefore, is to investigate the influence of these priors on the generalisation performance. Two different priors for the weights were tried out.

- **ARD:** Weights on connections between the input and the hidden layers are divided into seperate weight groups. Weights exiting the $g$th input unit are drawn from a zero-mean Gaussian with variance $\alpha_g^{-1}$. If an input is irrelevant, the coresponding $\alpha_g$ (the weight-decay constant for the $g$th weight group) may grow large and thus effectively switch this contribution

[2] Our simulations were carried out with Neal's programs, available from `http://www.cs.utoronto.ca/~radford`.

off. This scheme was developed by MacKay and Neal [2], who referred to it as *automatic relevance determination* (ARD).

- **No ARD:** The weights between the input and the hidden layers form one single weight group with a common Gaussian prior distribution.

The remaining parameters, that is, the weights between the hidden and output layers, the hidden unit biases, and the output unit biases, were given seperate zero-mean Gaussian priors. For the hyperparameters (weight-decay constants), we tried out four different priors: **V:** A vague prior. **L:** An informative (narrow) prior with a mode at a large weight-decay parameter of $\alpha = 100.0$. **M:** An informative prior with a mode at a medium weight-decay parameter of $\alpha = 1.0$. **S:** An informative prior with a mode at a small weight-decay parameter of $\alpha = 0.01$. As a functional form, we chose a gamma distribution $P(\alpha) \propto \alpha^{\nu/2-1} \exp\left(-\nu\alpha/(2\mu)\right)$, which is conjugate[3] to the Gaussian distribution for the parameters and can easily be sampled from[4]. All simulations were carried out with a one-hidden-layer network with all-to-all connections between adjacent layers and eight tanh units in the hidden layer. The reported results were obtained by averaging over the final 50 % of the configurations obtained from the MCMC simulations, where typical sample sizes were between 400 and 1000 configurations (see Figure 1). For benchmarking we chose a set of two synthetic and three real-world classification problems. A description of these data sets and further references can be found in [3]. Different from [3], we added two additional irrelevant inputs, drawn from the uniform distribution over the unit interval, to the data sets 'Ripley' and 'Tremor'. The input variables of all data sets were normalised to zero mean and unit variance. A brief overview of the characteristics of the data is given in Table 1.

| Problem | Characterisation | | Number of inputs | | Classes | Training exemplars |
|---|---|---|---|---|---|---|
| | synthetic | real | total | irrelevant | | |
| Neal | • | | 4 | 2 | 3 | 400 |
| Ripley | • | | 4 | 2 | 2 | 250 |
| Tremor | | • | 4 | 2 | 2 | 179 |
| Ionos | | • | 34 | – | 2 | 200 |
| Vowel | | • | 10 | – | 11 | 528 |

Table 1: Overview of the classification problems used in this study.

[3] This means that the posterior distribution is of the same functional form as the prior.

[4] The exact setting of the hyper-hyperparameters was as follows. V: $\mu = 100m, \nu = 0.1$, L: $\mu = 100.0m, \nu = 1.0$, M: $\mu = 1.0m, \nu = 1.0$, S: $\mu = 0.01m, \nu = 1.0$. Hidden-to-input weights: $\mu = 400H, \nu = 0.5$ (where $H$ denotes the number of hidden units; see [2] for the reason of this scaling.) Biases: $\mu = 400, \nu = 0.5$.

| | No ARD | | | | ARD | | | |
|---|---|---|---|---|---|---|---|---|
| | V | L | M | S | V | L | M | S |
| Neal | 14.3 | 14.0 | 14.0 | – | 13.7 | 13.8 | 14.8 | – |
| Ripley | 10.7 | 10.9 | 10.2 | 9.4 | 8.9 | 9.6 | 8.8 | 9.1 |
| Tremor | 15.7 | 15.7 | 16.3 | 15.7 | 15.2 | 16.3 | 16.3 | 16.3 |
| Ionos | 1.3 | 1.3 | 1.3 | 1.3 | 10.0 | 6.7 | 6.7 | 6.7 |
| Ionos (PCA) | 1.3 | 1.3 | 1.3 | 1.3 | 4.0 | 4.7 | 3.3 | 2.7 |
| Vowel | 47.9 | 48.7 | 47.8 | – | 56.1 | 44.4 | 43.5 | – |

Table 2: Dependence of the generalisation performance on the prior. The figures show the percent mis-classification on the test set for a hard classification scheme, which assigns the class label to the mode (over $k$) of $P(y_t^k|\mathbf{x}_t)$.

| | No ARD | | | | ARD | | | |
|---|---|---|---|---|---|---|---|---|
| | V | L | M | S | V | L | M | S |
| Neal | 0.355 | 0.354 | 0.355 | – | 0.347 | 0.348 | 0.351 | – |
| Ripley | 0.270 | 0.276 | 0.250 | 0.239 | 0.226 | 0.238 | 0.227 | 0.245 |
| Tremor | 0.371 | 0.381 | 0.373 | 0.375 | 0.411 | 0.390 | 0.407 | 0.407 |
| Ionos (PCA) | 0.116 | 0.119 | 0.115 | 0.115 | 0.142 | 0.124 | 0.104 | 0.097 |

Table 3: Average negative log likelihood of the target, $-\langle \ln P(\mathbf{y}_t|\mathbf{x}_t)\rangle$ (test set).

# 3 Results

Figure 1, top row, shows that the application of an ARD prior leads to the suppression of the two irrelevant inputs. This is reflected by the improved classification results obtained with ARD for Ripley's synthetic classification problem, both in terms of percent mis-classification, Table 2, and average negative log likelihood of the target, Table 3. Note that the second performance measure is more informative (though not more illustrative) than the first, since it captures information about the predictive distribution, which can be drawn on in connection with a rejection option. However, for Neal's synthetic three-way classification problem, the classification performance is only enhanced slightly, and for the real-world problems 'Tremor' and 'Ionos', the classification performance even gets worse as a result of using an ARD prior for the weights. The failure for the tremor data can be understood from the graphs in the middle row of Figure 1. While ARD is successful in suppressing the contributions of the irrelevant inputs, it also leads to low-frequency oscillations of the average weight size in one of the weight groups pertaining to the relevant inputs, which suggests that much longer sampling times than those applied in the present study are needed. For the 'Ionos' data, the inputs are known to be highly correlated. The ARD prior, on the other hand, factorises into independent components pertaining to the different weight groups, which inherently assumes that the inputs are independent. The prior is therefore only poorly matched

| | Neal | Ripley | Tremor | Ionos | Vowel |
|---|---|---|---|---|---|
| Nearest N'bour | 24.2 | 17.4 | 21.4 | 7.3 | 43.7 |
| Nearest N'bour (*) | 18.5 | 14.4 | 12.9 | – | – |
| Logistic regression | 34.7 | 11.2 | 29.2 | 11.3 | 58.0 |
| Nonbayesian NN (*) | – | 9.4 | 15.5 | > 4.0 | 47.0 - 56.0 |
| Bayes. NN, ARD | 14.1 (.4) | 9.2 (.2) | 16.0 (.3) | 3.7 (.4) | 44.0 (.5) |
| Bayes. NN, no ARD | 14.1 (.1) | 10.3 (.3) | 15.9 (.2) | 1.3 (0) | 48.1 (.3) |

Table 4: Comparison between percent mis-classification obtained with non-bayesian methods and Bayesian networks. Details about the nonbayesian neural networks are given in [3]. Simulations indicated by '(*)' did not add any irrelevant inputs to the data sets 'Neal', 'Ripley' and 'Tremor'.

to the data, and in fact a decorrelation of the inputs by subjecting them to a Karhunen-Loève transformation (principal component analysis: PCA) leads to a significant improvement of the classification results (Table 2 and especially Table 3). For the vowel data, ARD leads to improved results when applied in combination with an informative prior, but to a dramatic deterioration in combination with a vague prior. This can be understood from the fact that for a broad prior, the posterior distribution of the hyperparameters may also be broad , as illustrated by a comparison of the two graphs at the bottom of Figure 1. Consequently, the simulation times required for equilibration and sampling may increase significantly, and the bottom graphs of Figure 1 do indeed suggest that the poor performance obtained with the vague prior is most probably the result of the system not having reached equilibrium in the given simulation time.

For comparing the results with alternative classification techniques, we proceeded as follows. The poor result obtained with the vague ARD prior on the vowel data was discarded, as would be suggested by an inspection of the corresponding MCMC trajectories (Figure 1, bottom). Moreover, we assumed that prior knowledge about the correlation of the explanatory variables for the 'Ionos' data is available, which suggests preprocessing of the data with the Karhunen-Loève transform. The results thus obtained are presented in Table 4. They suggest that, overall, Bayesian sampling of network parameters compares very favourably with the alternative schemes. Note that the performance of the nearest neighbour classifier varies largely over the data sets. Moreover, note that the application of nonbayesian neural networks requires extensive experimentation with respect to the weight-decay parameters (unless the Bayesian evidence scheme, [1], is applied), and that the better alternative results for the 'Tremor' data were obtained without additional irrelevant inputs.

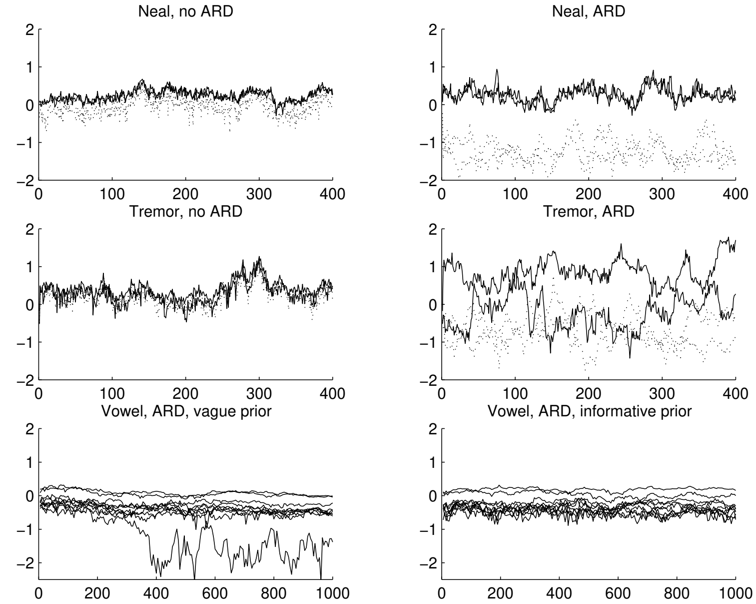


Figure 1. Evolution of the square root of the average square weight magnitude for the different input-to-hidden weight groups during Monte Carlo sampling. Dashed lines correpond to irrelevant inputs. Top left: Neal's problem, no ARD. Top right: Neal's problem, with ARD. Note that the average size of weights on connections exiting the irrelevant inputs are significantly suppressed. Middle left: Tremor data, no ARD. Middle right: Tremor data, with ARD. The irrelevant inputs are suppressed, but one of the relevant inputs gives rise to low-frequency oscillations in the average weight size. Bottom left: Vowel data, ARD, vague prior. Bottom right: Vowel data, ARD, informative prior.

# 4 Conclusion

The present study has applied the method of Bayesian sampling of network parameters to five benchmark classification problems. Of main interest was the dependence of the generalisation performance on the prior distribution of the parameters and hyperparameters. The upshot is that, overall, the method is fairly robust with respect to variations in the prior distribution, and that it tends to outperform alternative classification methods. However, two cautioning findings were made. First, the application of an ARD prior for the input-to-hidden weights is not a panacea for improving the generalisation performance. In fact, in two out of five problems the results thus obtained had actually deteriorated. Second, the application of a vague prior, which at best expresses the prior state of ignorance about what the proper values of the hyperparameters should be, needs to be done with caution, since the resulting posterior distribution may also become very vague, which can considerably decelerate the equilibration process and require long sampling times.

# References


[1] MacKay D.J.C. (1992): Bayesian Interpolation. *Neural Computation* **4**, 415-447.

[2] Neal R.M. (1995): *Bayesian Learning for Neural Networks.* PhD thesis, University of Toronto, Canada.

[3] Penny W.D., Roberts S.J. (1998): Bayesian neural networks for classification: how useful is the evidence? *Neural Networks*, to appear.