# Batch gradient method with smoothing $L_{1/2}$ regularization for training of feedforward neural networks[☆]

Wei Wu[a], Qinwei Fan[a,b], Jacek M. Zurada[b,c,*], Jian Wang[a,d], Dakun Yang[a], Yan Liu[a,e]

[a] School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, PR China
[b] Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY 40292, USA
[c] Spoleczna Akademia Nauk, 90-011 Lodz, Poland
[d] School of Mathematics and Computational Sciences, China University of Petroleum, Dongying 257061, PR China
[e] School of information Science and Engineering, Dalian Polytechnic University, Dalian 116034, PR China

**Abstract**

The aim of this paper is to develop a novel method to prune feedforward neural networks by introducing an $L_{1/2}$ regularization term into the error function. This procedure forces weights to become smaller during the training and can eventually removed after the training. The usual $L_{1/2}$ regularization term involves absolute values and is not differentiable at the origin, which typically causes oscillation of the gradient of the error function during the training. A key point of this paper is to modify the usual $L_{1/2}$ regularization term by smoothing it at the origin. This approach offers the following three advantages: First, it removes the oscillation of the gradient value. Second, it gives better pruning, namely the final weights to be removed are smaller than those produced through the usual $L_{1/2}$ regularization. Third, it makes it possible to prove the convergence of the training. Supporting numerical examples are also provided.

*Key words:* Feedforward neural networks, Batch gradient method, Smoothing $L_{1/2}$ regularization, Convergence

## 1. Introduction

Artificial neural networks have been widely used in many applications such as intelligent information processing, pattern recognition, feature extraction and others [1–8]. Backpropagation algorithm is a popular learning algorithm for feedforward neural networks (FNN) [9–12]. There are two practical ways to implement the backpropagation algorithm: batch updating approach and online updating approach. They perform the update after all the training samples are, or each sample is, supplied to the network during the learning process, respectively. The batch approach is considered in this paper.

The aim of this paper is to prune the FNN by introducing into the conventional error function a smoothing $L_{1/2}$ regularization term as a brute-force to prevent the weights from taking too large values. This term works to drive unnecessary weights to zero during the training.

Penalty (or regularization) terms are often introduced into the learning procedure and have shown to be efficient to improve the generalization performance and to decrease the magnitude of the network weights [1, 10, 13, 14]. A typical example of the penalty term was proposed in [9, 11] by introducing a "weight decay term" $hW$ or $2\lambda W$ into the backpropagation algorithm, where $h$ or $\lambda$ is a very small positive number and $W$ is the weight vector. This is equivalent to adding the quadratic sum of the weight parameters values as a penalty term into the conventional error function. Then, the error function with the penalty term is defined as follows:

$$E(W) = \tilde{E}(W) + \lambda \sum_{i \in C} w_i^2 = \frac{1}{2} \sum_{j=1}^{J} e_j^2 + \lambda \|W\|^2 \quad (1.1)$$

where $\tilde{E}(W) = \sum_{j=1}^{J} e_j^2$ is the conventional square error function for measuring the accuracy of the network, $J$ is the number of training samples, $e_j$ is the error between the network output value and the ideal output value for the $j$th sample, $C$ is the set of all weights, $\lambda$ is the penalty coefficient, and $\|\cdot\|$ stands for the Eu-

[☆]This work was supported by National Science Foundation of China (No.11171367) and China Scholar ship Council (CSC).

[*]Corresponding author.
*Email address:* jacek.zurada@louisville.edu (Jacek M. Zurada[b,c,*])

clidean norm. Now, the original weight updating rule $\Delta W = -\eta \frac{\partial E(W)}{\partial W}$ is changed into

$$\Delta W = -\eta(\frac{\partial \tilde{E}(W)}{\partial W} + 2\lambda W)$$

where $\eta$ is the learning rate. A convergence analysis was carried out in [12] for FNN with a hidden layer as a special case of (1.1). Similar penalty term methods are also studied in e.g. [9, 10, 15–18]. But these penalty methods do not work very well for the purpose of driving unnecessary weights to zero.

The idea of $L_{1/2}$ regularization has been successfully applied in variable selection and feature extraction problems in high dimensional and massive data analysis. The traditional variable selection criteria such as AIC, BIC and CP [19–21] involve solving an NP hard optimization problem so they are infeasible for high dimensional data. Consequently, innovative variable selection procedure is expected to cope with very high dimensionality, which is one of the hot topics in machine learning. The regularization methods are recently used as feasible approaches to solve the problem. In general, the regularization methods have the form:

$$\min\{\frac{1}{n}\sum_{i=1}^{n} l(y_i, f(x_i)) + \lambda\|f\|_k\} \qquad (1.2)$$

where $l(.,.)$ is a loss function, $(x_i, y_i)_{i=1}^n$ is a data set, and $\lambda$ is the regularization parameter. When $f$ is in the linear form and the loss function is square loss, $\|f\|_k$ is normally taken as the norm of the coefficient of linear model.

In 2008, [25] proposed the following $L_{1/2}$ regularizer:

$$\hat{\beta}_{L_{\frac{1}{2}}} = argmin\{\frac{1}{n}\sum_{i=1}^{n}(Y_i - X_i^T\beta)^2 + \lambda\sum_{i=1}^{p}|\beta_i|^{\frac{1}{2}}\} \qquad (1.3)$$

where $\lambda$ is the tuning parameter. $L_{1/2}$ regularizer has a nonconvex penalty and possesses many promising properties such as unbiasedness, sparsity and oracle properties. Fig.1 show the sparsity by the graphics of the $L_{1/2}$ and $L_2$ regularizers. As shown in Fig.1, the sparsity solution is the first place at which the contours touch the constraint region, and this will concur at a corner corresponding to a zero coefficient. It is obvious that the solution of $L_{1/2}$ regularizer occurs at a corner with a higher possibility, which imply that it is more sparse than the $L_2$ regularizer. In fact, the training of FNNs is equivalent to minimizing an unconstrained optimization problem, there are some other methods for the nonsmooth optimization problems, such as the generalized gradient and recurrent neural network methods shown as [22–24].

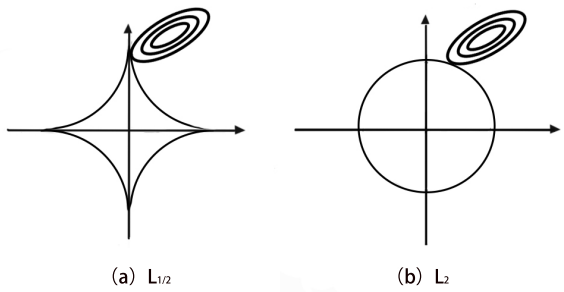


Figure 1: The sparsity property of (a) $L_{1/2}$ (b) $L_2$ regularizers

In this paper, the $L_{1/2}$ regularization term is introduced into the batch gradient learning algorithm for the pruning of FNN. In doing so, we notice that the usual $L_{1/2}$ regularization term is not smooth at the origin, which causes difficulty in the convergence analysis and more importantly, oscillation in the numerical computation as observed in our numerical experiments. To overcome this drawback, a modified $L_{1/2}$ regularization term is proposed by smoothing the usual one at the origin. Numerical examples are carried out to show the effectiveness of the proposed approach. The convergence of the corresponding batch gradient method is also proved.

The rest of this paper is arranged as follows. The batch gradient method with smoothing $L_{1/2}$ regularization penalty term (BGMSR) is described in Section 2. In Section 3, the convergence results of BGMSR are presented, and the detailed proofs of the main results are stated as Appendix. Supporting numerical experiments are presented in Section 4. In Section 5, we conclude this paper with some remarks.

## 2. Batch gradient method with smoothing $L_{1/2}$ regularization (BGMSR)

### 2.1. Batch gradient method with $L_{1/2}$ regularization (BGMR)

Consider a three-layer network consisting of $p$ input nodes, $q$ hidden nodes, and 1 output node. Let $w_0 = (w_{10}, w_{20}, \cdots, w_{q0})^T \in R^q$ be the weight vector between the hidden units and the output unit, and $w_i = (w_{i1}, w_{i2}, \cdots, w_{ip})^T \in R^p$ be the weight vector between the input units and the hidden unit $i$ $(i = 1, 2, \cdots, q)$. To simplify the presentation, we write all the weight parameters in a compact form $W = (w_0^T, w_1^T, \cdots, w_q^T)^T \in R^{q+pq}$ and we define a matrix $V = (w_1, w_2, \cdots, w_q)^T \in$

$R^{q\times p}$. Let $g : R \to R$ be a given transfer function for the hidden and output nodes, which is typically, but not necessarily, a sigmoid function. We also define a vector function $G : R^q \to R^q$, $G(x) = (g(x_1), g(x_2), \cdots, g(x_q))^T$ for $x = (x_1, x_2, \cdots, x_q)^T \in R^q$. Suppose that $\{\xi^j, O^j\}_{j=1}^J \subset R^p \times R$ is a given set of training samples. We denote the error containing no penalty term by $\tilde{E}(W)$, i.e.,

$$\begin{aligned}\tilde{E}(W) &= \frac{1}{2}\sum_{j=1}^{J}(O^j - g(w_0 \cdot G(V\xi^j)))^2 \\ &= \sum_{j=1}^{J} g_j(w_0 \cdot G(V\xi^j))\end{aligned}$$

where $g_j(t) := \frac{1}{2}(O^j - g(t))^2$. Then

$$\tilde{E}_{w_{i0}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j))g(w_i\xi^j)$$

$$\tilde{E}_{w_{ik}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j))w_{i0}g'(w_i \cdot \xi^j)\xi_k^j$$

The error function with the $L_{1/2}$ regularization penalty term is

$$E(W) = \tilde{E}(W) + \lambda\sum_{i=1}^{q}\sum_{k=0}^{p}|w_{ik}|^{\frac{1}{2}} \tag{2.1}$$

where $|\cdot|$ denotes the absolute value. The gradient of the error function is as follow:

$$\begin{aligned}E_W(W) =&(E_{w_{10}}(W), E_{w_{20}}(W), \cdots, E_{w_{q0}}(W),\\ &E_{w_{11}}(W), E_{w_{12}}(W), \cdots, E_{w_{1p}}(W),\\ &E_{w_{21}}(W), E_{w_{22}}(W), \cdots, E_{w_{2p}}(W),\\ &\cdots E_{w_{q1}}(W), E_{w_{q2}}(W), \cdots, E_{w_{qp}}(W))^T\end{aligned} \tag{2.2}$$

with

$$E_{w_{i0}}(W) = \tilde{E}_{w_{i0}}(W) + \frac{\lambda sgn(w_{i0})}{2|w_{i0}|^{\frac{1}{2}}} \tag{2.3}$$

$$E_{w_{ik}}(W) = \tilde{E}_{w_{ik}}(W) + \frac{\lambda sgn(w_{ik})}{2|w_{ik}|^{\frac{1}{2}}} \tag{2.4}$$

where $i = 1, 2, ..., q$, $k = 1, 2, ..., p$.

Starting with an arbitrary initial value $W^0$, the weights $\{W^n\}$ are updated iteratively by

$$W^{n+1} = W^n + \Delta W^n, \quad n = 0, 1, 2, \cdots \tag{2.5}$$

with

$$\Delta w_{i0}^n = -\eta[\tilde{E}_{w_{i0}^n}(W) + \frac{\lambda sgn(w_{i0}^n)}{2|w_{i0}^n|^{\frac{1}{2}}}] \tag{2.6}$$

$$\Delta w_{ik}^n = -\eta[\tilde{E}_{w_{ik}^n}(W) + \frac{\lambda sgn(w_{ik}^n)}{2|w_{ik}^n|^{\frac{1}{2}}}] \tag{2.7}$$

where $i = 1, 2, ..., q$, $k = 1, 2, ..., p$, and the learning rate $\eta > 0$ is a constant.

### 2.2. Smoothing $L_{1/2}$ regularization (BGMSR)

The usual $L_{1/2}$ regularization term used in the last subsection is not smooth at the origin, which causes difficulty in the convergence analysis and, more importantly, oscillation in the numerical computation as observed in our numerical experiments (see Section IV). To overcome this drawback, a modified $L_{1/2}$ regularization term is proposed by smoothing the usual one at the origin, resulting in the following error function with a smoothing $L_{1/2}$ regularization penalty term:

$$E(W) = \tilde{E}(W) + \lambda\sum_{i=1}^{q}\sum_{k=0}^{p} f(w_{ik})^{\frac{1}{2}} \tag{2.8}$$

where $f(x)$ is a smooth function that approximates $|x|$. For definiteness and simplicity, we choose $f(x)$ as a piecewise polynomial function:

$$f(x) = \begin{cases} -x, & x \le -a \\ -\frac{1}{8a^3}x^4 + \frac{3}{4a}x^2 + \frac{3}{8}a, & -a < x < a \\ x, & x \ge a \end{cases} \tag{2.9}$$

It is not hard to get

$$f(x) \in [\frac{3}{8}a, +\infty), \quad f'(x) \in [-1, 1], \quad f''(x) \in [0, \frac{3}{2a}]$$

where $a$ is a small positive constant which is close to zero. Fig.2 show that $f(x)$ approximate to $|x|$, in order to see the approximate effect clearly we set $a = 0.005$.

The gradient of the error function can be written as (2.2),where

$$E_{w_{i0}}(W) = \tilde{E}_{w_{i0}}(W) + \lambda\frac{f'(w_{i0})}{2f(w_{i0})^{\frac{1}{2}}} \tag{2.10}$$

$$E_{w_{ik}}(W) = \tilde{E}_{w_{ik}}(W) + \lambda\frac{f'(w_{ik})}{2f(w_{ik})^{\frac{1}{2}}} \tag{2.11}$$

where $i = 1, 2, ..., q$, $k = 1, 2, ..., p$.

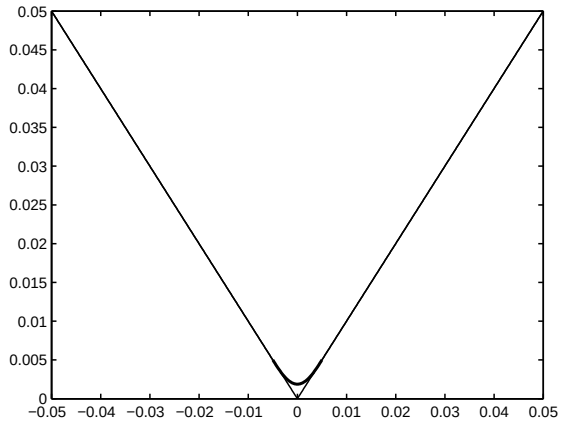


Figure 2: $f(x)$ approximate to $|x|$

Our BGMSR for the training of the network is: Starting with an initial value $W^0$, the weights $\{W^n\}$ are updated iteratively by

$$W^{n+1} = W^n + \Delta W^n, \quad n = 0, 1, 2, \cdots \tag{2.12}$$

where

$$\Delta w_{i0}^n = -\eta[\tilde{E}_{w_{i0}^n}(W) + \lambda \frac{f'(w_{i0}^n)}{2f(w_{i0}^n)^{\frac{1}{2}}}] \tag{2.13}$$

$$\Delta w_{ik}^n = -\eta[\tilde{E}_{w_{ik}^n}(W) + \lambda \frac{f'(w_{ik}^n)}{2f(w_{ik}^n)^{\frac{1}{2}}}] \tag{2.14}$$

where $i = 1, 2, ..., q$, $k = 1, 2, ..., p$, and the learning rate $\eta > 0$ is a constant.

## 3. Convergence Results

In this section we present some convergence theorems of the batch gradient method with smoothing $L_{1/2}$ regularization (2.12). Their proofs are given in the appendix. Some sufficient conditions for the convergence are as follows:

$(A1)$ $|g(t)|$, $|g'(t)|$, $|g''(t)|$ are uniformly bounded for $t \in R$.

$(A2)$ $\|w_0^n\|$ $(n = 0, 1, 2, ...)$ are uniformly bounded.

$(A3)$ $\eta$ and $\lambda$ are chosen to satisfy $0 < \eta < \frac{1}{M\lambda + C_1}$, where $M = \frac{\sqrt{6}}{4\sqrt{a^3}}$ and $C_1$ is a constant defined in (3.1) below.

$(A4)$ There exists a compact set $\Phi$ such that $W^n \in \Phi$ and the set $\Phi_0 = \{W \in \Phi : E_W(W) = 0\}$ contains finite points.

Some special constants used here and later on in the proofs are as follows:

$$\begin{aligned} C_1 &= J(1 + C_2)C_3 \max\{C_2^2, C_5^2\} + \frac{1}{2}J(1 + C_2)C_3 \\ &\quad + \frac{1}{2}JC_3^2C_4^2C_5, \\ C_2 &= \max\{\sqrt{q}C_3, (C_3C_4)^2\}, \\ C_3 &= \max\{\sup_{t\in R}|g(t)|, \sup_{t\in R}|g'(t)|, \sup_{t\in R}|g''(t)|, \\ &\quad \sup_{t\in R, 1\le j\le J}|g_j'(t)|, \sup_{t\in R, 1\le j\le J}|g_j''(t)|\}, \\ C_4 &= \max_{1\le j\le J}\|\xi^j\|, \quad C_5 = \sup_{n\in N}\|w_0^n\|. \end{aligned} \tag{3.1}$$

**Theorem 3.1.** Let the error function be defined by (2.8), and the weight sequence $\{W^n\}$ be generated by the iteration algorithm (2.12) for an arbitrary initial value. If assumptions $(A1) - (A3)$ are valid, then we have

(i) $E(W^{n+1}) \le E(W^n)$, $n = 0, 1, 2, ...$;

(ii) There exists $E^* \ge 0$ such that $\lim_{k\to\infty} E(W^n) = E^*$;

(iii) $\lim_{n\to\infty} \|E_W(W^n)\| = 0$.

Furthermore, if assumption $(A4)$ also holds, then we have the following strong convergence:

(iv) There exists a point $W^* \in \Phi_0$ such that $\lim_{n\to\infty}(W^n) = W^*$.

## 4. Numerical Examples

To demonstrate the sparsity of our batch gradient method with $L_{1/2}$ regularization smoothing, we compare it with the BGMR for the error function (2.1), the batch gradient method with square penalty (BGMS) for the conventional error function (1.1) in [9] with the following three numerical examples: XOR problem, Parity problem and the Sonar benchmark problem,

### *4.1. XOR problem and Parity problem*

In this example, the logistic function $g(x) = tansig(x)$ is employed as the activation function. In order to evaluate the advantage of our algorithm, a typical performance of the three different algorithms in one of the ten trials is shown in Fig.3-8. The two networks with the structure 3-5-1 (input, hidden and output nodes) and 6-7-1, respectively.

#### *4.1.1. Parameters involved in network training*

In the XOR problem, the learning rate $\eta$ and the penalty parameter $\lambda$ were 0.03 and 0.019 respectively.

In the Parity problem, the learning rate $\eta$ and the penalty parameter $\lambda$ were 0.1 and 0.01 respectively.

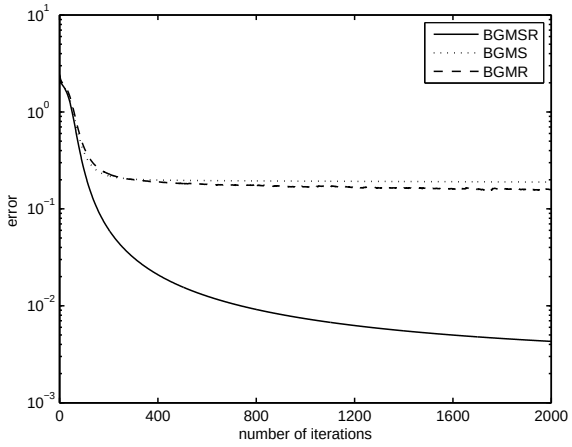


Figure 3: XOR problem: Value of the error function with different algorithms (BGMSR,BGMS, BGMR) for $\eta = 0.03$, $\lambda = 0.01$.

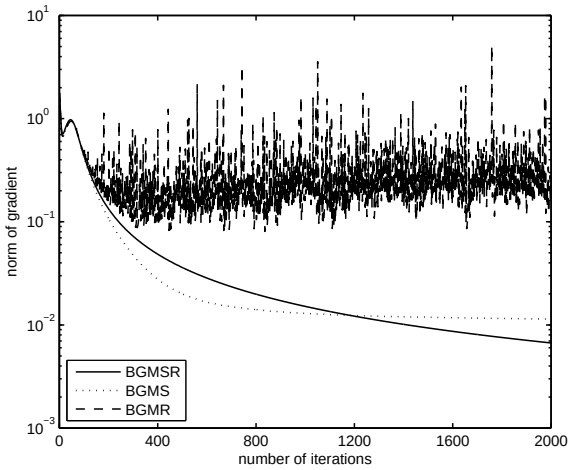


Figure 4: XOR problem: Norm of gradient with different algorithms (BGMSR, BGMS, BGMR) for $\eta = 0.03$, $\lambda = 0.01$.

#### *4.1.2. Numerical results*

In the XOR and parity problem, the experiment results for error and gradient are shown in Fig.3 and 4, Fig.5 and 6 respectively.

From Fig.3 and 5, Fig.4 and 6 we observe that the error function of BGMSR, BGMS and BGMR decreases monotonically, and both the norm of gradient of error function approaches to a very small positive constant, as predicted by Theorem 3.1. It can also be seen that with the same training parameters the error of BGMSR is smaller than BGMS and BGMR, and the BGMSR converges faster than BGMS and BGMR.

The preponderance of sparseness for BGMSR can be expressed by selected average of top smallest of all weights.In the XOR problem, first we fix $\eta$ and then the $\lambda$ variable is changed between 0 to 0.025, we take the average from the smallest 12 weights. For the parity problem, in terms of the XOR problem, we take the average from the smallest 33 weights, and then compare them by BGMSR, BGMR and BGMS.

From Fig.7 and 8 shown that the average of top 12 smallest for BGMSR in XOR problem and top 33 smallest for BGMSR in parity problem are much smaller than BGMR and BGMS, and this demonstrates the effectiveness of the algorithm in pruning the magnitude of weights and eventually removed. These results also support the Theorem 3.1.

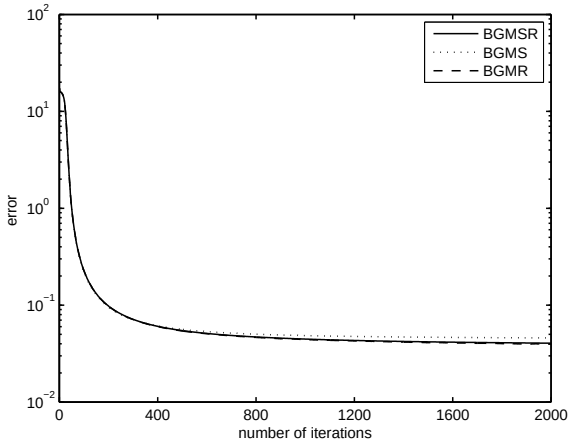


Figure 5: Parity problem: Value of the error function with different algorithms (BGMSR, BGMS, BGMR) $\eta = 0.02$, $\lambda = 0.001$.

### *4.2. Sonar problem*

The Sonar benchmark is a well-known classification problem. The data set is publicly available from the UC Irvine Machine Learning Repository at

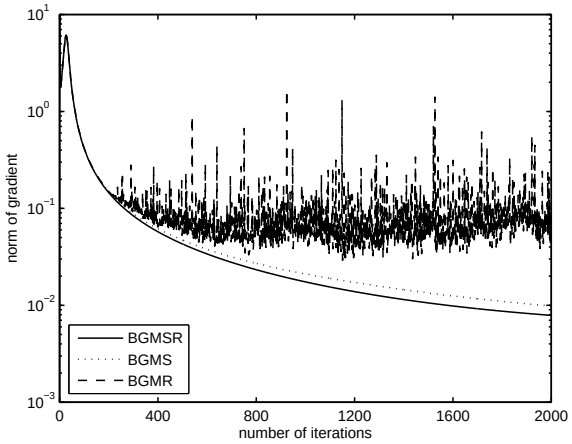


Figure 6: Parity problem: Norm of gradient with different algorithms (BGMSR, BGMS, BGMR) $\eta = 0.02$, $\lambda = 0.001$.

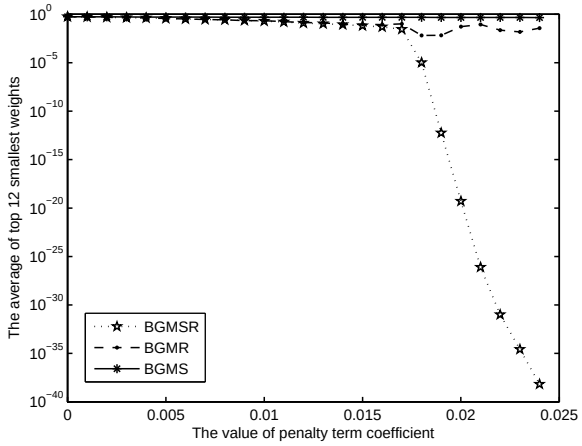


Figure 7: XOR problem: The average of top 12 smallest weights with different algorithms (BGMSR, BGMR, BGMS) for $\eta = 0.03$, $\lambda < 0.025$.

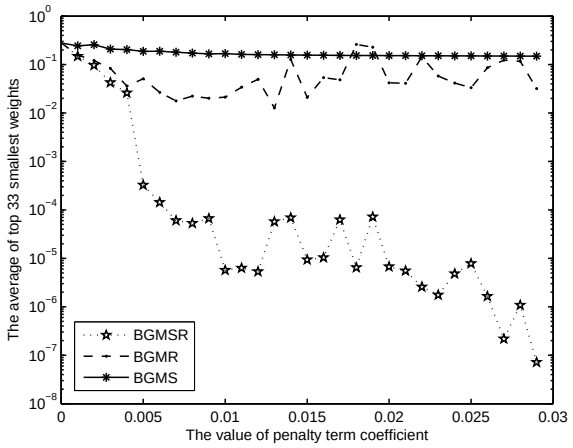


Figure 8: Parity problem: The average of top 33 smallest weights with different algorithms (BGMSR, BGMR, BGMS) for $\eta = 0.1$, $\lambda < 0.03$.

http://archive.ics.uci.edu/ml/datasets.html. The task is to classify reflected sonar signals in two categories (metal cylinders and rocks). The related data set comprises 208 input vectors, each with 60 components.

In this example, one hundred and eight samples are stochastically selected as training set, one hundred samples are selected as test set for the three algorithms. The initial learning rate $\eta$ is 0.005 and the penalty parameter $\lambda$ is 0.001, respectively. The termination criteria for the three networks are the maximum number of epochs is set to 8000, and the training is considered to be successful whenever Fahlman's "40-20-40" criterion is satisfied [26]. it means that values in the lowest 40% of the output range are treated as logical zero, values in the highest 40% are treated as logical one, and values in the middle 20% are treated as indeterminate and therefore considered as incorrect. Table.1 show the compared results, this demonstrates that BGMSR has better generalization performance than BGMR and BGMS.

Table 1: Performance comparison for BGMS, BGMR and BGMRS

| Learning algorithms | Accuracy(%) training set | Accuracy(%) testing set |
|---|---|---|
| *BGMS* | 91.20 | 87.32 |
| *BGMR* | 90.18 | 85.95 |
| *BGMRS* | 94.05 | 90.12 |

## 5. Conclusions

In this paper, the smoothing $L_{1/2}$ regularization term is introduced into the batch gradient learning algorithm

to prune FNN, the weak and strong convergence for three-layer BP neural networks is proved. In comparison to BGMR and BGMS, the pruning results in this study is a new extension of regularization approaches.

## 6. Appendix

**Lemma 6.1.** [27, Lemma 1] Suppose that $F : R^Q \to R$ is continuous and differentiable on a compact set $D \subset R^Q$, and that $\bar{\Omega} = \{x \in D | \frac{\partial F(x)}{\partial x} = 0\}$ contains only finite number of points. If a sequence $\{x^k\} \subset D$ satisfies

$$\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0, \quad \lim_{k\to\infty} \|\frac{\partial F(x^k)}{\partial x}\| = 0,$$

then, there exists $x^* \in \bar{\Omega}$ such that $\lim_{k\to\infty} x_k = x^*$.

The proof of Theorem 3.1 is divided into four parts dealing with with Statements (i), (ii), (iii) and (iv), respectively.

*Proof to (i) of Theorem 3.1.* For convenience, we use the following notations:

$$\begin{aligned} G^{n,j} &= G(V^n \xi^j) \\ \psi^{n,j} &= G^{n+1,j} - G^{n,j} \\ \sigma^n &= \sum_{i=1}^{q}\sum_{k=0}^{p} (\Delta w_{ik}^n)^2 \end{aligned} \tag{6.1}$$

Then, by the error function (2.8) we have

$$E(W^{n+1}) = \sum_{j=1}^{J} g_j(w_0^{n+1} \cdot G(V^{n+1}\xi^j)) + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} f(w_{ik}^{n+1})^{\frac{1}{2}} \tag{6.2}$$

$$E(W^n) = \sum_{j=1}^{J} g_j(w_0^{n} \cdot G(V^{n}\xi^j)) + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} f(w_{ik}^{n})^{\frac{1}{2}} \tag{6.3}$$

Applying the Taylor mean value theorem with Lagrange remainder, we have

$$\begin{aligned} & E(W^{n+1}) - E(W^n) \\ &= \sum_{j=1}^{J} (g_j(w_0^{n+1} \cdot G^{n+1,j}) - g_j(w_0^n \cdot G^{n,j})) \\ &\quad + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} (f(w_{ik}^{n+1})^{\frac{1}{2}} - f(w_{ik}^n)^{\frac{1}{2}}) \\ &= \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) G^{n,j} \Delta w_0^n \\ &\quad + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} (f(w_{ik}^{n+1})^{\frac{1}{2}} - f(w_{ik}^n)^{\frac{1}{2}}) \\ &\quad + \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_0^n \psi^{n,j} \\ &\quad + \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \Delta w_0^n \psi^{n,j} + \delta_1 \end{aligned} \tag{6.4}$$

where $\delta_1 = \frac{1}{2}\sum_{j=1}^{J} g_j''(s_{n,j})(w_0^{n+1} \cdot G^{n+1,j} - w_0^n \cdot G^{n,j})^2$, and $s_{n,j} \in R$ is a constant between $w_0^n \cdot G^{n,j}$ and $w_0^{n+1} \cdot G^{n+1,j}$.

By the Taylor mean value theorem with Lagrange remainder and the equation (2.14), we get

$$\begin{aligned} & \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_0^n \psi^{n,j} \\ &= \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \sum_{i=1}^{q} w_{i0}^n (g(w_i^{n+1} \cdot \xi^j) - g(w_i^n \cdot \xi^j)) \\ &= \sum_{i=1}^{q}\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{i0}^n g'(w_i^n \cdot \xi^j) \cdot \Delta w_i^n \cdot \xi^j \\ &\quad + \frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{i0}^n g''(t_{i,j,n})(\Delta w_i^n \cdot \xi^j)^2 \\ &= -\frac{1}{\eta}\sum_{k=1}^{p}\sum_{i=1}^{q} (\Delta w_{ik}^n)^2 - \lambda \sum_{k=1}^{p}\sum_{i=1}^{q} \frac{f'(w_{ik}^n) \cdot \Delta w_{ik}^n}{2 f(w_{ik}^n)^{\frac{1}{2}}} \\ &\quad + \frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{i0}^n g''(t_{i,j,n})(\Delta w_i^n \cdot \xi^j)^2 \end{aligned} \tag{6.5}$$

where $t_{i,j,n} \in R$ is between $w_i^n \cdot \xi^j$ and $w_i^{n+1} \cdot \xi^j$.

Noting the equation (2.13), we have

$$\begin{aligned}
&\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) G^{n,j} \Delta w_0^n \\
&= \sum_{i=1}^{q} \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) g(w_i^n \xi^j) \cdot \Delta w_{i0}^n \qquad (6.6) \\
&= -\frac{1}{\eta} \sum_{i=1}^{q} (\Delta w_{i0}^n)^2 - \lambda \sum_{i=1}^{q} \frac{f'(w_{i0}^n) \cdot \Delta w_{i0}^n}{2 f(w_{i0}^n)^{\frac{1}{2}}}
\end{aligned}$$

Substituting (6.5) and (6.6) into (6.4) and using the Lagrange mean value theorem for $f(x)$, we have

$$\begin{aligned}
&E(W^{n+1}) - E(W^n) \\
&= -\frac{1}{\eta} \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2 - \lambda \sum_{k=0}^{p} \sum_{i=1}^{q} \frac{f'(w_{ik}^n) \cdot \Delta w_{ik}^n}{2 f(w_{ik}^n)^{\frac{1}{2}}} \\
&\quad + \lambda \sum_{i=1}^{q} \sum_{k=0}^{p} (f(w_{ik}^{n+1})^{\frac{1}{2}} - f(w_{ik}^n)^{\frac{1}{2}}) \\
&\quad + \frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{0i}^n g''(t_{i,j,n}) (\Delta w_i^n \cdot \xi^j)^2 \\
&\quad + \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \Delta w_0^n \psi^{n,j} + \delta_1 \\
&= -\frac{1}{\eta} \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2 + \frac{\lambda}{2} \sum_{k=0}^{p} \sum_{i=1}^{q} F''(t_{i,k,n}) (\Delta w_{ik}^n)^2 \\
&\quad + \frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{0i}^n g''(t_{i,j,n}) (\Delta w_i^n \cdot \xi^j)^2 \\
&\quad + \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \Delta w_0^n \psi^{n,j} + \delta_1 \\
&\le -(\frac{1}{\eta} - M\lambda) \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \\
&\quad + \frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{0i}^n g''(t_{i,j,n}) (\Delta w_i^n \cdot \xi^j)^2 \\
&\quad + \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \Delta w_0^n \psi^{n,j} + \delta_1
\end{aligned} \qquad (6.7)$$

where $t_{i,k,n} \in R$ is between $w_{ik}^n$ and $w_{ik}^{n+1}$, $M = \frac{\sqrt{6}}{4\sqrt{a^3}}$, and $F(x) \equiv (f(x))^{\frac{1}{2}}$. Note that

$$\begin{aligned}
F'(x) &= \frac{f'(x)}{2\sqrt{f(x)}} \\
F''(x) &= \frac{2f''(x) \cdot f(x) - [f'(x)]^2}{4[f(x)]^{\frac{3}{2}}} \\
&\le \frac{f''(x)}{2\sqrt{f(x)}} \\
&\le \frac{\sqrt{6}}{2\sqrt{a^3}}
\end{aligned}$$

By the Lagrange mean value theorem, $(A1)$ and (3.1), we have

$$\|\psi^{n,j}\|^2 = \left\| \begin{pmatrix} g'(\tilde{t}_{1,j,n})(w_1^{n+1} - w_1^n) \cdot \xi^j \\ \vdots \\ g'(\tilde{t}_{q,j,n})(w_q^{n+1} - w_q^n) \cdot \xi^j \end{pmatrix} \right\|^2 \le C_2 \sum_{i=1}^{q} \|\Delta w_i^n\|^2 \qquad (6.8)$$

and

$$\|G(x)\| \le \sqrt{q} \sup_{t \in R} |g(t)| \le C_2, \quad x \in R^q \qquad (6.9)$$

where $\tilde{t}_{i,j,n} \in R\ (1 \le i \le q)$ is between $w_i^n \cdot \xi^j$ and $w_i^{n+1} \cdot \xi^j$.

It follows from (3.1), (6.8), (6.9) and the Cauchy-Schwartz inequality that

$$\begin{aligned}
|\delta_1| &\le \frac{C_3}{2} \sum_{j=1}^{J} (\Delta w_0^n G^{n+1,j} + w_0^n \psi^{n,j})^2 \\
&\le \frac{C_3}{2} \sum_{j=1}^{J} 2(C_2^2 \|\Delta w_0^n\|^2 + C_5^2 \|\psi^{n,j}\|^2) \\
&\le C_6 \sum_{j=1}^{J} (\|\Delta w_0^n\|^2 + C_2 \sum_{i=1}^{q} \|\Delta w_i^n\|^2) \qquad (6.10) \\
&\le JC_6(1 + C_2) \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \\
&= C_7 \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2
\end{aligned}$$

where $C_6 = C_3 \max\{C_2^2, C_5^2\}$, $C_7 = JC_6(1 + C_2)$

Similarly, we have

$$\begin{aligned}
&|\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) \Delta w_0^n \psi^{n,j}| \\
&\le \frac{C_3}{2} \sum_{j=1}^{J} (\|\Delta w_0^n\|^2 + C_2 \sum_{i=1}^{q} \|\Delta w_i^n\|^2) \\
&\le \frac{1}{2} JC_3(1 + C_2) \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \\
&= C_8 \sum_{k=0}^{p} \sum_{i=1}^{q} (\Delta w_{ik}^n)^2
\end{aligned} \qquad (6.11)$$

and

$$
\begin{aligned}
&|\frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j}) w_{0i}^n g''(t_{i,j,n})(\Delta w_i^n \cdot \xi^j)^2| \\
&\leq \frac{1}{2} J C_3^2 C_4^2 C_5 \sum_{i=1}^{q} \|\Delta w_i^n\|^2 \qquad (6.12) \\
&= C_9 \sum_{k=0}^{p}\sum_{i=1}^{q} (\Delta w_{ik}^n)^2
\end{aligned}
$$

where $C_8 = \frac{1}{2} J C_3 (1 + C_2)$ and $C_9 = \frac{1}{2} J C_3^2 C_4^2 C_5$, and take $C_1 = C_7 + C_8 + C_9$.

From (6.7)-(6.12) and $(A3)$, we can obtain

$$
\begin{aligned}
&E(W^{n+1}) - E(W^n) \\
&\leq -(\frac{1}{\eta} - \frac{\lambda}{2} F''(t_{i,k,n})) \sum_{k=0}^{p}\sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \\
&\quad + C_1 \sum_{k=0}^{p}\sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \qquad (6.13) \\
&\leq -(\frac{1}{\eta} - M\lambda - C_1) \sum_{k=0}^{p}\sum_{i=1}^{q} (\Delta w_{ik}^n)^2 \\
&\leq 0
\end{aligned}
$$

This completes the proof to Statement (i) of Theorem 3.1.

*Proof to (ii) of Theorem 3.1.* From the conclusion of (i), we know that the nonnegative sequence $\{E(W^n)\}$ is monotone. But it is also bounded below. Hence, there must exist $E^* \geq 0$ such that $\lim_{k\to\infty} E(W^n) = E^*$. The proof to (ii) is thus completed.

*Proof to (iii) of Theorem 3.1.* It follows from Assumption $(A3)$ that $\beta > 0$. Taking $\beta = \frac{1}{\eta} - M\lambda - C_1$ and using (6.13), we get

$$
E(W^{n+1}) \leq E(W^n) - \beta\sigma^n \leq \cdots \leq E(W^0) - \beta \sum_{k=0}^{n} \sigma^k
$$

Since $E(W^{n+1}) \geq 0$, we have

$$
\beta \sum_{k=0}^{n} \sigma^k \leq E(W^0)
$$

Let $n \to \infty$, then

$$
\sum_{k=0}^{\infty} \sigma^k \leq \frac{1}{\beta} E(W^0) < \infty
$$

This results in

$$
\lim_{n\to\infty} \sigma^n = 0
$$

It follows from (2.10)-(2.14) and (6.1), we have

$$
\lim_{n\to\infty} \|\Delta W^n\| = 0, \quad \lim_{n\to\infty} \|E_W(W^n)\| = 0 \qquad (6.14)
$$

The proof to (iii) is thus completed.

*Proof to (iv) of Theorem 3.1.* Note that the error function$E(W)$ defined in (2.8) is continuous and differentiable. According to (6.14), Assumption $(A4)$ and Lemma 6.1, we can easily get the desired result, i.e., there exists a point $W^* \in \Phi_0$ such that

$$
\lim_{n\to\infty} (W^n) = W^*
$$

This completes the proof to (iv).

## References


[1] Haykin, S. (2001). Neural Networks: A Comprehensive Foundation. 2nd edition. Beijing: Tsinghua University Press, PrenticeHall, Beijing.

[2] Magoulas, G. D., Vrahatis, M. N., & Androulakis, G. S. (1999). Improving the Convergence of the Backpropagation Algorithm Using Learning Rate Adaptation Methods. Neural Computation, 11 (7), 1769-1796.

[3] Zhang, X. S. (2000). Neural Networks in Optimization. Boston: Kluwer Academic Publishers.

[4] Liu, W., & Dai, Y. H. (2001). Minimization algorithms based on supervisor and searcher cooperation. Journal of Optimization Theory and Applications, 111 (2), 359-379.

[5] Chen, T. P., Lu, W. L., & Amari, S. (2002). Global Convergence Rate of Recurrently Connected Neural Networks. Neural Computation, 14 (12), 2947-2957.

[6] Wu, W. (2003). Computation of Neural Networks, Beijing: Higher Education Press.

[7] Xu, D. P., Zhang, H. S., & Liu, L. J. (2010). Convergence Analysis of Three Classes of Split-Complex Gradient Algorithms for Complex-Valued Recurrent Neural Networks. Neural Computation, 22 (10), 2655-2677.

[8] Zhou, W. & Zurada, J. M. (2010). Competitive Layer Model of Discrete-Time Recurrent Neural Networks with LT Neurons. Neural Computation, 22(8), 2137-2160.

[9] Plaut, D. C., Nowlan, S. J., & Hinton, G. E. (1986). Experiments On learning by backpropagation. Tech. Rep, CMU-CS-86-126, Computer Science Department. Carnegie-Mellon University, Pittsburgh.

[10] Setiono, R. (1994). A penalty-function approach for pruning feedforward neural networks. Neural Networks, 9 (1), 185-204.

[11] Wu, W., Shao, H. M., & Li, Z. X. (2006). Convergence of batch BP algorithm with penalty for FNN training. Neural Information Processing, Vol.4232 (PP.562-569).

[12] Zhang, H. S., Wu, W., Liu, F., & Yao, M. C. (2009). Boundedness and Convergence of Online Gradient Method With Penalty for Feedforward Neural Networks. IEEE Transactions On Neural Networks, 20 (6), 1050-1054.

[13] Reed, R. (1993). Pruning algorithms-a survey. IEEE Transactions on Neural Networks, 4 (5), 740-747.

[14] Ishikawa, M. (1996). Structual learning with forgetting. Neural Networks, 9 (3), 509-521.

[15] Hinton, G. E. (1989). Connectionist learning procedures. Artificial Intelligence, Vol.40 (pp.185-234).

[16] Ishikawa, M. (1989). A structural learning algorithm with forgetting of link weights. Neural Networks, IJCNN., International Joint Conference on.

[17] Weigend, A. S., Rumelhart, D. E., & Huberman, B. A. (1991). Generalization by weight-elimination applied to currency exchange rate prediction. Int. Conf. on Neural Networks, Singapore, pp.2374-2379.

[18] Kong, J., & Wu, W. (2001). On line gradient methods with a punishing term for neural networks. Northeastern Mathematieal Journal, 17 (3), 371-378.

[19] Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. Second International Symposium on Information Theory. Budapest: Akademiai Kiado, pp.267-281.

[20] Mallows, C. L. (1973). Some comments on $C_p$. Technometrics, 15 (4), 661-675.

[21] Schwarz, G. (1978). Estimating the dimension of a model. The Annals of Statistics, 6 (2), 461-464.

[22] Forti, M. Nistri, P. & Quincampoix,M. (2004). Generalized neural network for nonsmooth nonlinear programming problems, IEEE Transactions on Circuits and Systems-I, 51 (9), 1741-1754.

[23] Liu, Q. S., Guo, Z. S., & Wang, J., (2012). A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization. Neural Networks, Vol.26 (pp.99-109).

[24] Liu, Q. S., & Cao, J. D. (2010). A recurrent neural network based on projection operator for extended general variational inequalities. IEEE Trans. Systems, Man, and Cybernetics-Part B, 40 (3), 928-938.

[25] Xu, Z. B., Zhang, H., Wang, Y., & Chang, X. Y. (2009). $L_{1/2}$ Regularizer, Sci China Ser F-Inf Sci, Jan. 52 (1), 1-9.

[26] Ampazis, N., & Perantonis, S. J. (2002). Two highly efficient second-order algorithms for training feedforward networks. IEEE Transactions on Neural Networks, 13 (5), 1064-1074.

[27] Wu, W., Li, L., Yang, J., & Liu, Y. (2010). A modified gradient-based neuro-fuzzy learning algorithm and its convergence. Information Sciences, 180 (9), 1630-1642.