# Empirical Evaluation of Bootstrap Methodologies for Neural Networks

-

Students promo 2019-2021
Anh Tuan DAO - Thu Ha PHI

## Abstract

We explore bootstrap methods for deep networks to qualify uncertainty by approximately calculating posterior distribution of regression models. Weighted Bayesian Bootstrap (WBB) is a simulation-based algorithm for assessing uncertainty in machine learning and statistics [1]. In this project, we implement weighted bayesian bootstrap on datasets supplied from the UCI machine learning repository and make comparisons with the results of probabilistic backpropagation method (PBP) [2] and Dropout's uncertainty [3].

**Keywords:** Bootstrap, Weighted Likelihood Bootstrap, Weighted Bayesian Bootstrap, Deep Learning

# Table of contents

## I. Introduction

Deep learning has gained a lot of attention from researchers in fields like computer vision, speech recognition, natural language processing..., where deep learning has been proven to perform as well as or even better than humans. Despite impressive classification accuracy and mean squared errors, standard deep learning for regression and classification do not capture model uncertainty which is very important in sensitive domains such as healthcare and autonomous control. Uncertainty is also important in reinforcement learning as well (Szepesvari, 2010 Dropout). With uncertainty information an agent can decide when to exploit and when to explore its environment.

Recently, many methods are applied to obtain uncertainty from deep models such as probabilistic backpropagation (PBP) [2] and Dropout Uncertainty [3]. In this project, we describe the Weighted Bayesian Bootstrap method and its algorithm. We implement Weighted Bayesian Bootstrap method on datasets supplied from the UCI machine learning repository and estimate predictive uncertainty compared with results of probabilistic backpropagation (PBP) [2] and Dropout's uncertainty [3].

The rest of the report is outlined as follows. Section II simply describes the background knowledge such as Bootstrap method and Weighted Likelihood Bootstrap. Section III describes Weighted Bayesian Bootstrap and its algorithm. In section IV, we talk about the implementation and the result on UCI dataset. Finally, conclusions are included in section V.

## II. Background

### 1. Bootstrap

Bootstrap is a method typically used to estimate the expected prediction error. Suppose that we have a model fit for a set of training data by $Z = (z_1, z_2, ...z_n)$ where $z_i = (x_i, y_i)$. The basic idea of Bootstrap is: we randomly draw M training sets $Z_m^*$, m = 1,...,M each of size N with replacement from the original training set. We note that each dataset must have the same size as the original training set. Then, we refit the model to each of the bootstrap datasets, and examine the behavior of the fits over the M replications.

In practice, we have only a single dataset, thus Bootstrap is one of the approaches to introduce variability between the different models. Consider a regression problem in which we are trying to predict the value of a single continuous variable, and suppose we generate M bootstrap datasets and then use each to train a differently predictive model, called $y_m(x)$, where $m = 1, 2, ...M$. The final prediction is made by the average sum of M predictive model as following:

$$y_{COM}(x) = \frac{1}{M}\sum_{m=1}^{M} y_m(x) \quad (equation\ 2.1)$$

Suppose the true regression function that is used to generate the original dataset is h(x), and it is also the function we are trying to find within the committee. Thus, the output of each of the models can be written as the true value plus an error in the form:

$$y_m(x) = h(x) + \varepsilon_m(x) \quad (equation\ 2.2)$$

The average sum-of-squares error then takes the form:

$$E_x\left[\{y_m(x) - h(x)\}^2\right] = E_x\left[\varepsilon_m(x)^2\right] \quad (equation\ 2.3)$$

The average error made by the models is the average sum of Bootstrap models:

$$E_{AV} = \frac{1}{M}\sum_{m=1}^{M} E_x\left[\varepsilon_m(x)^2\right] \quad (equation\ 2.4)$$

Similarly, the expected error from the committee (equation 2.1) is given by

$$E_{COM} = E_x\left[\left\{\frac{1}{M}\sum_{m=1}^{M} y_m(x) - h(x)\right\}^2\right] = E_x\left[\left\{\frac{1}{M}\sum_{m=1}^{M} \varepsilon_m(x)\right\}^2\right]$$

$$(equation\ 2.5)$$

If we assume that the errors have zero mean and are uncorrelated, so that

$$E_x[\varepsilon_m(x)] = 0 \quad (equation\ 2.6)$$

$$E_x\left[\varepsilon_m(x)\varepsilon_l(x)\right] = 0, \quad m \neq l \quad (equation\ 2.7)$$

From that we obtain:

$$E_{COM} = \frac{1}{M} E_{AV} \quad (equation\ 2.8)$$

From equation 2.8, the result suggests that the average error of a model can be reduced by a factor of M simply by averaging M versions of Bootstrap model. However, this result depends on the key assumption (equation 2.6 & 2.7) that the errors due to the individual models are uncorrelated.

## 2. Weighted Likelihood Bootstrap

When the distribution function of $X_i$ is in a parametric family, LeCam (1956) proved that if the joint density function between $f(X|\theta)$ of X and the prior density $\pi(\theta)$ satisfy some regularity conditions, then the limiting posterior distribution of $\theta$ given *X* is normal. It means that, the large sample of posterior distribution is not dependent on the prior distribution of $\theta$. In particular, whatever the prior distribution of $\theta$ is, we can always use the normal distribution with an estimate covariance matrix to approximate the posterior distribution of theta. An explanation for this result is that, in large samples, the data totally dominate the prior beliefs [5].

However, to calculate the posterior distribution of theta based on the normal approximation, we have to know the form of the information matrix and this calculation requires a hard level of derivation. Thus, New-ton and Raftery (1994) [4] suggested the weighted likelihood bootstrap (WLB) method to address this problem without calculating the information matrix, which is an extension of Rubin's Bayesian bootstrap [6]. The below is the ideal of this method:

Assume that $X_1$, $X_2$,...$X_n$ are independent. For given $\theta$, each $X_i$ has a density function $f_i(X_i|\theta)$. The values of $\theta$ are found by maximizing the likelihood function:

$$L(\theta) = \sum_{i=1}^{n} f_i(X_i|\theta) \quad (equation\ 2.9)$$

To stimulate the posterior distribution of $\theta$ for any prior density $\pi(\theta)$, we generate a weight vector $w = (w_1, w_2, ..., w_n)$, which has some probability distribution determined by statisticians. We assume that w belongs to the uniform Dirichlet , i.e $w_i = Z_i / \sum_{j=1}^{n} Z_i$ , or over- (down-) dispersed relative to the Dirichlet distribution, i.e, $w_i \sim Z_i^{\alpha},\ \alpha > 1\ (\alpha < 1)$ where $Z_1, Z_2, ..., Z_n$ independent

of X, and i.i.d from the exponential distribution with mean 1. We then define the weighted likelihood function:

$$\widehat{L}(\theta) = \sum_{i=1}^{n} f_i(X_i|\theta)^{w_i} \quad (equation\ 2.10)$$

Let $\widehat{\theta}_n$ be any value maximizing $\widehat{L}(\theta)$. We can estimate approximately the posterior distribution of $\theta$ by combining the set of $\widehat{\theta}_n$ which is obtained from repeatedly generating weight vectors and maximizing $\widehat{L}(\theta)$. In particular, let $\widehat{\theta}_{nb}$ be the value that maximizes the weighted likelihood function corresponding to the $b^{th}$ weight vector, where b = 1,2,...,B. The empirical distribution of $\widehat{\theta}_{nb}$, b = 1, …,B, can be used to approximate the posterior distribution of $\theta$.

### 3. Predictive Log-likelihood

Predictive log-likelihood captures how well a model fits the data, with larger values indicating better model fit [3]. Given a dataset X; Y and a new data point $x^*$ we can calculate the probability of possible output values $y^*$ using the predictive probability p($y^*|x^*$; X; Y). The log of the predictive likelihood captures how well the model fits the data, with larger values indicating better model fit.

For regression we have

$$log\ p(y^*|x^*, X, Y) \approx logsumexp\left(-\tfrac{1}{2}\tau||y - \widehat{y}_t||^2\right) - log\ T\ -\ \tfrac{1}{2}log\ 2\pi\ -\tfrac{1}{2}log\ \tau^{-1}$$

$$(equation\ 2.11)$$

where τ is our precision parameter.

## III. Weighted Bayesian Bootstrap

Unlikely weighted likelihood bootstrap regardless of the priority distribution of $\theta$, weighted Bayesian bootstrap uses information of the priority distribution of $\theta$. It means that we add a penalty function beside likelihood function, then our optimization duality now becomes:

$$minimize_{\theta \in R^d}\ l(y|\theta) + \lambda\phi(\theta) \quad (equation\ 3.1)\ [1]$$

Let define each particular element in equation 3.1, and explain why we have to find the value of $\theta$ making $l(y|\theta) + \lambda\phi(\theta)$ minimized. From the Bayesian perspectives, the measure of fit, $l(y|\theta) = -log\, f(y;\theta)$, and the penalty function, $\lambda\phi(\theta)$, correspond to the negative logarithms of the likelihood and prior distribution in the hierarchical model, as following:

$$f(y;\theta) = p(y|\theta) \sim exp\{-l(y|\theta)\}, \quad p(\theta) \sim exp\{-\lambda\phi(\theta)\} \quad (equation\ 3.2)$$

Then the posterior distribution is:

$$p(\theta|y) \sim exp\{-(l(y|\theta) + \lambda\phi(\theta))\} \quad (equation\ 3.3)$$

From equation 3.3, we have to find the value of $\theta$ that makes the posterior distribution $p(\theta|y)$ maximized, and it is the value of $\theta$ making $(l(y|\theta) + \lambda\phi(\theta))$ become minimized.

**Weighted Bayesian Bootstrap algorithm [1]**

According to Newton and Raftery [4], the construction of a randomly weighted posterior distribution is denoted by:

$$w = (w_1, ..., w_n, w_p),\ p_w(\theta|y) \sim \prod_{i=1}^{n} p(y_i|\theta)^{w_i} p(\theta)^{w_p} \quad (equation\ 3.4)$$

Where the weights $w_p, w_i \sim Exp$ are randomly generated weights. It is equal to draw $w_i = log(1/U_i)$, where $U_i's$ are i.i.d. For the i.i.d observations, the likelihood can be factorized as $p(y|\theta) = \prod_{i=1}^{n} p(y_i|\theta)$.

Now, we can summarize the optimal duality for Weighted Bayesian Bootstrap:

$$\theta^*_{w,n} := arg_\theta max\, p_w(\theta|y) \equiv arg_\theta min \sum_{i=1}^{n} w_i l_i(y_i|\theta) + \lambda w_p \phi(\theta) \quad (equation\ 3.5)$$

Where $l_i(y_i|\theta) = -log\, p(y_i|\theta)$ and $\lambda\phi(\theta) = -log\, p(\theta)$.

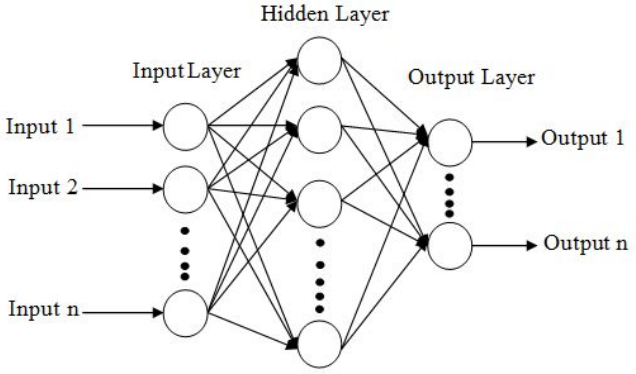
Here is the procedure to find the weighted posterior mode $\{\theta^*_{w,n}\}$:

1. Iterate: sample $w = \{w_1, w_2, ..., w_n, w_p\}$ via exponentials. $w_p, w_i \sim Exp(1)$.
2. For each "w', solve $\theta^*_{w,n} = arg_\theta min \sum_{i=1}^{n} w_i l_i(\theta) + \lambda w_p \phi(\theta)$.
3. The weighted Bayesian Bootstrap draws are approximate posterior samples. $\left\{\theta^{*(k)}_{w,n}\right\}_{k=1}^{K} \sim p(\theta|y)$.

## IV. Experimental results

### 1. Model

We construct a neural network with a single hidden layer for regression tasks.




We denote $W_1$ is the weight matrices connecting the input layer to the hidden layer and $W_2$ is the weight matrices connecting the hidden layer to the output layer and $b$ is the biases of hidden layer. We add Relu activation after hidden layer, denoted $h(.)$.

So the output of our neural network would be:

$$\widehat{y} = h(x * W_1 + b) * W_2 \quad (equation\ 4.1)$$

In weighted Bayesian bootstrap, we have a weighted likelihood and a new regularization parameter $\lambda W_p$ in optimized function:

$$\theta^*_{w,n} := arg_\theta max\, p_w(\theta|y) \equiv arg_\theta min \sum_{i=1}^{n} w_i l_i(y_i|\theta) + \lambda w_p \phi(\theta) \quad (equation\ 4.2)$$

Where $w = \{w_1; w_2; ...; w_n; w_p\}$ are sampled via exponentials. $w_p; w_i \sim Exp(1)$.

Thus we use square loss function weighted with $w_i$ sampled above:

$$E = \sum_{i=1}^{n} w_i * ||y_i - \widehat{y}_i||_2^2 \quad (equation\ 4.3)$$

$L_2$ regularisation weighted by weight decay λ and $W_p$ is added to minimization objective and finally WBB minimization objective become:

$$L_{WBB} = E + \lambda * W_p * \left(||W_1||_2^2 + ||W_2||_2^2 + ||b||_2^2\right) \quad (equation\ 4.4)$$

We use Adam optimization with learning rate equal 0.01 to train the network and set weight decay λ of $L_2$ regularisation equal 0.001.

## 2. Dataset

We train and test models on the datasets which are taken from the UCI machine learning repository. Due to the small size of the data, if we ourselves split the data we will most likely get different and non-comparable results. So we keep train on split datasets which are identical to the ones used in Hernández-Lobato's code and make comparisons.

We use the experimental setup proposed by Hernandez-Lobato and Adams ´ [2] for evaluating PBP. Each dataset is split into 20 train-test folds, except for the protein dataset which uses 5 folds and the Year Prediction MSD dataset which uses a single train-test split.

| Dataset | Dataset size | Input dimension |
|---|---|---|
| Boston Housing | 506 | 13 |
| Concrete Strength | 1,030 | 8 |
| Energy Efficiency | 768 | 8 |
| Kin8nm | 8,192 | 8 |
| Naval Propulsion | 11,934 | 16 |
| Power Plant | 9,568 | 4 |
| Protein Structure | 45,730 | 9 |
| Wine Quality Red | 1,599 | 11 |
| Yacht Hydrodynamics | 308 | 6 |
| Year Prediction MSD | 515,345 | 90 |

*Table* 1 : *Dataset information*

## 3. Results

In our experiment, we use the same network architecture: 1-hidden layer neural network containing 50 hidden units with a ReLU activation.

For each splitted train-test folds, we train an ensemble of 100 networks with different random initializations. We train 300 epochs for each model. Our results are shown in table below, along with the results of PBP method [2] and Dropout method [3].

| | **Avg. Test RMSE and Std. Errors** | | | **Avg. Test LL and Std. Errors** | | |
|---|---|---|---|---|---|---|
| **Dataset** | **PBP [2]** | **Dropout [3]** | **our method** | **PBP [2]** | **Dropout [3]** | **our method** |
| Boston Housing | 3.01 ± 0.18 | 2.97 ± 0.19 | **1.50 ± 0.16** | -2.57 ± 0.09 | **-2.46 ± 0.06** | -2.90 ± 0.56 |
| Concrete Strength | 5.67 ± 0.09 | 5.23 ± 0.12 | **3.01 ± 0.21** | -3.16 ± 0.02 | **-3.04 ± 0.02** | -7.04 ± 1.00 |
| Energy Efficiency | 1.80 ± 0.05 | 1.66 ± 0.04 | **0.34 ± 0.00** | -2.04 ± 0.02 | -1.99 ± 0.02 | **-0.29 ± 0.01** |
| Kin8nm | 0.10 ± 0.00 | 0.10 ± 0.00 | **0.06 ± 0.00** | 0.90 ± 0.01 | **0.95 ± 0.01** | 0.2 ± 0.00 |
| Naval Propulsion | 0.01 ± 0.00 | 0.01 ± 0.00 | **0.00 ± 0.00** | 3.73 ± 0.01 | **3.80 ± 0.01** | 0.23 ± 0.00 |
| Power Plant | 4.12 ± 0.03 | 4.02 ± 0.04 | Not Run | -2.84 ± 0.01 | -2.80 ± 0.01 | Not Run |
| Protein Structure | 473 ± 0.01 | 4.36 ± 0.01 | Not Run | -2.97 ± 0.00 | -2.89 ± 0.00 | Not Run |
| Wine Quality Red | 0.64 ± 0.01 | 0.62 ± 0.01 | **0.42 ± 0.02** | -0.97 ± 0.01 | -0.93 ± 0.01 | **-0.43 ± 0.04** |
| Yacht Hydrodynamics | 1.02 ± 0.05 | 1.11 ± 0.09 | **0.32 ± 0.04** | -1.63 ± 0.02 | -1.55 ± 0.03 | **-0.29 ± 0.07** |
| Year Prediction MSD | 8.879 ± NA | 8.849 ± NA | Not Run | -3.603 ± NA | -3.588 ± NA | Not Run |

*Table 2 : Average test performance in RMSE and predictive log likelihood*

Because training requires a lot of time, we have trained 7/10 datasets except Power Plant, Protein Structure and Year Prediction MSD which are large datasets. We observe that our method has better RMSE in all datasets. In terms of predictive log likelihood, our method is better in Energy Efficiency,

Wine Quality Red, Yacht Hydrodynamics and the result is worst in other datasets.

## V. Conclusions

We explored bootstrap methods and weighted bayesian bootstrap algorithm [1] to estimate model uncertainty in deep network.

We also implemented a deep network for regression task in PyTorch and applied weighted bayesian bootstrap algorithm to estimate model uncertainty, then we made comparisons with results of probabilistic backpropagation (PBP) method [2] and Dropout method [3].

## References


[1] Newton, Michael, Nicholas G. Polson, and Jianeng Xu. "Weighted Bayesian Bootstrap for Scalable Bayes." *arXiv preprint arXiv:1803.04559* (2018).

[2] Hernández-Lobato, José Miguel, and Ryan Adams. "Probabilistic backpropagation for scalable learning of bayesian neural networks." *International Conference on Machine Learning*. 2015.

[3] Gal, Yarin, and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning." *international conference on machine learning*. 2016.

[4] Newton, Michael A., and Adrian E. Raftery. "Approximate Bayesian inference with the weighted likelihood bootstrap." *Journal of the Royal Statistical Society: Series B (Methodological)* 56.1 (1994): 3-26.

[5] Shao, Jun, and Dongsheng Tu. "Bayesian Bootstrap and Random Weighting." *The Jackknife and Bootstrap*. Springer, New York, NY, 1995. 416-446.

[6] Rubin, Donald B. "The bayesian bootstrap." *The annals of statistics* (1981): 130-134.