ESANN'2000 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2000, D-Facto public., ISBN 2-930307-00-5, pp. 75-80

Confidence Estimation Methods for Neural Networks: A Practical Comparison

G. Papadopoulos, P.J. Edwards, A.F. Murray

Department of Electronics and Electrical Engineering,
University of Edinburgh

Abstract. Feed-forward neural networks (Multi-Layered Perceptrons) are used widely in real-world regression or classification tasks. A reliable and practical measure of prediction "confidence" is essential in real-world tasks. This paper compares three approaches to prediction confidence estimation, using both artificial and real data. The three methods are *maximum likelihood*, *approximate Bayesian* and *bootstrap*. Both noise inherent to the data and model uncertainty are considered.

1. Introduction

Truly reliable neural prediction systems require the prediction to be qualified by a *confidence measure*. This important issue has, surprisingly, received little systematic study and most references to confidence measures take an *ad hoc* approach. This paper offers a systematic comparison of three commonly-used confidence estimation methods for neural networks and takes a first step towards a better understanding of the practical issues involving prediction uncertainty.

Neural network predictions suffer uncertainty due to (a) inaccuracies in the training data and (b) the limitations of the model. The training set is typically noisy and incomplete (not all possible input-output examples are available). Noise is inherent to all real data and contributes to the total prediction variance as *data noise variance* σ_ν^2. Moreover, the limitations of the model and the training algorithm introduce further uncertainty to the network's prediction. Neural networks are trained using an iterative optimisation algorithm (*e.g.* steepest descent). The resultant weight values often correspond, therefore, to a local rather than the global minimum of the error function. Additionally, as the optimisation algorithm can only "use" the information available in the training set, the solution is likely to be valid only for regions sufficiently represented by the training data [1]. We call this *model uncertainty* and its contribution to the total prediction variance *model uncertainty variance* σ_m^2.

These two uncertainty sources are assumed to be independent and the total prediction variance is given by the sum of their variances: $\sigma_{\text{TOTAL}}^2 = \sigma_\nu^2 + \sigma_m^2$. Confidence estimation must take into account both sources. Some researchers ignore model uncertainty assuming that the regression model is correct (*e.g.* [8]). However, in reality a system can be presented with *novel* inputs that differ from the training data. For this reason, the inclusion of model uncertainty estimation is crucial. Finally, we do not make

ESANN'2000 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2000, D-Facto public., ISBN 2-930307-00-5, pp. 75-80

the oversimplification that the data noise variance, σ_ν^2, is constant for all input data, as this is unlikely in complex real-world problems.

2. Methods for Neural Network prediction uncertainty estimation

In regression tasks, the problem is to estimate an unknown function $f(\mathbf{x})$ given a set of input-target pairs $D = \{\mathbf{x}^n, t^n\}, n = 1, ..., N$. The targets are assumed to be corrupted by additive noise $t^n = f(\mathbf{x}^n) + e^n$. The errors e are modelled as Gaussian i.i.d. with zero mean and variance $\sigma_\nu^2(\mathbf{x})$.

Several approaches to prediction confidence estimation have been reported in the literature. In [8] a method for obtaining an input-dependent σ_ν^2 estimate using maximum likelihood (ML) is presented. The traditional network architecture is extended and a new set of hidden units is used to compute $\sigma_\nu^2(\mathbf{x}; \hat{\mathbf{u}})$, the network estimate for data noise variance. The variance output unit has exponential activation function so that σ_ν^2 can only take positive values. The network is trained by joint minimisation of the total error:

$$C = \frac{1}{2}\sum_{n=1}^{N}\left\{\frac{\left[t^n - y(\mathbf{x}^n;\mathbf{w})\right]^2}{\sigma_\nu^2(\mathbf{x}^n;\mathbf{u})} + \ln\sigma_\nu^2(\mathbf{x}^n;\mathbf{u})\right\} + \frac{\alpha_w}{2}\sum_{i=1}^{W} w_i^2 + \frac{\alpha_u}{2}\sum_{i=1}^{U} u_i^2 \quad (1)$$

where α_w and α_u are the regularisation parameters for weights $\mathbf{w}$ (the regression hidden layer connections) and $\mathbf{u}$ (the variance connections) respectively. The main disadvantage of ML is that, as the estimate of the regression fits some of the data noise, the obtained data noise variance estimate is biased.

The Bayesian approach with Gaussian approximation to the posterior can be used to obtain regression and variance estimates, allowing σ_ν^2 to be a function of the inputs [9]. The exact Bayesian approach requires time-consuming Monte-Carlo integration over weight space. It is therefore inappropriate for multi-dimensional, real-world applications for which computing power and run-time must be kept to a minimum. The network is trained in two phases by minimising $C_r(\mathbf{w})$ and $C_v(\mathbf{u})$ alternately:

$$C_r(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\frac{[t^n - y(\mathbf{x}^n;\mathbf{w})]^2}{\sigma_\nu^2(\mathbf{x}^n;\mathbf{u})} + \frac{\alpha_w}{2}\sum_{i=1}^{W} w_i^2 \quad (2)$$

$$C_v(\mathbf{u}) = \frac{1}{2}\sum_{n=1}^{N}\left\{\frac{[t^n - y(\mathbf{x}^n;\mathbf{w})]^2}{\sigma_\nu^2(\mathbf{x}^n;\mathbf{u})} + \ln\sigma_\nu^2(\mathbf{x}^n;\mathbf{u})\right\} + \frac{1}{2}\ln|\mathbf{H}| + \frac{\alpha_u}{2}\sum_{i=1}^{U} u_i^2 \quad (3)$$

where $\mathbf{H}$ is the exact Hessian of error $C_r(\mathbf{w})$. The term $\frac{1}{2}\ln|\mathbf{H}|$ is due to *marginalization* over weights $\mathbf{w}$. This removes the dependence of the variance weights estimate $\hat{\mathbf{u}}$ on the regression estimate $\hat{y}$ resulting in an unbiased data noise variance estimate.

If either ML or the approximate Bayesian approach is used, σ_m^2 can be estimated using the delta estimator, a variant of the linearization method [2]:

$$\sigma_m^2(\mathbf{x}) = \mathbf{g}^T(\mathbf{x})\mathbf{V}^{-1}\mathbf{g}(\mathbf{x}) \quad (4)$$

where $\mathbf{g}(\mathbf{x})$ is a vector whose k-th element is $\partial\hat{y}(\mathbf{x})/\partial\hat{w}_k$ and $\mathbf{V}$ is the covariance matrix of weights $\mathbf{w}$. The covariance matrix is usually estimated by the exact Hessian matrix

ESANN'2000 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2000, D-Facto public., ISBN 2-930307-00-5, pp. 75-80

$\mathbf{H}$ or the outer product approximation to the Hessian $\tilde{\mathbf{H}}$:

$$\mathbf{H} = \sum_{n=1}^{N} \frac{1}{\sigma_\nu^2(\mathbf{x}^n; \hat{\mathbf{u}})} \frac{\partial^2 E^n}{\partial \mathbf{w}^2} + \alpha_w \mathbf{I} \tag{5}$$

$$\tilde{\mathbf{H}} = \sum_{n=1}^{N} \frac{1}{\sigma_\nu^2(\mathbf{x}^n; \hat{\mathbf{u}})} \mathbf{g}(\mathbf{x}^n)\mathbf{g}^T(\mathbf{x}^n) + \alpha_w \mathbf{I} \tag{6}$$

where $\mathbf{I}^n$ is the unitary matrix and $E^n = 1/2(t^n - \hat{y}^n)^2$ is the least-square error for data point n. All quantities are evaluated at the estimated weight values.

Finally, the bootstrap technique offers an altogether different approach to network training and uncertainty estimation [4]. The bootstrap algorithm is given below.

1. Generate B "bootstrap replicates" D_i^* of the original set using *resampling with replacement*. Remove multiple occurrences of the same pair.
2. For each i train a network of the same architecture on D_i^*. If required, the remaining out-of-sample set $D - D_i^*$ can be used for validation.
3. Obtain the bootstrap committee's regression and σ_m^2 estimates by:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{B} \hat{y}_i(\mathbf{x})/B \tag{7}$$

$$\hat{\sigma}_m^2(\mathbf{x}) = \sum_{i=1}^{B} [\hat{y}_i(\mathbf{x}) - \hat{y}(\mathbf{x})]^2/(B-1) \tag{8}$$

respectively, where $\hat{y}_i(\mathbf{x})$ is the prediction of network i.

After the models are trained, σ_ν^2 can be estimated by training an additional network using the model residuals as targets [6]. The residuals are computed using the less biased, out-of-sample regression estimate $\hat{y}_{\text{unbiased}}$ instead of $\hat{y}$:

$$\hat{y}_{\text{unbiased}}(\mathbf{x}) = \frac{\sum_{i=1}^{B} v_i(\mathbf{x}) \cdot \hat{y}_i(\mathbf{x})}{\sum_{i=1}^{B} v_i(\mathbf{x})} \tag{9}$$

where $v_i(\mathbf{x})$ is one for patterns in the out-of-sample set and zero in all other cases. Moreover, the model uncertainty variance estimate $\hat{\sigma}_m^2$ is subtracted from the model residual so that it does not influence σ_ν^2 estimation. The network is trained using error function (1). As the regression estimate is now known, optimisation is performed over weights $\mathbf{u}$ only.

3. Simulation results

We assess confidence estimation performance by evaluating the *Prediction Interval Coverage Probability (PICP)* (see *e.g.* [7]). PICP is the probability that the target of an input pattern lies within the prediction interval (PI). This probability has a nominal value of 95% for $2\sigma_{\text{TOTAL}}$ PI. For each method the coverage probability (CP) is evaluated by calculating the percentage of test points for which the target lies within the $2\sigma_{\text{TOTAL}}$ PI.

ESANN'2000 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2000, D-Facto public., ISBN 2-930307-00-5, pp. 75-80

An optimal method will yield coverage consistently close to 95%, the nominal value. All the networks were trained using steepest descent with line search and weight decay with the same (constant) regularisation parameter.

The methods were tested on both artificial and real tasks. The artificial tasks are variations of the problem proposed by Freidman in [5]. The input is five-dimensional x_i, $i = 1, \ldots, 5$ and the targets are given by:

$$t = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + e \tag{10}$$

Each x_i is randomly sampled from $(0, 1)$. The errors have Gaussian distribution $N(0, \sigma_\nu(\mathbf{x}))$. The standard deviation is given by:

$$\sigma_\nu(\mathbf{x}) = 2g(x_1 + x_2 - 2x_3 - 5x_4 + 2x_5) + 0.05 \tag{11}$$

where g is the sigmoidal function. Three artificial data sets were constructed to investigate performance under different input data density situations. The three data sets, marked *L*, *H* and *A*, have a data density that is approximately double than elsewhere, in the Low, High and Average data noise variance region respectively. In other words, set *L* for example, contains less training data originating from the input space region of high σ_ν values and more data from the region of low σ_ν values. Of course, in a 5-dimensional input space, points from regions not represented in the training set may have similar standard deviation. This situation may occur in real-world, multi-dimensional problems for which the available data are typically quite sparse. Each training set contained 120 examples and a separate set of 10000 examples was used for testing. The regression and variance hidden layers contained five and one units respectively and the regularisation parameter was set to 0.01.

The real data set is the "paper curl" data set described in [3]. *Curl* is a paper quality parameter whose value can only be measured after manufacture. The curl data set contains 554 examples of which 369 are used for training and the remaining 185 for testing. The input vector is eight-dimensional. Curl prediction was performed in [3] using a committee of networks, under the assumption of constant σ_ν^2. Here we investigate the effect of allowing σ_ν^2 to be a function of the input. The constant variance approach serves as the baseline. The constant σ_ν^2 estimate is computed by $\sigma_\nu^2 = \sum_{n=1}^{N}[t^n - \hat{y}(\mathbf{x}^n)]^2/(N-1)$. After experimentation it was found that one hidden unit is enough to model σ_ν^2 while eight units are used for the regression model. The regularisation parameter was set to 0.01.

To reduce bias in the results, 100 networks were trained and the reported results are the average over 100 committees formed by choosing networks at random from the pool. The committee size was set to 20 networks. This is a reasonable number for real-world applications where training and prediction run-times must be taken under consideration. The prediction of the ML and Bayesian committees is the mean prediction $y = <y_i>$ while the committee total variance is given by $\sigma^2 = <\sigma_i^2> + (<y_i^2> - <y_i>^2)$ where σ_i^2 and y_i are the total variance and prediction of net i respectively.

The results are summarised in fig. 1. The PICP mean and standard deviation over the 100 simulations are shown for each data set and method. The PICP is expressed as difference from the nominal value (95%). The bootstrap and the Bayesian with approximate Hessian techniques appear to perform better for sets *L* and *A*, although the

ESANN'2000 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2000, D-Facto public., ISBN 2-930307-00-5, pp. 75-80

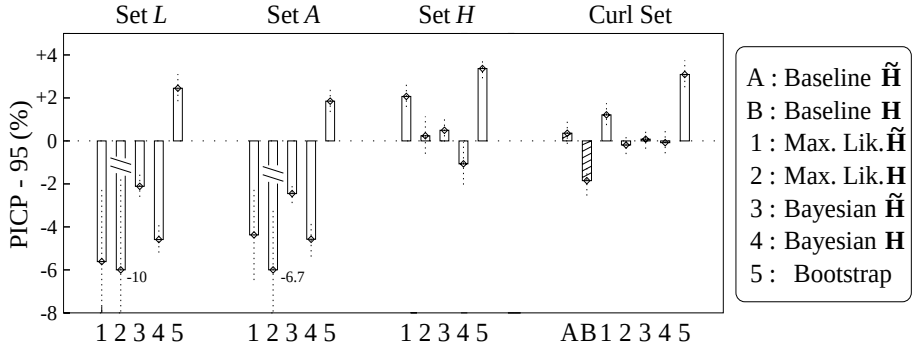

Figure 1: Prediction Interval Coverage (mean and standard deviation) for the simulated and real problems, expressed as difference from the nominal value (95%).

bootstrap overestimates coverage consistently for all sets. A larger bootstrap committee is likely to produce better results, but we chose to restrict committee size to a realistic minimum. ML and Bayes perform best for sets *H* and the curl set.

The approximate Bayesian approach gives consistently better coverage than ML especially for sets *L* and *A*. In these sets, data density is low in the high noise variance region and the bias in the ML prediction becomes more apparent (see [9] for a more detailed explanation). ML and Bayes perform similarly for sets *H* and the curl set. Set *H* has larger input data density in the high variance region, thus the bias in the ML σ_ν^2 estimate is reduced and ML performs almost as well as the Bayesian method.

Substituting the exact Hessian with the approximate in the delta estimator results in a small increase of the coverage. There is no significant deterioration when the approximate Hessian is used. This result agrees with previous findings using non-linear models [2].

For the real "curl" data set, using constant or input-dependent σ_ν^2 does not appear to have a great impact on the coverage. However, using ML and Bayes committees results in a 2 and 2.5% improvement in the test error over the constant σ_ν^2 committee. As the test set for the paper curl task contains only 185 examples, it is possible that the results are biased. However, there is strong indication that the flexible σ_ν^2 models represent the data better than the constant σ_ν^2 model.

4. Discussion and further work

Three popular and trusted methods for confidence estimation in neural networks have been tested using three artificial and one real data set. This selective set of exemplars are all representative of a very common category of real world applications, namely regression using sparse, multi-dimensional and noisy data. Unlike previous studies, both data noise and model uncertainty have been considered. Moreover, data noise variance has been treated as a function of the inputs and it has been shown that this leads to better results for the paper curl estimation task. The training time required

for the approximate Bayesian approach is significantly larger than the times for either ML or the bootstrap technique. Since the curl estimator has to be retrained regularly to ensure that the model represents the current conditions in the paper plant, it may be more realistic to use one of the latter (faster) approaches. In terms of test error, the bootstrap technique is better than ML by an average of 2.5%. It may be preferable to use a bootstrap committee for the curl estimator, even though the obtained PICP is slightly larger than the nominal value.

To our knowledge, the approximate Bayesian approach with input-dependent σ_ν^2 has not been tested previously using neural networks. The results indicate that the Gaussian approximation works satisfactorily at least for the problems presented here. The disadvantage of this method is the long training times required due to evaluation of the Hessian matrix. This method yields unbiased σ_ν^2 estimates and it outperforms ML especially when the training set contains regions of high noise-low input data density. However, ML still performs satisfactorily and is a possible candidate for applications where training times are crucial.

The methods have been compared using the prediction interval coverage probability. The PICP is only sensitive to the average size of the interval and in particular, whether or not the interval includes the target. However, from a practical point of view the ideal confidence measure should associate high confidence with accurate predictions and low confidence with inaccurate ones. The PICP can not be used to assess this since it does not take into account the local quality of the total variance estimate. The development of such a method is the subject of ongoing work.

References

[1] C. M. Bishop. Novelty detection and neural network validation. In *IEE Proceedings in Vision, Image and Signal Processing*, volume 141, pages 217–222, 1994.

[2] J. R. Donaldson and R. B. Schnabel. Computational experience with confidence regions and confidence intervals for nonlinear least squares. *Technometrics*, 29(1):67–82, 1987.

[3] P.J. Edwards, A.F. Murray, G. Papadopoulos, A.R. Wallace, J. Barnard, and G. Smith. The application of neural networks to the papermaking industry. *IEEE Transactions on neural networks*, 10(6):1456–1464, November 1999.

[4] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, London, UK, 1993.

[5] J. Freidman. Multivariate adaptive regression splines. *Annals of Statistics*, 19:1–141, 1991.

[6] Tom Heskes. Practical confidence and prediction intervals. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 176–182. The MIT Press, 1997.

[7] J. T. Gene Hwang and A. Adam Ding. Prediction intervals for artificial neural networks. *Journal of the American Statistical Association*, 92(438):748–757, June 1997.

[8] David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of the IJCNN'94*, pages 55–60. IEEE, 1994.

[9] C.S. Qazaz. *Bayesian Error Bars for Regression.* PhD thesis, Aston University, 1996.