ELSEVIER

Neurocomputing 48 (2002) 175–183

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Bootstrap for neural model selection

Riadh Kallel[a], Marie Cottrell[a,*], Vincent Vigneron[a,b]

[a]*MATISSE-SAMOS, Universite Paris 1, 90, rue de Tolbiac, 75634 Paris cedex 13, France*
[b]*CEMIF, 40 rue du Pelvoux, 91020 Evry Courcouronnes, France*

Received 9 November 2000; accepted 6 June 2001

Abstract

Bootstrap techniques (also called *resampling computation techniques*) have introduced new advances in modeling and model evaluation (Principles of Neural Model Identification, Selection and Adequacy, Springer, New York, 1999). Using resampling methods to construct a series of new samples which are based on the original data set, allows to estimate the stability of the parameters. Properties such as convergence and asymptotic normality can be checked for any particular observed data set. In most cases, the statistics computed on the generated data sets give a good idea of the confidence regions of the estimates. In this paper, we debate on the contribution of such methods for model selection, in the case of feedforward neural networks. The method is described and compared with the leave-one-out resampling method. The effectiveness of the bootstrap method, versus the leave-one-out method, is checked through a number of examples. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Bootstrap; Model selection; Multilayer perceptron

1. Multilayer perceptrons (MLP)

Suppose a set of n independent observations of a continuous variable y that we have to explain from a set of p explanatory variables $(x_1, x_2, \ldots, x_p)$. We want to use the non-linear models called *multilayer perceptrons*. These models are nowadays commonly used for non-linear regression, forecasting, pattern recognition, and are particular examples of artificial neural networks. In such a network, units are

* Corresponding author. Fax: +33-1-40-77-19-22.
E-mail addresses: kallel@univ-paris1.fr (R. Kallel), cottrell@univ-paris1.fr (M. Cottrell), vvigne@iup.univ-evry.fr (V. Vigneron).

organized in successive layers with links connecting one layer to the following one. See Cheng and Titterington [2] or Hertz et al. [7] for details.

We consider in the following a multilayer perceptron (MLP) with p inputs, one hidden layer with H hidden units and one output layer. The model can be analytically expressed in the following form: the output y is given by

$$y = \phi_0\left(w_0 + \sum_{h=1}^{H} w_h \phi\left(b_h + \sum_{j=1}^{p} w_{jh} x_j\right)\right) + \varepsilon, \tag{1}$$

where ε is the residual term, with zero mean, variance σ^2 (with normal distribution or not), y is a continuous variable, ϕ_0 is the identity output function, ϕ is (in most cases) the sigmoid

$$\phi(x) = \frac{1}{1+\exp(-x)}.$$

Let $\boldsymbol{\theta} = (w_0, w_1, \ldots, w_H, w_{11}, \ldots, w_{pH})$ be the parameter vector of the network and let $y(\boldsymbol{x};\boldsymbol{\theta})$ be the computed value for an input $\boldsymbol{x} = (x_1, \ldots, x_p)$ and a parameter vector $\boldsymbol{\theta}$. There are $H(p+1)+H+1$ parameters to estimate.

Classically, if there are numerous data, the first step consists in the division of the supplied data into two sets: a *training set* and a *test set*. The so-called training set

$$\{(\boldsymbol{x}_1; y_1), \ldots, (\boldsymbol{x}_m; y_m); (1 \leqslant i \leqslant m; m < n)\}$$

is used to estimate the weights of the model by minimizing an error function

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - y(\boldsymbol{x}_i;\boldsymbol{\theta}))^2$$

using optimization techniques such as gradient descent, conjugate gradient or quasi-Newton methods.

The resulting least-squares estimator of $\boldsymbol{\theta}$ is denoted by $\hat{\boldsymbol{\theta}}$, and the resulting lack of fit for the training set is the *learning error*

$$MSE_{\mathrm{a}} = \frac{1}{m}\sum_{i=1}^{m}(y_i - y(\boldsymbol{x}_i;\hat{\boldsymbol{\theta}}))^2. \tag{2}$$

The training set is used to derive the parameters of the model and the resulting model is tested on the test set. A good regression method would generalize well on examples that have not been seen before, by learning the underlying function without the associated noise. The *test error* can be defined by

$$MSE_{\mathrm{t}} = \frac{1}{n-m}\sum_{i=m+1}^{n}(y_i - y(\boldsymbol{x}_i;\hat{\boldsymbol{\theta}}))^2. \tag{3}$$

Most optimization techniques (that are variants of gradient methods) provide local minima of the error function and not a global one. Practically, different learning conditions (initialization of weights, learning adaptation parameter, sequential order in the sample presentation, etc.) give different solutions that it is difficult to compare. It is not easy to know if a minimum is reached, because the decrease of the error function is slow, an over-learning phenomenon can occur, etc. For these reasons, numerous stopping and validation techniques are proposed, see for example Borowiak [1], or Hertz et al. [7].

For multilayer perceptrons, the choice of a model is equivalent to the choice of the *architecture* of the network. If one has to select a model among a lot of them, an exhaustive (but not realistic) method would consist in exploring the whole set of possible models, and in testing all these models on the given problem. The estimation of the performances is then a very crucial point, all the more so since many factors intervene to complicate this evaluation. It is necessary to be certain that the convergence has occurred, to have at disposal a good-quality criterion which allows to decide what is the *best model*. In fact it is impossible to try all the possible models, so bootstrap method can be very useful.

2. Bootstrap for parameter estimation

Bootstrap techniques were introduced by Efron [3,4] and are simulation techniques based on the empirical distribution of the observed sample. Let $\boldsymbol{x}=(x_1,\ldots,x_n)$ an n-sample, with an unknown distribution function $\mathscr{F}$, depending on an unknown real parameter $\boldsymbol{\theta}$. The problem consists in estimating this parameter $\boldsymbol{\theta}$ by a statistic $\hat{\boldsymbol{\theta}}=s(\boldsymbol{x})$ from the sample $\boldsymbol{x}$ and in evaluating the estimate accuracy, although the distribution $\mathscr{F}$ is unknown. In order to evaluate this accuracy, B samples are built from the initial sample $\boldsymbol{x}$, by re-sampling. These samples are called *bootstrapped samples* [9] and denoted by x^{*b}.

A bootstrapped sample $\boldsymbol{x}^{*b}=(x_1^{*b},\ldots,x_n^{*b})$ is built by a random drawing (with repetitions) in the initial sample $\boldsymbol{x}$:

$$P_{\mathrm{U}}(x_i^{*b}=x_j)=\frac{1}{n} \quad i,j=(1,\ldots,n),$$

where P_{U} is the uniform distribution on the original data set $\boldsymbol{x}=(x_1,\ldots,x_n)$. The distribution function of a bootstrapped sample $\boldsymbol{x}^{*b}$ is $\hat{\mathscr{F}}$, i.e. the empirical distribution of $\boldsymbol{x}$. A bootstrap replicate of the estimator $\hat{\boldsymbol{\theta}}=s(\boldsymbol{x})$ will be $\hat{\boldsymbol{\theta}}^{*b}=s(\boldsymbol{x}^{*b})$. For example, for the mean of the sample $\boldsymbol{x}$, the estimator is $s(\boldsymbol{x})=(1/n)\sum_{i=1}^{n}x_i$, and a bootstrap replicate will be $s(\boldsymbol{x}^{*b})=(1/n)\sum_{i=1}^{n}x_i^{*b}$.

Then, the bootstrap estimate of the standard deviation of $\hat{\boldsymbol{\theta}}$ denoted by $\hat{\sigma}_{\mathrm{boot}}(\hat{\boldsymbol{\theta}})$ is given by

$$\hat{\sigma}_{\mathrm{boot}}(\hat{\boldsymbol{\theta}}^*)=\left[\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\boldsymbol{\theta}}^{*b}-\hat{\boldsymbol{\theta}}^*(.))^2\right]^{1/2}$$

and

$$\hat{\boldsymbol{\theta}}^*(\cdot)=\frac{1}{B}\sum_{b=1}^{B}\hat{\boldsymbol{\theta}}^{*b}.$$

It is computed by replacing the unknown distribution function $\mathscr{F}$ with the empirical distribution $\hat{\mathscr{F}}$. In conjunction with these re-sampling procedures, hypothesis tests and confidence regions for statistics of interest can be constructed.

In the following, the method we propose as a tool to select a MLP model is similar to the bootstrap method, since it relies on re-sampling techniques, but it is non-parametric.

3. Bootstrap applied to selection model for MLPs

Let $\mathscr{B}_0$ be a data set of size n,

$$\mathscr{B}_0 = \{(\boldsymbol{x}_1; y_1), \ldots, (\boldsymbol{x}_n; y_n); (1 \leqslant i \leqslant n)\},$$

where $\boldsymbol{x}_i$ is the ith value of a p-vector of explanatory variables and y_i is the response to $\boldsymbol{x}_i$. From the original data set $\mathscr{B}_0$ (called *initial base*), one generates B bootstrapped bases $\mathscr{B}_b^*$, $1 \leqslant b \leqslant B$, (i.e. B uniform drawings of n data points in $\mathscr{B}_0$ with repetitions). For any generated data set $\mathscr{B}_b^*$, an estimator of the MLP parameter vector $\boldsymbol{\theta}$, denoted by $\hat{\boldsymbol{\theta}}^{*b}$, is found by application of the backpropagation algorithm [8] for example, but any minimization algorithm can be used. So the bootstrap procedure provides B replications $\hat{\boldsymbol{\theta}}^{*b}$ for model (1).

Then we use $\mathscr{B}_0$ as a test base, and evaluate for each $b = 1, \ldots, B$ and each $i = 1, \ldots, n$ the residual estimate:

$$\varepsilon_{\text{test},i}^{*b} = y_i - y(\boldsymbol{x}_i; \hat{\boldsymbol{\theta}}^{*b}).$$

The study of the histograms of these estimated residuals allows to evaluate the distribution of the error term ε, to control its whiteness, etc. For each bootstrapped sample $\mathscr{B}_b^*$, $b = 1, \ldots, B$, (that is for each $\hat{\boldsymbol{\theta}}^{*b}$), the sum of squares of the residuals on the test base $\mathscr{B}_0$ is computed

$$TSSE(b) = \sum_{i=1}^{n} (\varepsilon_{\text{test},i}^{*b})^2$$

as well as the mean of the squares of the residuals on the test base $\mathscr{B}_0$:

$$TMSE(b) = \frac{1}{n} \sum_{i=1}^{n} (\varepsilon_{\text{test},i}^{*b})^2.$$

So, we get a vector $TMSE$ whose mean value is

$$\mu_{\text{boot}} = \frac{1}{B} \sum_{b=1}^{B} TMSE(b) \tag{4}$$

and standard deviation is

$$\sigma_{\text{boot}} = \left[\frac{1}{B-1} \sum_{b=1}^{B} (TMSE(b) - \mu_{\text{boot}})^2 \right]^{1/2}. \tag{5}$$

Table 1
Re-sampling algorithm (bootstrap procedure) used to compute μ_{boot} and σ_{boot} (typically $20 \leqslant B \leqslant 200$)

1. To generate B samples of size n by random drawings with repetitions in the initial base $\{\mathscr{B}_0\} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$. Let us denote by $\{\mathscr{B}_b^*\} = \{(\boldsymbol{x}_1^{*b}, y_1^{*b}), \ldots, (\boldsymbol{x}_n^{*b}, y_n^{*b})\}$ the bth bootstrapped sample, $b = 1, \ldots, B$.

2. For each bootstrapped sample, $b = 1, \ldots, B$, to estimate $\boldsymbol{\theta}$ by minimizing $\sum_{i=1}^{n} [y_i^{*b} - y(\boldsymbol{x}_i^{*b}; \boldsymbol{\theta})]^2$, we get $\hat{\boldsymbol{\theta}}^{*b}$.

3. The bootstrap standard deviation is given by

$$\sigma_{\text{boot}} = \left[\frac{1}{B-1} \sum_{b=1}^{B} (TMSE(b) - \mu_{\text{boot}})^2\right]^{1/2},$$

where

$$\mu_{\text{boot}} = \frac{1}{B} \sum_{b=1}^{B} TMSE(b).$$

These two values measure the residual variance of the model, estimated from the bootstrapped samples, and the stability of the parameter vector estimations. So this technique allows to evaluate a model from only one sample (without splitting it into a training base and a test base, which decreases the number of data used for the estimation).

To choose between several architectures $M_1, M_2, \ldots$, these computations are repeated for each of them, and the best one will be this one that has the best compromise (the ideal would be to simultaneously minimize μ_{boot} and σ_{boot}). The approach is summarized in Table 1.

Two main disadvantages must be outlined

- The *computer simulation time*: If n or p is high, computation time can be very long even with second-order optimization techniques as BFGS, but it still remains less than computing time for empirical exploration.
- The *repetition of extremal data*: The risk exists to select a re-sampling data set for which iterative methods will converge with difficulty. But ignoring these repetitions could introduce a bias.

Many other re-sampling procedures have been proposed in the statistical literature: cross-validation, Jackkniffe, leave-one-out, etc. See Hamamoto [6] and Borowiak [1] for details.

4. Examples

We wish to illustrate the bootstrap method on two examples with simulated data. The third example is an application of our method on a real data set. For each example, we built $B = 50$ bootstrapped samples and three models with different architectures are compared, in order to choose the best one.

Table 2
Summary table: comparison results of bootstrap method and leave-one-out method

Model		Bootstrap		Leave-one-out[a]	
		μ_{boot}	σ_{boot}	μ_{loo}	σ_{loo}
Exp 1	M_1	**3.9525**	**0.0155**	4.76268	6.49886
	M_2	3.9020	0.5985	4.81903	6.54536
	M_3	3.9475	0.4259	4.73803	6.54557
Exp 2	M_2	**0.04277**	**0.00019**	0.04999	0.06807
	M_4	0.04271	0.00029	0.05303	0.07553
	M_6	0.04277	0.00028	0.04895	0.06772
Exp 3	M_{30}	**0.0473**	**0.0052**	0.03961	0.05347
	M_{35}	0.0599	0.0069	0.05132	0.07873
	M_{40}	0.0492	0.0049	0.04763	0.08161

[a]We use 50 data bases replications for every training.

A comparison is made with the leave-one-out method, which is also based on data bases replication, but in a different way. We use an uniform distribution on the original data to leave one observation. Hence, we train the MLP on $B=50$ data bases replications with $n-1$ observations, and we compute the values $TMSE(b)$ using the observation that we left as a test base. We use the same B for both methods to be able to compare them using the same number of replications. We get a vector $TMSE$ and compute its mean μ_{loo} and its standard deviation σ_{loo}, as before.

4.1. Example 1. linear model

Consider the problem of fitting a linear model:

$$y=\theta_0+\theta_1x_1+\theta_2x_2+\cdots+\theta_px_p+\varepsilon.$$

We simulate a data set $\mathscr{B}_0=(x_1^{(i)},x_2^{(i)},y_i),\ i=1,\ldots,500$ by putting

$$x_1^{(i)}=i,\quad x_2^{(i)}=i^{1/2},\quad y_i=2+0.7x_1^{(i)}+0.5x_2^{(i)}+\varepsilon_i,$$

where ε_i is a random variable which possesses the distribution $\mathcal{N}(0,4)$, (4 is the variance). We consider three models:

- Model M_1: $p=2,\ y=\theta_0+\theta_1x_1+\theta_2x_2+\varepsilon$: true model.
- Model M_2: $p=1,\ y=\theta_0+\theta_1x_1+\varepsilon$.
- Model M_3: $p=3,\ y=\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_3+\varepsilon$, with $x_3^{(i)}=i^{3/2}$ and $\theta_3=1$.

We compute $\mu_{\text{boot}}(M_i)$, $\mu_{\text{loo}}(M_i)$ (Eq. (4)), $\sigma_{\text{boot}}(M_i)$ and $\sigma_{\text{loo}}(M_i)$ (Eq. (5)) for each model, the results are in Table 2. With the bootstrap method, we see that the best model is the model M_1 i.e. the true model. With the leave-one-out method we cannot conclude, because there is no significant differences between the three values of μ_{loo} and of σ_{loo}. Note that the mean μ_{loo} is over-estimated and that σ_{loo} has an order 10 times greater than σ_{boot}.

4.2. Example 2. non-linear modeling with simulated data

We use Eq. (1) with sigmoid transfert function ϕ to simulate a data set

$$\mathscr{B}_0 = (x_1^{(i)}, x_2^{(i)}, y_i), \quad i = 1, \ldots, 500$$

by computing y_i as a noisy output of a multilayer perceptron, defined by $p = 2$ input variables, $x_1 \sim \mathcal{N}(0.2, 4)$, $x_2 \sim \mathcal{N}(-0.1, 0.25)$, there are one hidden layer and 4 neurones on the hidden layer, $\boldsymbol{\theta} = (0.5, -0.1, 0.2, 0.5, -0.4, 0.2, 0.1, 3, 0.3, 2, 0.5, 0.1, 0.2, 2, 0.2, 3, 0.1)$, as defined in Section 1, ε possesses a distribution $\mathcal{N}(0, 0.04)$.

We consider three models:

- Model M_2: two inputs, one hidden layer with 2 hidden neurons.
- Model M_4: two inputs, one hidden layer with 4 hidden neurons: true model.
- Model M_6: two inputs, one hidden layer with 6 hidden neurons.

We compute $\mu_{\mathrm{boot}}(M_i)$, $\mu_{\mathrm{loo}}(M_i)$ (Eq. (4)), $\sigma_{\mathrm{boot}}(M_i)$ and $\sigma_{\mathrm{loo}}(M_i)$ (Eq. (5)) for each model. Table 2 shows the results. Bootstrap method shows that the best model is the model M_2. It is not the true model, but it is the best. It is not so surprising since the multilayer perceptrons are always over-parametrized, and that there is no unicity of the multilayer perceptron function which can model a given function. With the leave-one-out method, we cannot conclude, because it eliminates the true model, and do not separate the first and the third models.

4.3. Example 3. non-linear model with real data

In this section, we study a real data set to set the efficiency of the model selection method that we propose.

The power peak control in the core of nuclear reactors is explored. The problem has already been studied in the past, namely by Gaudier [5], who constructed a neuronal model with 22 input variables, 2 hidden layers, (the first one with 26 neurons, the other with 40 neurons). The model accounts for physical localization of uranium bars and diffusion processes, and was set to reproduce the classical calculus code, while winning in terms of computing time.

- Model M_{40}: 22 inputs, two hidden layers with, respectively, 26 and 40 hidden neurons.
- Model M_{35}: 22 inputs, two hidden layers with, respectively, 26 and 35 hidden neurons.
- Model M_{30}: 22 inputs, two hidden layers with, respectively, 26 and 30 hidden neurons.

For each model, we compute $\mu_{\text{boot}}(M_i)$, $\mu_{\text{loo}}(M_i)$ (Eq. (4)), $\sigma_{\text{boot}}(M_i)$ and $\sigma_{\text{loo}}(M_i)$ (Eq. (5)).

The bootstrap method (Table 2) shows that the model M_{30} seems to be the best, (its residual variance is the smallest for a similar value of μ_{boot}). The leave-one-out method confirms our conclusion in this case. But $\sigma_{\text{boot}} \ll \sigma_{\text{loo}}$ for each model, which is important to ensure the stability of the model. In that case, it would be necessary to study other architectures different from the three that we have considered.

We remark that in all the cases, $\sigma_{\text{boot}} \ll \sigma_{\text{loo}}$, so the estimation of the variance of the model is much more precise with the bootstrap method than with the leave-one-out method.

5. Conclusion

These examples indicate that our technique is better then the leave-one-out method. The bootstrap method can be used for a great variety of situations. We have applied it for many other cases, and the results seem to be very interesting to help for model selection.

References

[1] D.S. Borowiak, Model Discrimination for Nonlinear Regression Models, Marcel Dekker, New York, 1990.

[2] B. Cheng, D.M. Titterington, Neural networks: a review from a statistical perspective, Statist. Sci. 9 (1994) 2–54.

[3] B. Efron, The convex hull of a random set of points, Biometrika 52 (1979) 331–342.

[4] B. Efron, R. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, London, 1993.

[5] F. Gaudier, Optimisation et réseaux de neurones pour le repositionnement des barres de combustible nucléaire, Thèse de doctorat de l'université Paris VI, ENS Cachan, 1998.

[6] Y. Hamamoto, S. Uchimura, S. Tomita, A bootstrap technique for nearest neighbor classifier design, IEEE Trans. PAMI 19 (1997) 73–79.

[7] J. Hertz, A. Krogh, R. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley, Redwood City, CA, 1991.

[8] D.E. Rumelhart, G.E. Hinton, R.J. Wiliams, Learning Internal Representations by Error Propagation, Parallel Distributed Processing, Vol. 8, MIT Press, Cambridge.

[9] A. Zapranis, A.-P. Refenes, Principles of Neural Model Identification, Selection and Adequacy, Springer, London, 1999.

Marie Cottrell was born in Béthune (France) in 1943. She was a student at the Ecole Normale Supérieure de Jeunes Filles, and received the Agrégation de Mathématiques in 1964, and the Thèse d'Etat (Modélisation de réseaux de neurones par des chaînes de Markov et autres applications) in 1988. Since 1964, she was a High School Teacher and joined the University of Paris-Sud (Orsay) in 1968 as an Assistant Professor. From 1970 to 1973, she was a Professor at the University of Habana, Cuba. She is currently a full Professor at the University Paris 1, Panthéon-Sorbonne. She is in charge of the SAMOS-MATISSE (Statistique Appliquée et Modélisation Stochastique), Research Center of the University Paris 1. Her research interests include stochastic algorithms, biomathematics, data analysis, statistics. Since 1985, her main work deals with artificial and biological neural networks and the relations with data analysis.

Vincent Vigneron is 31 years old. He is a assistant professor of Computer Sciences at the University Evry Val d'Essonne (France). Since 1994, he is working on signal processing algorithms, neural and bayesian networks applied to vision and medical signals.

Riadh Kallel was born in Sfax (Tunisia) in 1971. He is Ph.D. student at the University Paris 1, Panthéon-Sorbonne. He is preparing his Ph.D. diploma in SAMOS-MATISSE laboratory. He is working on bootstrap method for neural networks selection model and data fusion.