1999年度日本オペレーションズ・リサーチ学会
秋季研究発表会

Bootstrapping for Neural Network Learning

02991810 筑波大学大学院　*DUPRET Georges
01105930 筑波大学　香田正人　KODA Masato
koda@shako.sk.tsukuba.ac.jp

1. Introduction

Bootstrapping is a computer-based re-sampling technique for assigning measures of accuracy to statistical estimates. The bootstrap method, originated by Efron (Ref. Efron and Tibshirani (1993)), depends on the notion of a *bootstrap sample*. Suppose we observe a set of independent data points $x_1, x_2, \cdots, x_n$, denoted by a vector $x = (x_1, x_2, \cdots, x_n)^T$, from which we compute statistics of interest $s(x)$. Let $\hat{F}$ be the empirical distribution, putting probability $1/n$ on each of the observed values $x_i, i = 1,2,\cdots,n$. A bootstrap sample is defined to be a random sample of size n drawn from $\hat{F}$, say $x^* = (x_1^*, x_2^*, \cdots, x_n^*)^T$, i.e., $\hat{F} \rightarrow x^* = (x_1^*, x_2^*, \cdots, x_n^*)^T$. In other words, the bootstrap data points $x_1^*, x_2^*, \cdots, x_n^*$ are a random sample of size n drawn *with replacement* from the original data points $x_1, x_2, \cdots, x_n$. Hence, the bootstrap algorithm begins by generating a large number of independent bootstrap samples $x^{*1}, x^{*2}, \cdots, x^{*B}$, each of size n. Typical values of B, i.e., the number of bootstrap samples, may range from 50 to 200 for standard error estimation. Let us denote by $s(x^{*b})$ the value of the statistics s evaluated at x^{*b}, for $b = 1,2,\cdots,B$. Then the quantity $s(x^{*b})$ is the result of applying the same function $s(\bullet)$ to x^{*b} as was applied to x.

In our problem setting, we apply bootstrapping techniques to a training of neural networks by re-sampling the same number of sample size from the original training data, and then evaluate and test the network performance based on the complete set of original data. The neural networks used in this study belong to the class of back-propagation networks. Back-propagation refers to the method of computing the error gradient. Standard back-propagation, however, is a tedious, slow batch learning method. Thus, we have applied the optimization algorithm known as the conjugate gradient method in order to remedy the slow convergence of back-propagation networks. Conjugate gradient training techniques proposed in this study may require more iterations, but each iteration requires less floating-point computation and improves slower convergence. We have also applied this conjugate gradient training techniques to a stochastic learning algorithm referred to as Stochastic Noise Reaction (SNR) proposed by the second author (Koda (1995) (1997a)(1997b)).

2. Conjugate Gradient Training Techniques

In neural computations, the direction of the gradient descent is computed through conventional gradient algorithms. This defines the point in the parameter space from which we calculate the new gradient to update connection weights. The inconvenience of the method is, however, at each iteration, we may loose information we calculated in the previous step because the new search direction does not keep the property of being at the minimum on the line defined by the former direction. The proposed conjugate gradient training method attempts to remedy this inconvenience and accelerates the convergence.

Let $\mathbf{g}_0$ and $\mathbf{h}_0$ be equal to the negative gradient at the starting point in the weight space. The search direction at step i is defined by:

$$\mathbf{h}_i = \mathbf{g}_i + \gamma . \mathbf{h}_{i-1} \quad \text{where} \quad \gamma = \frac{(\mathbf{g}_i - \mathbf{g}_{i-1})^T . \mathbf{g}_i}{\mathbf{g}_{i-1}^T . \mathbf{g}_{i-1}}$$

The sequence of $\mathbf{h}_i$ vectors is mutually conjugate under suitable conditions.

3. Bootstrapping

This study proposes a method of bootstrapping for a random training of conjugate gradient learning using bootstrap samples with replicated runs. The focus is on learning accuracy and we would like to address empirically the following issue: if we know the true distribution function of training patterns, how should we sample in order to optimize the neural computation?

We will present and assess the bootstrap estimates based on the numerical experiments meeting the following requirements:

1. Sample space is finite and small.
2. Distribution function is unbalanced so that the effect of bootstrapping is easy to assess.

We will review and discuss the learning capability of the proposed network, whose performance could possibly be enhanced by the bootstrap training.

References

Efron, B., and Tibshirani, R. (1993) An Introduction to the Bootstrap. Chapman & Hall.

Koda, M. (1995) "Stochastic sensitivity analysis method for neural network learning." *Int. J. Syst. Sci.*, **26**, 703-711.

Koda, M. (1997a) "Neural network learning based on stochastic sensitivity analysis." *IEEE Trans. Syst., Man, and Cybern. Part B: Cybernetics*, **27**, 132-135.

Koda, M. (1997b) "Stochastic sensitivity analysis and Langevin simulation for neural network learning." *Reliability Eng. and Syst. Safety*, **57**, 71-78.