Computers and Mathematics with Applications 62 (2011) 4576–4581

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

Balanced bootstrap resampling method for neural model selection

Wen-Liang Hung[a,*], E. Stanley Lee[b], Shun-Chin Chuang[c]

[a] *Department of Applied Mathematics, National Hsinchu University of Education, Hsin-Chu, 30013, Taiwan*
[b] *Department of Industrial and Manufacturing Systems Engineering, Kansas State University, KS 66506, USA*
[c] *Headquarters Administration of Cultural Heritage, Council for Cultural Affairs, Tai-Chung, 40247, Taiwan*

ARTICLE INFO

Article history:
Received 20 July 2011
Accepted 13 October 2011

Keywords:
Backpropagation multilayer perceptron
Bootstrap
Balanced resampling
Model selection

ABSTRACT

Uniform resampling is the easiest to apply and is a general recipe for all problems, but it may require a large replication size B. To save computational effort in uniform resampling, balanced bootstrap resampling is proposed to change the bootstrap resampling plan. This resampling plan is effective for approximating the center of the bootstrap distribution. Therefore, this paper applies it to neural model selection. Numerical experiments indicate that it is possible to considerably reduce the replication size B. Moreover, the efficiency of balanced bootstrap resampling is also discussed in this paper.

1. Introduction

It is well-known that backpropagation multilayer perceptrons (MLPs) are often used to describe neural network structures. Their applications are in a wide range of areas, such as pattern recognition, signal processing, data compression, and automatic control. A backpropagation MLP is an adaptive network whose nodes (or neurons) perform the same function on incoming signals; and this node function is usually a composite of the weight sum and a differentiable nonlinear activation function. Three activation functions are the most commonly used in backpropagation MLPs: logistic, hyperbolic tangent and identity functions. In this paper, for simplicity, we assume that the logistic function is used in backpropagation MLP.

Let $(x_1, x_2, \dots, x_p)$ be a a set of p explanatory variables to explain a continuous variable y. An MLP with p inputs, one hidden layer with H hidden units and one output layer is used to model these data as follows:

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

where ϵ is the residual term, with zero mean, variance σ^2 (with normal distribution or not), and ϕ is the logistic function. Let $y(x; \theta)$ be the computed value for an input $x = (x_1, \dots, x_p)$ and a parameter $\theta = (w_0, w_1, \dots, w_H, w_{11}, \dots, w_{pH})$. For MLPs, the choice of an appropriate mode is an important problem. Kallel et al. [1] applied the bootstrap method [2] for neural model selection, since it is more effective than the leave-one-out method. To reduce the bootstrap method's computational load, Lendasse et al. [3] proposed a fast bootstrap (FB) methodology for regression model selection. According to their numerical experiments on multi-layer perceptrons, radial-basis function networks and least-square support vector machines, the FB can reduce the bootstrap replication size. However, they do not discuss the asymptotic efficiency of FB relative uniform resampling (i.e., the original bootstrap method). Recently, Chuang et al. [4] proposed a weighted bootstrap to selection model for MLPs. Based on their numerical results, the performance of the weighted bootstrap is better than the bootstrap method. However, there is a problem in Chuang et al. [4]: How to select the optimal weight in the weighted

[*] Corresponding author. Tel.: +886 3 5213132; fax: +886 3 5611128.
E-mail address: wlhung@mail.nhcue.edu.tw (W.-L. Hung).

0898-1221/$ – see front matter Crown Copyright © 2011 Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2011.10.039

bootstrap? To solve this problem, we apply the balanced bootstrap resampling, which is not necessary to select the weight, to selection model for MLPs.

The remaining of this paper is organized as follows. In Section 2 we apply an efficient bootstrap-balanced resampling-to selection model for MLPs. The balanced resampling scheme ensures that each data appears the same number of times in all the resamples. Furthermore, the asymptotic efficiency of balanced resampling relative uniform resampling is considered in Section 3. Some numerical results are given in Section 4 and conclusions are given in the last section.

2. Balanced resampling in selection model for MLPS

While the bootstrap is based on the principle of substitution and mimicking sampling behavior, its application is carried out with data resampling, the Monte Carlo method is required for computing the bootstrap estimators. The robustness property of the bootstrap provides a strong motivation for the use of this data-resampling method. Although the simple Monte Carlo method is easy to use, the cost and time of the computations may be a burden. Therefore, it is necessary to develop some bootstrap computational methods that are more efficient than the simple Monte Carlo method so that we can reduce the number of computations.

By viewing the bootstrap sampling as a Monte Carlo estimate, Davison et al. [5] and Davison [6] introduced the technique of balanced resampling to reduce the computation effort. The balanced resampling plan is similar to a balanced randomized block design. An algorithm for performing balanced resampling has been described by Gleason [7]. Hall [8] discussed the asymptotic efficiency of balanced resampling relative to uniform resampling, in problems of distribution function and quantile estimations. It turns out that the balanced resampling is considerably more efficient compared to uniform resampling in estimating the central probabilities.

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

$$\mathcal{B}_0 = \{(x_1; y_1), \dots, (x_n; y_n)\},$$

where x_i is the ith value of a p-vector of explanatory variables and y_i is the response to x_i. First, we use the data set $\mathcal{B}_0$ to estimate the parameter θ of the model (1) and the resulting least-squares estimator of θ is denoted by $\hat{\theta}$. Thus, the residual for the ith observation is denoted by e_i and is defined as follows:

$$e_i = y_i - y(x_i; \hat{\theta}),$$

and the sum of squares, denoted by SSE, is:

$$\text{SSE} = \sum_{i=1}^{n} \Big(y_i - y(x_i; \hat{\theta})\Big)^2 = \sum_{i=1}^{n} e_i^2,$$

where SSE stands for the *error sum of squares* or *residual sum of squares*. Hence, the appropriate mean square error, denoted by MSE, is:

$$\text{MSE} = \frac{\text{SSE}}{n} = \frac{\sum_{i=1}^{n} e_i^2}{n},$$

where MSE stands for *error mean square* or *residual mean square*.

The standard deviation σ of the residual term ϵ in model (1) needs to be estimated for indicating the variability of the probability distributions of y. In this study, we use $\sqrt{\text{MSE}}$ to estimate σ. For convenient, $\sqrt{\text{MSE}}$ is denoted by $\hat{\sigma}$. It is natural to pose the basic question: "How accurate is $\hat{\sigma}$?". If we know the distribution of $\sqrt{n}(\hat{\sigma} - \sigma)$, that is,

$$F(x) = P\left\{\sqrt{n}(\hat{\sigma} - \sigma) \le x\right\},$$

then we automatically solve this problem. Following the recipe for obtaining bootstrap variance estimators, we can immediately obtain the bootstrap estimator of $F(x)$. In general, the bootstrap estimator of $F(x)$ does not have an explicit form. However, we may use a Monte Carlo method to approximate the bootstrap estimator of $F(x)$.

For the bootstrap approximation of the distribution of $\sqrt{n}(\hat{\sigma} - \sigma)$, based on $\mathcal{B}_0$. Instead of drawing B random samples (with replacement) uniformly from the original data set, $\mathcal{B}_0 = \{(x_1; y_1), \dots, (x_n; y_n)\}$, we apply the balanced resampling technique to obtain the bootstrap samples in the Monte Carlo simulation procedure. The idea of drawing resamples in a balanced manner is to ensure that each data appears the same number of times in all the resamples. The scheme of balanced resampling with B bootstrap resamples each of size n in estimating the distribution of $\sqrt{n}(\hat{\sigma} - \sigma)$ under balanced resampling is performed by the following steps:

Step 1. Repeatedly list each data B times, so that there are totally Bn data as the following:

$$\underbrace{(x_1, y_1), \dots, (x_1, y_1)}_{B\text{ times}}, \underbrace{(x_2, y_2), \dots, (x_2, y_2)}_{B\text{ times}}, \dots, \underbrace{(x_n, y_n), \dots, (x_n, y_n)}_{B\text{ times}}.$$

Step 2. Permute the above string of length Bn randomly, then group every n data to obtain B balanced resamples:

$$\underbrace{(x_{11}^{\dagger}, y_{11}^{\dagger}), \dots, (x_{1n}^{\dagger}, y_{1n}^{\dagger})}_{\mathcal{B}_1^{\dagger}}, \underbrace{(x_{21}^{\dagger}, y_{21}^{\dagger}), \dots, (x_{2n}^{\dagger}, y_{2n}^{\dagger})}_{\mathcal{B}_2^{\dagger}}, \dots, \underbrace{(x_{B1}^{\dagger}, y_{B1}^{\dagger}), \dots, (x_{Bn}^{\dagger}, y_{Bn}^{\dagger})}_{\mathcal{B}_B^{\dagger}}.$$

Step 3. The resulting balanced resamples are denoted by

$$\mathscr{B}_b^\dagger = \left\{(x_{b1}^\dagger, y_{b1}^\dagger), \dots, (x_{bn}^\dagger, y_{bn}^\dagger)\right\}, \quad b = 1, \dots, B.$$

For each balanced resample, $\mathscr{B}_b^\dagger, b = 1, \dots, B$, the bootstrap estimate of θ by minimizing $\sum_{i=1}^n \left(y_{bi}^\dagger - y(x_{bi}^\dagger; \theta)\right)^2$, we get $\hat{\theta}_b^\dagger$. Then we have bootstrap estimate of $\hat{\sigma}$ is:

$$\hat{\sigma}_b^\dagger = \sqrt{\frac{1}{n}\sum_{i=1}^n \left(y_{bi}^\dagger - y(x_{bi}^\dagger; \hat{\theta}_b^\dagger)\right)^2}.$$

The distribution of $\sqrt{n}(\hat{\sigma} - \sigma)$ can therefore be approximated by the empirical distribution of $\sqrt{n}(\hat{\sigma}_b^\dagger - \hat{\sigma}), b = 1, \dots, B$.

Let $\hat{\xi}_1, \hat{\xi}_2, \hat{\xi}_3$ be the first, second and third quartiles of the empirical distribution of $\sqrt{n}(\hat{\sigma}_b^\dagger - \hat{\sigma})$, $b = 1, \dots, B$. The difference between the third and first quartiles is called the interquartile range, IQR. The two values, $\hat{\xi}_2$ and IQR, measure the standard deviation of the model error term ϵ and the stability of the parameter estimation and they have the virtue of being robustness property. To choose between several models $M_1, M_2, \dots$, these computations are repeated for each of them, and the best one will be the one that has the best compromise to simultaneously minimize $\hat{\xi}_2$ and IQR.

3. Asymptotic relative efficiency

To estimate the distribution function F of $T = \sqrt{n}(\hat{\sigma} - \sigma)/\sqrt{\hat{\Sigma}}$, $\hat{\Sigma}$ being the estimated variance of $\hat{\sigma}$, let $\mathscr{B}^* = \{(x_1^*, y_1^*), \dots, (x_n^*, y_n^*)\}$ denote a resample draw randomly from $\mathscr{B}_0 = \{(x_1, y_1), \dots, (x_n, y_n)\}$ with replacement. Write $\hat{\sigma}^*$ (computed based on $\mathscr{B}^*$) for the version of $\hat{\sigma}$. Then the bootstrap estimate of F is $\hat{F}(x) = P(T^* \le x \mid \mathscr{B}_0)$, where $T^* = \sqrt{n}(\hat{\sigma}^* - \sigma)/\sqrt{\hat{\Sigma}^*}$, where $\hat{\Sigma}^*$ denote the version of $\hat{\Sigma}$ computed from $\mathscr{B}^*$. Since the Monte-Carlo simulation of $\hat{F}$ based on uniformly resampling (i.e., the original bootstrap method) is conditional on $\mathscr{B}_0$, draw B independent resamples $\mathscr{B}_1^*, \dots, \mathscr{B}_B^*$ by resampling uniformly from $\mathscr{B}_0$, and let T_b^* denote the version of T computed in $\mathscr{B}_b^*$. Then the uniformly resampling approximation of $\hat{F}$ is

$$\hat{F}_u(x) = \frac{1}{B}\sum_{b=1}^B I(T_b^* \le x),$$

where $I(\cdot)$ is the indicator function. By the strong law large numbers, conditional on $\mathscr{B}_0$, $\hat{F}_u \to \hat{F}$ with probability 1 as $B \to \infty$ and that $E(\hat{F}_u \mid \mathscr{B}_0) = \hat{F}$ and $Var(\hat{F}_u \mid \mathscr{B}_0) = B^{-1}\hat{F}(1 - \hat{F})$. Furthermore, by Glivenko–Cantelli's theorem, we have with probability 1, as $n \to \infty$,

$$\sup_{-\infty<x<\infty} |\hat{F}(x) - \Phi(x)| \to 0,$$

where $\Phi(x)$ is the distribution function of the standard normal distribution. Therefore, with probability 1, as $n \to \infty$,

$$\sup_{-\infty<x<\infty} |BVar(\hat{F}_u(x) \mid \mathscr{B}_0) - [1 - \Phi(x)]\Phi(x)| \to 0, \quad \text{as } B \to \infty. \tag{2}$$

To construct a balanced resampling approximation of $\hat{F}$, we draw B balanced resamples $\mathscr{B}_b^\dagger, b = 1, \dots, B$ (see Section 2). Let $\hat{\sigma}_b^\dagger$ denote the version of $\hat{\sigma}$ computed from $\mathscr{B}_b^\dagger$. Then the balanced resampling approximation of $\hat{F}$ is

$$\hat{F}_b(x) = \frac{1}{B}\sum_{b=1}^B I(T_b^\dagger \le x),$$

where $T_b^\dagger = \sqrt{n}(\hat{\sigma}_b^\dagger - \hat{\sigma})$. To obtain the asymptotic efficiency of $\hat{F}_b$ relative to $\hat{F}_u$, we need to calculate the asymptotic variance of $\hat{F}_b$. By theorem of Hall [8], we have, with probability 1, as $n \to \infty, B \to \infty$

$$\sup_{-\infty<x<\infty} |BVar(\hat{F}_b(x) \mid \mathscr{B}_0) - A(x)| \to 0, \tag{3}$$

where $A(x) = [1 - \Phi(x)]\Phi(x) - \phi(x)^2, \phi(x) = \Phi'(x)$.

Comparing results of Eqs. (2) and (3), we get the relative efficiency of bootstrap estimate under uniformly resampling and balanced resampling, as $n \to \infty$,

$$\begin{aligned} r(x) &= \frac{\text{variance of } \hat{F}_u(x)}{\text{variance of } \hat{F}_b(x)} \\ &\sim \frac{\Phi(x)[1 - \Phi(x)]}{\Phi(x)[1 - \Phi(x)] - \phi(x)^2}. \end{aligned}$$

Table 1
Comparison results of OB, WB and BR algorithms with different bootstrap replications B.

	B	M_1		M_2		M_3	
		$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR
WB	25	21.360	2.480	1.103	0.076	**1.087**	**0.074**
	50	21.329	2.419	1.090	0.075	1.070	0.070
BR	20	21.125	2.453	1.091	0.075	**1.075**	**0.073**
	40	20.796	2.359	1.063	0.073	1.043	0.068
OB	50	22.430	2.823	**1.111**	**0.113**	1.129	0.113
	100	22.325	2.531	1.103	0.107	1.114	0.109

Note that the second quartile $\hat{\xi}_2$ is corresponding to $x = 0$, i.e., $\Phi(x) = 0.5$. Thus the asymptotic relative efficiency is $r(0) = 2.752$ for $\hat{\xi}_2$.

4. Numerical experiments

To compare the original bootstrap (OB), weighted bootstrap (WB) and balanced resampling (BR), we borrow the examples from Chuang et al. [4] and Kallel et al. [1]. Based on Efron and Tibshirani's [9] experience, there are two rules of thumb to evaluate how large B is suitable: (i) Even a small number of bootstrap replications, say $B = 25$, is usually informative, while $B = 50$ is often enough to use; (ii) only very seldom are more than $B = 200$ replications needed. Therefore, we consider $B = 50$ and 100 for OB algorithm. In Chuang et al. [4], they consider $B = 25$ and 50 in the WB algorithm. Davison et al. [5] used the BR with $B = 19$ to approximate the bootstrap bias estimator of the sample variance. Hence, in this section, we consider $B = 20$ and 40 in the BR algorithm.

Example 1 (*cf. Chuang et al. [4]*)**.** Consider the problem of fitting a polynomial model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_p x^p + \epsilon.$$

A data set $\mathcal{B}_0$ is generated by putting

$$x_i = i^{1/3}, \qquad y_i = 4 + x_i + 2x_i^2 + 3x_i^3 + \epsilon_i, \quad i = 1, \ldots, 500,$$

where ϵ_i is a random error term which has the standard normal distribution. We consider three models

Model M_1: $y = \theta_0 + \theta_1 x + \epsilon$.
Model M_2: $y = \theta_0 + \theta_1 x + \theta_2 x^2 + \epsilon$.
Model M_3: $y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \epsilon$, (true model).

For each model, we compute $\hat{\xi}_2$ and IQR based on OB, WB and BR algorithms. The results are listed in Table 1, indicating that the best model for WB and BR algorithms is M_3 with $B = 25, 50, 20$ and 40, respectively. It is natural to pose the question: "Which one is appropriate?". Since the difference between 1.087 and 1.070$((1.087 - 1.070)/1.070 = 1.6\%)$ is negligible, we choose $B = 25$ for the WB algorithm. By the same reason, we also choose $B = 20$ for the BR algorithm. However, the best model for the OB algorithm is M_2, and the difference between 1.111 and 1.103$((1.111 - 1.103)/1.103 = 0.73\%)$ is also negligible. Thus, we choose $B = 50$ for the OB algorithm. But M_2 is not a true model. Therefore, the WB and BR algorithms not only permit a reduction in replication size but also select the correct model. However the advantage of BR is not necessary to select the weight in resampling.

Example 2. Kallel et al. [1] used Eq. (1) with sigmoid transfer function ϕ to simulate a data set

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

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

For each model, we compute $\hat{\xi}_2$ and IQR based on OB, WB and BR algorithms. Table 2 shows that the best model for WB and OB algorithms is M_2. The number of bootstrap replications B equals 50 for OB algorithm, the WB algorithms only need $B = 25$. But M_2 is not true model. However, the BR algorithm with $B = 20$ select M_4 to be the best and it is a true model. This illustrates that the BR algorithm can overcome the overparameterized problem in the multilayer perceptrons.

Example 3. Chuang et al. [4] discussed a real data set from p. 296 in Draper and Smith [10]. A proposed model for this data set, based on theoretical considerations, is $\log_{10} Y = \log_{10}\alpha + \beta \log_{10} X_1 + \gamma \log_{10} X_2 + \delta \log_{10} X_3 + \epsilon$, where $\alpha = 0.05$.

Table 2
Comparison results of OB, WB and BR algorithms with different bootstrap replications B.

	B	M_2		M_4		M_6	
		$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR
WB	25	**0.042**	**0.004**	0.043	0.004	0.044	0.004
	50	0.042	0.004	0.042	0.004	0.043	0.004
BR	20	0.043	0.004	**0.042**	**0.004**	0.044	0.004
	40	0.041	0.004	0.041	0.004	0.042	0.004
OB	50	**0.043**	**0.004**	0.044	0.004	0.045	0.004
	100	0.043	0.004	0.044	0.004	0.045	0.004

Table 3
Comparison results of OB, WB and BR algorithms with different bootstrap replications B.

	B	M_{123}		M_{12}		M_{13}		M_{23}	
		$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR
WB	25	**0.024**	**0.007**	0.028	0.008	0.307	0.071	0.029	0.007
	50	0.024	0.005	0.027	0.007	0.305	0.066	0.028	0.007
BR	20	**0.024**	**0.007**	0.028	0.008	0.306	0.071	0.029	0.007
	40	0.024	0.005	0.027	0.007	0.301	0.065	0.028	0.007
OB	50	**0.025**	**0.007**	0.028	0.009	0.319	0.079	0.029	0.008
	100	0.024	0.007	0.027	0.008	0.310	0.074	0.028	0.007

Table 4
Comparison results of OB, WB and BR algorithms with different bootstrap replications B.

	B	M_{30}		M_{35}		M_{40}	
		$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR	$\hat{\xi}_2$	IQR
WB	25	**0.046**	**0.007**	0.059	0.009	0.048	0.007
	50	0.046	0.007	0.058	0.009	0.047	0.006
BR	20	**0.049**	**0.007**	0.062	0.009	0.050	0.007
	40	0.047	0.005	0.057	0.008	0.047	0.006
OB	50	**0.047**	**0.007**	0.060	0.009	0.049	0.007
	100	0.047	0.007	0.060	0.009	0.049	0.006

In the following, we consider four models

Model M_{123}: $\log_{10} Y = \log_{10}\alpha + \beta\log_{10} X_1 + \gamma\log_{10} X_2 + \delta\log_{10} X_3 + \epsilon$ (true model).
Model M_{12}: $\log_{10} Y = \log_{10}\alpha + \beta\log_{10} X_1 + \gamma\log_{10} X_2 + \epsilon$.
Model M_{13}: $\log_{10} Y = \log_{10}\alpha + \beta\log_{10} X_1 + \delta\log_{10} X_3 + \epsilon$.
Model M_{23}: $\log_{10} Y = \log_{10}\alpha + \gamma\log_{10} X_2 + \delta\log_{10} X_3 + \epsilon$.

For each model, we compute $\hat{\xi}_2$ and IQR based on OB, WB and BR algorithms. Table 3 shows that the best model for these algorithms is M_{123}. From Table 3, we choose $B = 50$ for the OB algorithm. But the WB and BR algorithms only need $B = 25, 20$, respectively. It indicates that the WB and BR algorithms do permit a reduction in replication size.

Example 4. Kallel et al. [1] considered a real data set which explores the power peak control in the core of nuclear reactors [11]. Gaudier [11] constructed a neuron model with 22 input variables and 2 hidden layers, (the first one with 26 neurons, the other 40 neurons). The following models 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 $\hat{\xi}_2$ and IQR based on OB, WB and BR algorithms. Table 4 shows that the best model for these algorithms is M_{30}. From Table 4, we choose $B = 50$ for the OB algorithm. But the WB and BR algorithms only need $B = 25$, 20, respectively. It also indicates that the WB and BR algorithms do permit a reduction in replication size.

5. Conclusions

In this paper we propose the BR algorithm to reduce computer effort of the bootstrap method for neural model selection. Compared with OB, WB and BR algorithms, the numerical results show that the BR algorithm not only is an effective means of reducing the bootstrap replications but also selects the correct model. According to the experimental results, the BR algorithm permits an approximately 2.5 to 1 reduction in replication size. This result reflects the fact that the asymptotic relative efficiency is $r(0) = 2.752$. We mentioned that Chuang et al. [4] did not discuss the asymptotic relative efficiency of WB. Therefore, the proposed BR algorithm should be considered in the neural model selection.

Support vector machines are useful for modeling in statistical applications. Shen et al. [12] combined the vague and ill-defined information to propose a support vector fuzzy adaptive network in regression analysis. Suppose there are p explanatory variables and a response variable y. Since some of the explanatory variables may not be actually related y, the use of all the p explanatory variables as predictors does not necessary produce an accurate prediction. Suppose that α is a subset of $\{1, 2, \ldots, p\}$. Our problems is "How to select α such that the corresponding model produces more efficient predictions?" In the future, we will use the proposed BR algorithm to solve this problem.

References

[1] R. Kallel, M. Cottrell, V. Vigneron, Bootstrap for neural model selection, Neurocomputing 48 (2002) 175–183.
[2] B. Efron, Bootstrap methods: another look at the jackknife, Ann. Statist. 7 (1979) 1–26.
[3] A. Lendasse, G. Simon, V. Wertz, M. Verleysen, Fast bootstrap for regression model selection, Neurocomputing 64 (2005) 161–181.
[4] S.C. Chuang, W.L. Hung, H.C. Fu, Weighted bootstrap for neural model selection, Internat. J. Systems Sci. 39 (2008) 557–562.
[5] A.C. Davison, D.V. Hinkley, E. Schechtman, Efficient bootstrap simulation, Biometrika 73 (1986) 555–566.
[6] A.G. Davison, Discussion of papers by D.V. Hinkley and by T.J. DiCiccio and J.P. Romano, J. R. Stat. Soc. B 50 (1988) 356–357.
[7] J.R. Gleason, Algorithms for balanced bootstrap simulations, Amer. Statist. 42 (1989) 263–266.
[8] P. Hall, Performance of balanced bootstrap resampling in distribution function and quantile problems, Probab. Theory Related Fields 85 (1990) 239–260.
[9] B. Efron, R. Tibshirani, An Introduction to Bootstrap, Chapman and Hall, New York, 1993.
[10] N.R. Draper, H. Smith, Applied Regression Analysis, John Wiley, New York, 1998.
[11] F. Gaudier, Optimization et réseaux de neurones pour le repositionnement des barres de combustible nucléaire, Thèse de I'université Paris VI, ENS Cachan, 1998.
[12] J. Shen, Y.R. Syau, E.S. Lee, Support vector fuzzy adaptive network in regression analysis, Comput. Math. Appl 54 (2007) 1353–1366.