Fan *et al. SpringerPlus* *(2016) 5:295*
DOI 10.1186/s40064-016-1931-0

SpringerPlus
a SpringerOpen Journal

RESEARCH

Open Access

Convergence of batch gradient learning with smoothing regularization and adaptive momentum for neural networks

Qinwei Fan[1,2,3*], Wei Wu[2] and Jacek M. Zurada[3,4]

*Correspondence: qinweifan@126.com
[1] School of Science, Xi'an Polytechnic University, Xi'an 710048, People's Republic of China
Full list of author information is available at the end of the article

Abstract
This paper presents new theoretical results on the backpropagation algorithm with smoothing $L_{1/2}$ regularization and adaptive momentum for feedforward neural networks with a single hidden layer, i.e., we show that the gradient of error function goes to zero and the weight sequence goes to a fixed point as n (n is iteration steps) tends to infinity, respectively. Also, our results are more general since we do not require the error function to be quadratic or uniformly convex, and neuronal activation functions are relaxed. Moreover, compared with existed algorithms, our novel algorithm can get more sparse network structure, namely it forces weights to become smaller during the training and can eventually removed after the training, which means that it can simply the network structure and lower operation time. Finally, two numerical experiments are presented to show the characteristics of the main results in detail.

Keywords: Feedforward neural networks, Adaptive momentum, Smoothing $L_{1/2}$ regularization, Convergence

Background

A multilayer perceptron network trained with a highly popular algorithm known as the error back-propagation (BP) has been dominating in the neural network literature for over two decades (Haykin 2008). BP uses two practical ways to implement the gradient method: the batch updating approach that accumulates the weight corrections over the training epoch before performing the update, while the online learning approach updates the network weights immediately after each training sample is processed (Wilson and Martinez 2003).

Note that training is usually done by iteratively updating of the weights that reduces error value, which is proportional to the negative gradient of a sum-square error (SSE) function. However, during the training of feedforward neural networks (FNN) with SSE, the weights might become very large or even unbounded. This drawback can be addressed by adding a regularization term to the error function. The extra term acts as a brute-force to drive unnecessary weights to zero to prevent the weights from taking too large values and then it can be used to remove weights that are not needed, and is also called penalty term (Haykin 2008; Wu et al. 2006; Karnin 1990; Reed 1993; Saito and Nakano 2000).

There are four main different penalty approaches for BP training: weight decay procedure (Hinton 1989), weight elimination (Weigend et al. 1991), approximate smoother procedure (Moody and Rognvaldsson 1997) and inner product penalty (Kong and Wu 2001).

In the weight decay procedure, the complexity penalty term is defined as the squared norm of the weight vector, and all weights in the multilayer perceptron are treated equally. In the weight elimination procedure, the complexity penalty represents the complexity of the network as function of weight magnitudes relative to a pre-assigned parameter (Reed 1993).

In approximate smoother procedure, this penalty term is used for a multilayer perceptron with a single hidden layer and a single neuron in the output layer. Compared with the earlier methods, it does two things. First, it distinguishes between the roles of weights in the hidden layer and those in the output layer. Second, it captures the interactions between these two sets of weights, however, it is much more demanding in computational complexity than weight decay or weight elimination methods. In Kong and Wu (2001) the inner-product form is proposed and its efficiency in general performance of controlling the weights is demonstrated. Convergence of the gradient method for the FNN has been considered by Zhang et al. (2015, 2009), Wang et al. (2012) and Shao and Zheng (2011).

The convergence of the gradient method with momentum is considered in Bhaya and Kaszkurewicz (2004), Torii and Hagan (2002), Zhang et al. (2006), in Bhaya and Kaszkurewicz (2004) and Torii and Hagan (2002) under the restriction that the error function is quadratic. Inspired by Chan and Fallside (1987), Zhang et al. (2006) considers the convergence of a gradient algorithm with adaptive momentum, without assuming the error function to be quadratic as in the existing results. However, in Zhang et al. (2006), the strong convergence result is based on the assumption that the error function is uniformly convex, which still seems a little intense.

The size of a hidden layer is one of the most important considerations when dealing with real life tasks using FNN. However, the existing pruning methods may not prune the unnecessary weights efficiently, so how to efficiently simplify the network structure becomes our main task.

Recently, considerable attention has been paid to the sparsity problems and a class of regularization methods was proposed which take the following form:

$$\min \left\{ \frac{1}{n} \sum_{i=1}^{n} l(y_i, h(x_i)) + \lambda \|h\|_k \right\} \tag{1}$$

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

For $k = 0$, (1) becomes L_0 regularization and can be understood as a penalized least squares with penalty $\|h\|_0$, which yields the most sparse solutions, but for large data analysis it faces the problem of combinatory optimization (Davis 1994; Natarajan 1995). In order to deal with such difficulty, Tibshirani (1996) proposed L_1 regularization where $k = 1$ and $\|h\|_1$ is the L_1 norm of n dimensional Euclidean space R^n, which just needs to solve a quadratic programming problem but is less sparse than the L_0 regularization. At the same time Donoho (1995, 2005) proved that under some conditions the solutions of the L_0 regularizer are equivalent to those of the L_1, so the hard NP optimization problem can be avoided in the L_1 regularizer. In order to find a new regularizer which is more sparse than the L_1 regularizer while it is still easier to be solved than the L_0 regularizer, in Xu et al. (2010) a modified $L_{1/2}$ regularizer is proposed of the following form:

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

where λ is the tuning parameter. As shown in Xu et al. (2010), $L_{1/2}$ regularizer has a non-convex penalty and possesses many promising properties such as unbiasedness, sparsity, oracle properties and can be taken as a representative of the L_r $(0 < r < 1)$ regularizer. Recently, we develop a novel method to prune FNNs through modify the usual $L_{1/2}$ regularization term by smoothing technique. The new algorithm not only removes the oscillation of the gradient value, but also get better pruning, namely the final weights to be removed are smaller than those produced through the usual $L_{1/2}$ regularization (Wu et al. 2014; Fan et al. 2014).

The focus of this paper is on extension of $L_{1/2}$ regularization beyond its basic concept though its augmentation with a momentum term. Also, there are some other applications of FNNs for optimization problems, such as the generalized gradient and recurrent neural network methods shown as Liu et al. (2012) and Liu and Cao (2010)

It is well known that a general drawback of gradient based BP learning process is its slow convergence. To accelerate learning, a momentum term is often added (Haykin 2001; Chan and Fallside 1987; Qiu et al. 1992; Istook and Martinez 2002). By adding momentum to the update formula, the current weight increment is a linear combination of the gradient of the error function and the previous weight increment. As a result, the updates respond not only to the local gradient but also to recent gradient in the error function. Selected reports discuss the NN training with momentum term in the literature (Torii and Hagan 2002; Perantonis and Karras 1995; Qian 1999).

As demonstrated in Torii and Hagan (2002), there always exists a momentum coefficient that will stabilize the steepest descent algorithm, regardless of the value of the learning rate (we will define it below). In addition, it shows how the value of the momentum coefficient changes the convergence properties. Momentum acceleration, its performance in terms of learning speed and scalability properties is evaluated and found superior to the performance of reputedly fast variants of the BP algorithm in several benchmark training tasks in Perantonis and Karras (1995). Qian (1999) shows that in the limit of continuous time, the momentum parameter is analogous to the mass of Newtonian particles that move through a viscous medium in a conservative force field.

In this paper, a modified batch gradient method with smoothing $L_{1/2}$ regularization penalty and adaptive momentum algorithm (BGSAM) is proposed. It damps oscillations present in the $L_{1/2}$ regularization and in the adaptive momentum algorithm (BGAM). In addition, without the requirement that the error function is quadratic or uniformly convex, we present a comprehensive study of the weak and strong convergence for BGSAM which offers an effective improvement in real life application.

The rest of this paper is arranged as follows. The algorithm BGSAM is described in "Batch gradient method with smoothing $L_{1/2}$ regularization and adaptive momentum (BGSAM)" section. In "Convergence results" section, the convergence results of BGSAM are presented, and the detailed proofs of the main results are stated in the "Appendix". The performance of BGSAM is compared to BGAM and the experimental results shown in "Numerical experiments" section. Concluding remarks are in "Conclusions" section.

Batch gradient method with smoothing $L_{1/2}$ regularization and adaptive momentum (BGSAM)

Batch gradient method with $L_{1/2}$ regularization and adaptive momentum (BGAM)

Here and below, some definitions and notations used in e.g. Wu et al. (2006), Shao and Zheng (2011), and Wu et al. (2006), Shao and Zheng (2011) have been re-defined and used without repeatedly citing the references. We consider a FNN with three layers, and we denote the numbers of neurons of the input, hidden and output layers by p, q and 1, respectively. Suppose that $\{\xi^j, O^j\}_{j=1}^J \subset R^p \times R$ is the given set of J training samples. Let $w_0 = (w_{10}, w_{20}, \ldots, w_{q0})^T \in R^q$ be the weight vector between the hidden units and the output unit, and $w_i = (w_{i1}, w_{i2}, \ldots, w_{ip})^T \in R^p$ be the weight vector between the input units and the hidden unit i $(i = 1, 2, \ldots, q)$. To simplify the presentation, we combine the weight vectors, and write $W = (w_0^T, w_1^T, \ldots, w_q^T)^T \in R^{q+pq}$ and we define a matrix $V = (w_1, w_2, \ldots, w_q)^T \in R^{q\times p}$. We also define a vector function $G : R^q \to R^q$, for $x = (x_1, x_2, \ldots, x_q)^T \in R^q$

$$G(x) = (g(x_1), g(x_2), \ldots, g(x_q))^T. \tag{3}$$

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. Then for each input $\xi \in R^p$, the actual output vector of the hidden layer is $G(V\xi)$ and the final output of the network is

$$g(w_0 \cdot G(V\xi)). \tag{4}$$

For a fixed W, the output error function with the $L_{1/2}$ regularization penalty term is

$$\begin{aligned} E(W) &= \frac{1}{2}\sum_{j=1}^{J}(O^j - g(w_0 \cdot G(V\xi^j)))^2 + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p}|w_{ik}|^{\frac{1}{2}} \\ &= \sum_{j=1}^{J} g_j(w_0 \cdot G(V\xi^j)) + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p}|w_{ik}|^{\frac{1}{2}} \end{aligned} \tag{5}$$

where $g_j(t) := \frac{1}{2}(O^j - g(t))^2$, $j = 1, 2, \ldots, J$, $t \in R$, $\lambda > 0$ is the penalty coefficient, and $|\cdot|$ denotes the absolute value. The gradient of the error function is

$$E_W(W) = (E_{w_0}^T(W), E_{w_1}^T(W), \ldots, E_{w_q}^T(W))^T \tag{6}$$

where

$$\begin{aligned}
&E_{w_0}^T(W) = (E_{w_{10}}(W), E_{w_{20}}(W), \ldots, E_{w_{q0}}(W))\\
&E_{w_1}^T(W) = (E_{w_{11}}(W), E_{w_{12}}(W), \ldots, E_{w_{1p}}(W))\\
&E_{w_2}^T(W) = (E_{w_{21}}(W), E_{w_{22}}(W), \ldots, E_{w_{2p}}(W))\\
&\cdots\\
&E_{w_q}^T(W) = (E_{w_{q1}}(W), E_{w_{q2}}(W), \ldots, E_{w_{qp}}(W))
\end{aligned}$$

The gradient of the error function with respect to w_{i0} and w_{ik} are, respectively, given by

$$E_{w_{i0}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j))g(w_i\xi^j) + \frac{\lambda sgn(w_{i0})}{2|w_{i0}|^{\frac{1}{2}}} \tag{7}$$

$$E_{w_{ik}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j))w_{i0}g'(w_i \cdot \xi^j)\xi_k^j + \frac{\lambda sgn(w_{ik})}{2|w_{ik}|^{\frac{1}{2}}} \tag{8}$$

where $i = 1, 2, \ldots, q$, and $k = 1, 2, \ldots, p$.

The detailed BGAM algorithm is presented as follows. We denote $W^{n+1} = W^n + \Delta W^n$, $n = 0, 1, 2, \ldots$, starting from an arbitrary initial value W^0 and W^1, and the weights $\{W^n\}$ are updated iteratively by

$$\Delta W^n = -\eta E_W(W^n) + \alpha_W^n \Delta W^{n-1}, \quad n = 0, 1, 2, \ldots \tag{9}$$

The learning rate is assumed constant and satisfies $\eta > 0$, and $\alpha_W^n = (\alpha_{w_0}^n, \alpha_{w_1}^n, \ldots, \alpha_{w_q}^n)$ is the momentum coefficient vector of the n-th training. It consists of coefficients $\alpha_{w_i}^n$ for each $\Delta w_{ik}^n (i = 1, 2, \ldots, q,\ k = 1, 2, \ldots, p)$, and $\alpha_{w_0}^n$ for each $\Delta w_{i0}^n (i = 1, 2, \ldots, q)$. Similar to Shao and Zheng (2011), for every $\alpha_{w_i}^n$, after each training epoch it is chosen as

$$\alpha_{w_i}^n = \begin{cases} \alpha \cdot \frac{-\eta E_{w_i}(W^n)\cdot \Delta w_i^{n-1}}{\|\Delta w_i^{n-1}\|^2}, & if\ E_{w_i}(W^n) \cdot \Delta w_i^{n-1} < 0 \\ 0, & otherwise \end{cases} \tag{10}$$

where $\alpha \in (0, 1)$ is the momentum factor. Compared with the traditional algorithm, the BGAM has better pruning performance, but we notice that this usual $L_{1/2}$ regularization term used in this part involves in absolute values and it is not differentiable at the origin, which will cause difficulty in the convergence analysis. More importantly, it causes oscillations of the error function and the norm of gradient. In order to overcome these drawbacks we improved the BGAM algorithm as follows:

Smoothing $L_{1/2}$ regularization and adaptive momentum (BGSAM)

This section introduces a modified algorithm with smoothing $L_{1/2}$ regularization and adaptive momentum term. The network structure is the same as the description in part of last subsection (BGAM). We modify the usual $L_{1/2}$ regularization term at the origin (i.e. we replace the absolute values of the weights by a smooth function in a neighborhood of the origin). Then we use a smooth function $f(x)$ to approximate $|x|$. We get the following error function with a smoothing $L_{1/2}$ regularization penalty term:

$$\begin{aligned} E(W) &= \frac{1}{2}\sum_{j=1}^{J}(O^j - g(w_0 \cdot G(V\xi^j)))^2 + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} f(w_{ik})^{\frac{1}{2}} \\ &= \sum_{j=1}^{J} g_j(w_0 \cdot G(V\xi^j)) + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p} f(w_{ik})^{\frac{1}{2}} \end{aligned} \tag{11}$$

where $g_j(t) := \frac{1}{2}(O^j - g(t))^2$, $j = 1, 2, \ldots, J$, $t \in R$, $\lambda > 0$ is the penalty coefficient. Here, by smoothing we mean that, in a neighborhood of the origin, we replace the absolute values of the weights by a smooth function of the weights. For definiteness and simplicity, we choose *f*(*x*) as a piecewise polynomial function such as:

$$f(x) = \begin{cases} |x|, & if\ |x| \geq a \\ -\frac{1}{8a^3}x^4 + \frac{3}{4a}x^2 + \frac{3}{8}a, & if\ |x| < a \end{cases} \tag{12}$$

where a is a small positive constant. and $|\cdot|$ denotes the absolute value. Then, from the definition of *f*(*x*) immediately yields

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

The gradient of the error function with respect to W as in (6), and the gradients of the error function with respect to w_{i0} and w_{ik} are then as follows:

$$E_{w_{i0}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j)) g(w_i \cdot \xi^j) + \lambda \frac{f'(w_{i0})}{2f(w_{i0})^{\frac{1}{2}}} \tag{13}$$

$$E_{w_{ik}}(W) = \sum_{j=1}^{J} g_j'(w_0 \cdot G(V\xi^j)) w_{i0} g'(w_i \cdot \xi^j)\xi_k^j + \lambda \frac{f'(w_{ik})}{2f(w_{ik})^{\frac{1}{2}}} \tag{14}$$

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

For BGSAM algorithm, we denote $W^{n+1} = W^n + \Delta W^n$, $n = 0, 1, 2, \ldots$. Starting with an initial value W^0 and W^1, the weights $\{W^n\}$ are updated iteratively by

$$\Delta W^n = -\eta E_W(W^n) + \alpha_W^n \Delta W^{n-1}, \quad n = 0, 1, 2, \ldots \tag{15}$$

Here the learning rate η, the momentum coefficient vector of the n-th training α_W^n and other coefficients are the same as the description of algorithm BGAM. For each $\alpha_{w_i}^n$, after each training epoch it is chosen as (10).

Convergence results

The following assumptions are needed to introduce the relating convergence theorems of BGSAM.

($A1$) $|g(t)|, |g'(t)|, |g''(t)|$ are uniformly bounded for $t \in R$.
($A2$) There exists a bounded region $\Omega \subset R^n$ such that $\{w_0^n\}_{n=0}^{\infty} \subset \Omega$.
($A3$) $0 < \eta < \frac{1}{(M\lambda + C_1)(1+\alpha)^2}$, where $M = \frac{\sqrt{6}}{4\sqrt{a^3}}$ and C_1 is a constant defined in (16) below.

Assume conditions $(A1)$–$(A2)$ is valid. Then there are some positive constants C_1–C_5 such that

$$
\begin{aligned}
&C_1 = J(1 + C_2)C_3 \max\{C_2^2, C_5^2\} + \frac{1}{2}J(1 + C_2)C_3 + \frac{1}{2}JC_3^2C_4^2C_5,\\
&C_2 = \max\left\{\sqrt{q}C_3, (C_3C_4)^2\right\},\\
&C_3 = \max\left\{\sup_{t\in R}|g(t)|, \sup_{t\in R}|g'(t)|, \sup_{t\in R}|g''(t)|, \sup_{t\in R, 1\le j\le J}|g_j'(t)|, \sup_{t\in R, 1\le j\le J}|g_j''(t)|\right\},\\
&C_4 = \max_{1\le j\le J}\|\xi^j\|, \quad C_5 = \sup_{n\in N}\|w_0^n\|.
\end{aligned}
\tag{16}
$$

Theorem 1 *If assumptions* $(A1)-(A3)$ *are valid for any arbitrary initial value* W^0 *and* W^1*, the error function be defined by (1), and let the learning sequence 1*$\{W^n\}$ *be generated by the iteration algorithm (15), then we have the following convergence*

(i) $\lim_{n\to\infty} E_W(W^n) = 0$*. Moreover, (A4) if there exists a compact set* Φ *such that* $W^n \in \Phi$ *and the set* $\Phi_0 = \{W \in \Phi : E_W(W) = 0\}$ *contains finite points also holds, then we have the following convergence*

(ii) $\lim_{n\to\infty}(W^n) = W^*$, *where* $W^* \in \Phi_0$.

Numerical experiments

This section presents the simulations that verify the performance of BGAM and BGSAM. Our theoretical results are experimentally verified with the 3-bit parity problem, which is a typical benchmark problem in area of the neural networks.

The two algorithms (BGAM and BGSAM) are implemented by the networks with the structure 5-7-1 (input $p = 5$, hidden $q = 7$ and output nodes, see Fig. 1). Each of the two algorithms are carried out fifty trials for 3-bit parity problem and then take the mean values, and the termination criterion is that the error is <1e−6 or 3000 iterations. For the network with linear output, we set the transfer function for hidden neurons to be *tansig*$(\cdot)$ and that for output layer to be $g(t) = t$. For the network with nonlinear output, the transfer functions for both hidden and output neurons are *tansig*$(\cdot)$. The inputs and the ideal outputs are shown in Table 1.

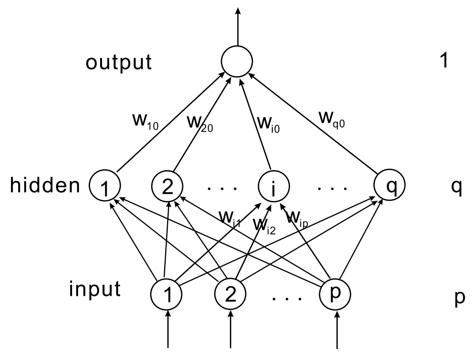

Fig. 1 Feedforward neural network with one hidden layer and one output

Table 1 3-bit parity problem

Input				Output	Input				Output
1	1	1	−1	1	1	−1	−1	−1	1
1	1	−1	−1	0	−1	1	1	−1	0
1	−1	1	−1	0	−1	−1	1	−1	1
−1	−1	−1	−1	0	−1	1	−1	−1	1

The performance results of BGAM and BGSAM are shown in the following figures. Figures 2, 3 and 4 present the comparison results for learning rate η, penalty parameter λ and momentum term α with 0.01, 0.0006 and 0.03, respectively.

From Figs. 2 and 3, it can be seen that the error function decreases monotonically and the norm of the gradient of the error function approaches zero as depicted by the convergence theorem, respectively. Also Fig. 3 show us that our modified algorithm

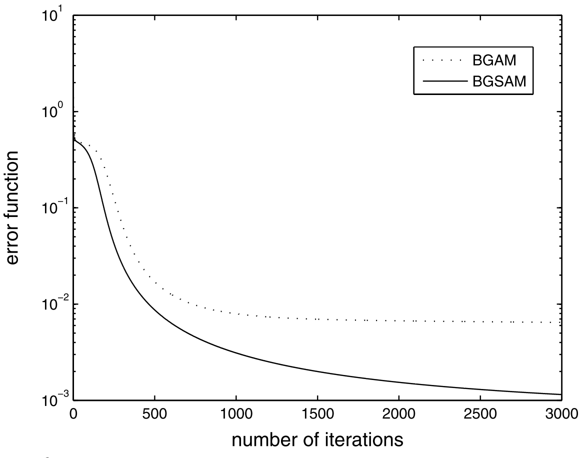

Fig. 2 The curve of error function

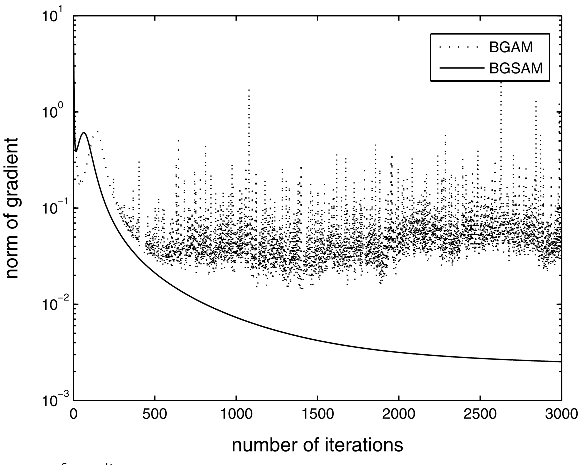

Fig. 3 The curve of norm of gradient

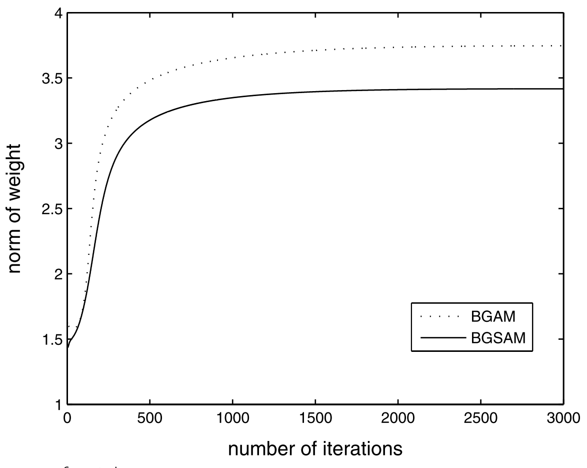

Fig. 4 The curve of norm of weight

overcomes the drawbacks of numerical oscillations, i.e., for BGSAM the norm of gradient curve is much smoother than BGAM. The reason as the following: Since the derivative of $|x|$ jumps from -1 to $+1$ near the $x = 0$, the learning process of *BGAM* will oscillate when a weight w_{ik} is close to zero, whic prevents it from getting further closer to zero. And on the contrary, the derivative of $f(x)$, which is a smooth approximation of $|x|$, is smooth and equal to zero at the origin, and will not cause any oscillation in the learning process when w_{ik} is close to zero. In the meantime, it can be seen that BGSAM convergence faster than BGAM. Fig. 4 demonstrates that the effectiveness of the algorithm BGSAM in controlling the magnitude of weights is better than BGAM.

Conclusions

In this paper, the smoothing $L_{1/2}$ regularization term with adaptive momentum is introduced into the batch gradient learning algorithm to prune FNN. First, it removes the oscillation of the gradient value. Second, the convergence results for three-layer FNN are proved under certain relaxed conditions. Third, the algorithm is applied to a 3-bit parity problem and the related results are supplied to support the theoretical findings above. Finally, this new algorithm will also effective for other types neural networks or big data processing.

Authors' contributions

This work was carried out by the three authors, in collaboration. All authors read and approved the final manuscript.

Author details

[1] School of Science, Xi'an Polytechnic University, Xi'an 710048, People's Republic of China. [2] School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, People's Republic of China. [3] Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY 40292, USA. [4] Spoleczna Akademia Nauk, 90-011 Lodz, Poland.

Acknowledgements

This work was supported by National Science Foundation of China (Nos. 11201051, 11501431, 11302158) and National Science Foundation for Tian yuan of China (No. 11426167), and the Science Plan Foundation of the Education Bureau of Shaanxi Province. The authors are grateful to the anonymous reviewers and editors for their profitable comments and suggestions, which greatly improves this paper.

Competing interests

The authors declare that they have no competing interests.

Appendix

In the following section three useful lemmas are given for convergence analysis for BGSAM algorithm. For the sake of description, we introduce the following notations:

$$\begin{aligned} G^{n,j} &= G(V^n \xi^j) \\ \psi^{n,j} &= G^{n+1,j} - G^{n,j} \\ \Delta w_i^n &= w_i^{n+1} - w_i^n \quad \textit{for } i = 1, 2, \ldots q \\ \Delta w_0^n &= w_0^{n+1} - w_0^n \end{aligned} \tag{17}$$

Using the error function (11) we have

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

Lemma 1 (Wu et al. 2010, 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*

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

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

The above lemma is crucial for the strong convergence analysis, and it basically follows the same proof as Lemma 1 in Wu et al. (2010). So its proof is omitted.

Now, we give the **proofs of Theorem** 1 through the following 4 steps.

Step 1. We show the following inequalities holds

$$\|G(x)\| \le \sqrt{q} \sup_{t\in R} |g(t)| \le C_2, \quad x \in R^q \tag{18}$$

$$\begin{aligned} \left\|\psi^{n,j}\right\|^2 &\le C_2 \sum_{i=1}^{q} \left\|\Delta w_i^n\right\|^2 \\ &\le C_2 \eta^2 (1+\alpha)^2 \sum_{i=1}^{q} \left\|E_{w_i}(W^n)\right\|^2 \quad j = 1, 2, \ldots, J. \end{aligned} \tag{19}$$

Proof With the assumption ($A1$), (16) and the definition of $G(x)$, it is easy to show that there exists a positive constant C_2 such that (18) holds.

By the Lagrange mean value theorem, ($A1$) and (16), we obtain that

$$\left\|\psi^{n,j}\right\|^2 = \left\| \begin{pmatrix} g'(\tilde{t}_{1,j,n})(w_1^{n+1} - w_1^n) \cdot \xi^j \\ g'(\tilde{t}_{2,j,n})(w_2^{n+1} - w_2^n) \cdot \xi^j \\ \vdots \\ g'(\tilde{t}_{q,j,n})(w_q^{n+1} - w_q^n) \cdot \xi^j \end{pmatrix} \right\|^2 \le C_2 \sum_{i=1}^{q} \left\|\Delta w_i^n\right\|^2 \tag{20}$$

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

Furthermore, in terms of (9) and (10) we can show that

$$\begin{aligned}\|\Delta w_i^n\| &= \left\| -\eta E_{w_i}(W^n) + \alpha_{w_i}^n \cdot \Delta w_i^{n-1} \right\| \\ &\le \eta(1+\alpha)\|E_{w_i}(W^n)\|\end{aligned} \tag{21}$$

On the basis of the above inequalities (20) and (21) we immediately have (19). □

Step 2. We show the following monotonicity of the sequence $\{E(W^n)\}$

$$E(W^{n+1}) \le E(W^n), \quad n = 0, 1, 2, \ldots \tag{22}$$

Proof According to the definition of w_0^n and $\psi^{n,j}$, we get

$$w_0^n \psi^{n,j} = \sum_{i=1}^{q} w_{i0}^n \left(g\left(w_i^{n+1} \cdot \xi^j\right) - g\left(w_i^n \cdot \xi^j\right)\right) \tag{23}$$

By the Taylor mean value theorem with Lagrange remainder to $g(t)$ at the point $w_i^n \cdot \xi^j$, we obtain

$$\begin{aligned}&g\left(w_i^{n+1} \cdot \xi^j\right) - g\left(w_i^n \cdot \xi^j\right) \\ &\quad = g'\left(w_i^n \cdot \xi^j\right) \cdot \Delta w_i^n \cdot \xi^j + \frac{1}{2} g''\left(t_{i,j,n}\right)\left(\Delta w_i^n \cdot \xi^j\right)^2\end{aligned} \tag{24}$$

where each $t_{i,j,n}$ lies on the segment between $w_i^{n+1} \cdot \xi^j$ and $w_i^n \cdot \xi^j$, $i = 1, 2, \ldots, q; j = 1, 2, \ldots, J$.

Together with (10) and (14), we get

$$\begin{aligned}&\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) w_0^n \psi^{n,j} \\ &\quad = \sum_{i=1}^{q}\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) w_{i0}^n g'\left(w_i^n \cdot \xi^j\right) \cdot \Delta w_i^n \cdot \xi^j + \delta_1 \\ &\quad = \sum_{k=1}^{p}\sum_{i=1}^{q}\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) w_{i0}^n g'\left(w_i^n \cdot \xi^j\right) \cdot \Delta w_{ik}^n \cdot \xi_k^j + \delta_1 \\ &\quad = \sum_{k=1}^{p}\sum_{i=1}^{q} E_{w_{ik}}(W^n) \cdot \Delta w_{ik}^n - \lambda \sum_{k=1}^{p}\sum_{i=1}^{q} \frac{f\left(w_{ik}^n\right) \cdot \Delta w_{ik}^n}{2 f\left(w_{ik}^n\right)^{\frac{1}{2}}} + \delta_1 \\ &\quad = \sum_{i=1}^{q} E_{w_i}(W^n) \cdot \Delta w_i^n - \lambda \sum_{k=1}^{p}\sum_{i=1}^{q} \frac{f\left(w_{ik}^n\right) \cdot \Delta w_{ik}^n}{2 f\left(w_{ik}^n\right)^{\frac{1}{2}}} + \delta_1 \\ &\quad = \sum_{i=1}^{q} E_{w_i}(W^n)\left(-\eta E_{w_i}(W^n) + \alpha_{w_i}^n \cdot \Delta w_i^{n-1}\right) \\ &\qquad - \lambda \sum_{k=1}^{p}\sum_{i=1}^{q} \frac{f\left(w_{ik}^n\right) \cdot \Delta w_{ik}^n}{2 f\left(w_{ik}^n\right)^{\frac{1}{2}}} + \delta_1 \\ &\quad = -\eta \sum_{i=1}^{q} \left\|E_{w_i}(W^n)\right\|^2 + \sum_{i=1}^{q} \alpha_{w_i}^n \cdot E_{w_i}(W^n)) \cdot \Delta w_i^{n-1} \\ &\qquad - \lambda \sum_{k=1}^{p}\sum_{i=1}^{q} \frac{f(w_{ik}^n) \cdot \Delta w_{ik}^n}{2 f(w_{ik}^n)^{\frac{1}{2}}} + \delta_1\end{aligned} \tag{25}$$

where $\delta_1 = \frac{1}{2}\sum_{i=1}^{q}\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) w_{i0}^n g''\left(t_{i,j,n}\right)\left(\Delta w_i^n \cdot \xi^j\right)^2$, and $t_{i,j,n} \in R$ is between $w_i^n \cdot \xi^j$ and $w_i^{n+1} \cdot \xi^j$.

Using (10) and (13), we obtain

$$\begin{aligned}
&\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) G^{n,j} \Delta w_0^n \\
&\quad= \sum_{i=1}^{q}\sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) g\left(w_i^n \xi^j\right) \cdot \Delta w_{i0}^n \\
&\quad= \sum_{i=1}^{q}\left(E_{w_{i0}}(W^n) - \lambda \frac{f'\left(w_{i0}^n\right)}{2f\left(w_{i0}^n\right)^{\frac{1}{2}}}\right) \cdot \Delta w_{i0}^n \\
&\quad= E_{w_0}(W^n)\Delta w_0^n - \lambda \sum_{i=1}^{q} \frac{f'\left(w_{i0}^n\right) \cdot \Delta w_{i0}^n}{2f\left(w_{i0}^n\right)^{\frac{1}{2}}} \\
&\quad= E_{w_0}(W^n)\left(-\eta E_{w_0}(W^n) + \alpha_{w_0}^n \cdot \Delta w_0^{n-1}\right) \\
&\qquad - \lambda \sum_{i=1}^{q} \frac{f'\left(w_{i0}^n\right) \cdot \Delta w_{i0}^n}{2f\left(w_{i0}^n\right)^{\frac{1}{2}}} \\
&\quad= -\eta \left\|E_{w_0}(W^n)\right\|^2 + \alpha_{w_0}^n E_{w_0}(W^n)\Delta w_0^{n-1} \\
&\qquad - \lambda \sum_{i=1}^{q} \frac{f'\left(w_{i0}^n\right) \cdot \Delta w_{i0}^n}{2f\left(w_{i0}^n\right)^{\frac{1}{2}}}
\end{aligned} \tag{26}$$

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

$$\begin{aligned}
&E(W^{n+1}) - E(W^n) \\
&\quad= \sum_{j=1}^{J}\left[g_j'\left(w_0^n \cdot G^{n,j}\right) \cdot (w_0^{n+1} \cdot G^{n+1,j} - w_0^n \cdot G^{n,j})\right] \\
&\qquad + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p}\left(f\left(w_{ik}^{n+1}\right)^{\frac{1}{2}} - f\left(w_{ik}^n\right)^{\frac{1}{2}}\right) + \delta_2 \\
&\quad= \sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) G^{n,j} \Delta w_0^n + \sum_{j=1}^{J} g_j'\left(w_0^n \cdot G^{n,j}\right) w_0^n \psi^{n,j} \\
&\qquad + \lambda \sum_{i=1}^{q}\sum_{k=0}^{p}\left(f\left(w_{ik}^{n+1}\right)^{\frac{1}{2}} - f\left(w_{ik}^n\right)^{\frac{1}{2}}\right) + \delta_2 + \delta_3
\end{aligned} \tag{27}$$

where $\delta_2 = \frac{1}{2}\sum_{j=1}^{J} g_j''(s_{n,j})(w_0^{n+1} \cdot G^{n+1,j} - w_0^n \cdot G^{n,j})^2$, $\delta_3 = \sum_{j=1}^{J} g_j'(w_0^n \cdot G^{n,j})\Delta w_0^n \psi^{n,j}$ and $s_{n,j} \in R$ is a constant between $w_0^n \cdot G^{n,j}$ and $w_0^{n+1} \cdot G^{n+1,j}$.

Substituting (25) and (26) into (27) and using the Lagrange mean value theorem for $f(x)$, we have

$$
\begin{aligned}
&E(W^{n+1}) - E(W^n) \\
&\quad = -\eta \left\| E_W(W^n) \right\|^2 - \lambda \sum_{i=1}^{q} \sum_{k=0}^{p} \frac{f'\left(w_{ik}^n\right) \cdot \Delta w_{ik}^n}{2 f\left(w_{ik}^n\right)^{\frac{1}{2}}} \\
&\qquad + \left[\sum_{i=1}^{q} \alpha_i^n E_{w_i}(W^n) \cdot \Delta w_i^{n-1} + \alpha_0^n E_{w_0}(W^n) \cdot \Delta w_0^{n-1} \right] \\
&\qquad + \delta_1 + \delta_2 + \delta_3 + \lambda \sum_{i=1}^{q} \sum_{k=0}^{p} \left(f\left(w_{ik}^{n+1}\right)^{\frac{1}{2}} - f\left(w_{ik}^n\right)^{\frac{1}{2}} \right) \\
&\quad \le -\eta \left\| E_W(W^n) \right\|^2 + \frac{\lambda}{2} \sum_{i=1}^{q} \sum_{k=0}^{p} F''(t_{i,k,n}) (\Delta w_{ik})^2 \\
&\qquad + \delta_1 + \delta_2 + \delta_3
\end{aligned}
\tag{28}
$$

where $t_{i,k,n} \in R$ is between w_{ik}^n and w_{ik}^{n+1}.

According to (21) and (28) we can conclude

$$
\begin{aligned}
&E(W^{n+1}) - E(W^n) \\
&\quad \le -\eta \left\| E_W(W^n) \right\|^2 + M\lambda \sum_{i=1}^{q} \sum_{k=0}^{p} (\Delta w_{ik})^2 + \delta_1 + \delta_2 + \delta_3 \\
&\quad = -\eta \left\| E_W(W^n) \right\|^2 + M\lambda \left[\sum_{i=1}^{q} \|\Delta w_i\|^2 + \|\Delta w_0\|^2 \right] + \delta_1 + \delta_2 + \delta_3 \\
&\quad \le -\eta \left\| E_W(W^n) \right\|^2 + M\lambda\eta^2 (1+\alpha)^2 \left\| E_W(W^n) \right\|^2 + \delta_1 + \delta_2 + \delta_3 \\
&\quad \le -\eta (1 - \lambda\eta(1+\alpha)^2 M) \left\| E_W(W^n) \right\|^2 + \delta_1 + \delta_2 + \delta_3
\end{aligned}
\tag{29}
$$

Set $M = \frac{\sqrt{6}}{4\sqrt{a^3}}$, and $F(x) \equiv (f(x))^{\frac{1}{2}}$. Note that

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

It follows from (16), (18), (19) and the Cauchy-Schwartz inequality that

$$\begin{aligned}
|\delta_2| &\leq \frac{C_3}{2}\sum_{j=1}^{J}\left(\Delta w_0^n G^{n+1,j} + w_0^n \psi^{n,j}\right)^2 \\
&\leq \frac{C_3}{2}\sum_{j=1}^{J}\left(C_2\left\|\Delta w_0^n\right\| + C_5\left\|\psi^{n,j}\right\|\right)^2 \\
&\leq C_6\sum_{j=1}^{J}\left(\left\|\Delta w_0^n\right\|^2 + \left\|\psi^{n,j}\right\|^2\right) \\
&\leq C_6\sum_{j=1}^{J}\left(\left\|\Delta w_0^n\right\|^2 + C_2\sum_{i=1}^{q}\left\|\Delta w_i^n\right\|^2\right) \\
&\leq JC_6(1+C_2)\left[\left\|\Delta w_0^n\right\|^2 + \sum_{i=1}^{q}\left\|\Delta w_i^n\right\|^2\right] \\
&\leq JC_6(1+C_2)\eta^2(1+\alpha)^2\left\|E_W(W^n)\right\|^2 \\
&= C_7\eta^2(1+\alpha)^2\left\|E_W(W^n)\right\|^2
\end{aligned} \tag{30}$$

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

Similarly, we can evaluate $|\delta_3|$ as follows:

$$\begin{aligned}
|\delta_3| &\leq \frac{C_3}{2}\sum_{j=1}^{J}\left(\left\|\Delta w_0^n\right\|^2 + \left\|\psi^{n,j}\right\|^2\right) \\
&\leq \frac{C_3}{2}\sum_{j=1}^{J}\left(\left\|\Delta w_0^n\right\|^2 + C_2\sum_{i=1}^{q}\left\|\Delta w_i^n\right\|^2\right) \\
&\leq \frac{1}{2}JC_3(1+C_2)\left[\left\|\Delta w_0^n\right\|^2 + \sum_{i=1}^{q}\left\|\Delta w_i^n\right\|^2\right] \\
&\leq \frac{1}{2}JC_3(1+C_2)\eta^2(1+\alpha)^2\left\|E_W(W^n)\right\|^2 \\
&= C_8\eta^2(1+\alpha)^2\left\|E_W(W^n)\right\|^2
\end{aligned} \tag{31}$$

where $C_8 = \frac{1}{2}JC_3(1+C_2)$.

Similarly, we obtain

$$\begin{aligned}
|\delta_1| &\leq \frac{1}{2}JC_3^2C_4^2C_5\sum_{i=1}^{q}\left\|\Delta w_i^n\right\|^2 \\
&\leq \frac{1}{2}JC_3^2C_4^2C_5\eta^2(1+\alpha)^2\sum_{i=1}^{q}\left\|E_{w_i}(W^n)\right\|^2 \\
&= C_9\eta^2(1+\alpha)^2\left\|E_W(W^n)\right\|^2
\end{aligned} \tag{32}$$

where $C_9 = \frac{1}{2}JC_3^2C_4^2C_5$.

Using $C_1 = C_7 + C_8 + C_9$, from (29)–(32) and $(A3)$ we can obtain

$$
\begin{aligned}
E(W^{n+1}) - E(W^n) &\leq -\eta(1 - \lambda\eta(1+\alpha)^2 M) \sum_{i=1}^{q}\sum_{k=0}^{p} \left(E_{w_{ik}}(W^n)\right)^2 + \delta_1 + \delta_2 + \delta_3 \\
&\leq\leq -\eta(1 - M\lambda\eta(1+\alpha)^2) \left\|E_W(W^n)\right\|^2 \\
&+ c_1\eta^2(1+\alpha)^2 \left\|E_W(W^n)\right\|^2 \\
&= -\eta(1 - M\lambda\eta(1+\alpha)^2 - C_1\eta(1+\alpha)^2) \left\|E_W(W^n)\right\|^2 \\
&\leq 0
\end{aligned} \tag{33}
$$

There holds $E(W^{n+1}) \leq E(W^n)$ (n = 1,2,...). □

Step 3. we show $\lim_{n\to\infty} \|E_W(W^n)\| = 0$.

Proof From Step 2, we know that the nonnegative sequence $\{E(W^n)\}$ is monotone and it is also bounded. Hence, there must exist $E^* \geq 0$ such that $\lim_{k\to\infty} E(W^n) = E^*$.

Taking $\beta = \eta - (M\lambda + C_1)\eta^2(1+\alpha)^2$, it follows from Assumption $(A3)$ that $\beta > 0$. $\sigma^n = \sum_{i=1}^{q}\sum_{k=0}^{p}(E_{w_{ik}}(W^n))^2 = \|E_W(W^n)\|^2$ and using (33), we get

$$
\begin{aligned}
E(W^{n+1}) &\leq E(W^n) - \beta\left\|E_W(W^n)\right\|^2 \\
&\leq \cdots \\
&\leq E(W^0) - \beta\sum_{k=0}^{n}\left\|E_W(W^k)\right\|^2
\end{aligned}
$$

Since $E(W^{n+1}) \geq 0$, it gives that

$$
\beta\sum_{k=0}^{n}\left\|E_W(W^k)\right\|^2 \leq E(W^0)
$$

Let $n \to \infty$, it holds that

$$
\sum_{k=0}^{\infty}\left\|E_W(W^k)\right\|^2 \leq \frac{1}{\beta}E(W^0) < \infty
$$

This results in

$$
\lim_{n\to\infty}\left\|E_W(W^n)\right\|^2 = 0
$$

Thus

$$
\lim_{n\to\infty}\left\|E_W(W^n)\right\| = 0. \tag{34}
$$

□

Step 4. Add the assumption $(A4)$, we show $\lim_{n\to\infty}(W^n) = W^*$, where $W^* \in \Phi_0$.

Proof Note that the error function $E(W)$ defined in (11) is continuous and differentiable. According to (21) and (34), we get

$$\sum_{i=1}^{q}\sum_{k=0}^{p}(\Delta w_{ik}^{n})^2 \leq \left\|E_W(W^n)\right\|^2$$

i.e.

$$\lim_{n\to\infty}\left\|W^{n+1}-W^{n}\right\|=0 \tag{35}$$

According to the assumption $(A4)$, (34), (35) and Lemma 1, there exists a point $W^* \in \Phi_0$, such that

$$\lim_{n\to\infty} W^n = W^*. \tag{36}$$

$\square$

Now, we proved the Theorem 1 by Step 1–Step 4.

Received: 19 October 2015 Accepted: 24 February 2016
Published online: 08 March 2016

References

Bhaya A, Kaszkurewicz E (2004) Steepest descent with momentum for quadratic functions is a version of the conjugate gradient method. Neural Netw 17:65–71

Chan LW, Fallside F (1987) An adaptive training algorithm for backpropagation networks. Comput Speech Lang 2:205–218

Davis G (1994) Adaptive nonlinear approximations, Ph.D. thesis, New York University

Donoho DL (1995) De-noising by soft-thresholding. IEEE Trans Inf Theory 41:613–627

Donoho DL (2005) Neighborly polytopes and the sparse solution of underdetermined systems of linear equations, Technical report, Statistics Department, Stanford University

Fan QW, Wei W, Zurada JM (2014) Convergence of online gradient method for feedforward neural networks with smoothing $L_{1/2}$ regularization penalty. Neurocomputing 131:208–216

Haykin S (2001) Neural networks: a comprehensive foundation, 2nd edn. Tsinghua University Press, Prentice Hall, Beijing

Haykin S (2008) Neural networks and learning machines. Prentice-Hall, Upper Saddle River

Hinton GE (1989) Connectionist learning procedures. Artif Intell 40(1–3):185–234

Istook E, Martinez T (2002) Improved backpropagation learning in neural networks with windowed momentum. Int J Neural Syst 12(3–4):303–318

Karnin ED (1990) A simple procedure for pruning back-propagation trained neural networks. IEEE Trans Neural Netw 1:239–242

Kong J, Wu W (2001) Online gradient methods with a punishing term for neural networks. Northeast Math J 173:371–378

Liu QS, Cao JD (2010) A recurrent neural network based on projection operator for extended general variational inequalities. IEEE Trans Syst Man Cybern B Cybern 40(3):928–938

Liu QS, Guo ZS, Wang J (2012) A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization. Neural Netw 26:99–109

Moody JE, Rognvaldsson TS (1997) Smoothing regularizers for projective basis function networks. In: Advances in neural information processing systems 9 (NIPS 1996). http://papers.nips.cc/book/advances-in-neuralinformation-processing-systems-9-1996

Natarajan BK (1995) Sparse approximate solutions to linear systems. SIAM J Comput 24:227–234

Perantonis SJ, Karras DA (1995) An efficient constrained learning algorithm with momentum acceleration. Neural Netw 8(2):237–249

Qian N (1999) On the momentum term in gradient descent learning algorithms. Neural Netw 12(1):145–151

Qiu G, Varley MR, Terrell TJ (1992) Accelerated training of backpropagation networks by using adaptive momentum step. IEEE Electron Lett 28(4):377–379

Reed R (1993) Pruning algorithms-a survey. IEEE Trans Neural Netw 4(5):740–747

Saito K, Nakano R (2000) Second-order learning algorithm with squared penalty term. Neural Comput 12(3):709–729

Shao H, Zheng G (2011) Convergence analysis of a back-propagation algorithm with adaptive momentum. Neurocomputing 74:749–752

Tibshirani R (1996) Regression shrinkage and selection via the Lasso. J R Stat Soc B 58:267–288

Torii M, Hagan MT (2002) Stability of steepest descent with momentum for quadratic functions. IEEE Trans Neural Netw 13(3):752–756

Wang J, Wu W, Zurada JM (2012) Computational properties and convergence analysis of BPNN for cyclic and almost cyclic learning with penalty. Neural Netw 33:127–135

Weigend AS, Rumelhart DE, Huberman BA (1991) Generalization by weight elimination applied to currency exchange rate prediction. In: Proceedings of the international joint conference on neural networks, vol 1, pp 837–841

Wilson DR, Martinez TR (2003) The general inefficiency of batch training for gradient descent learning. Neural Netw 16:1429–1451

Wu W, Shao H, Li Z (2006) Convergence of batch BP algorithm with penalty for FNN training. Lect Notes Comput Sci 4232:562–569

Wu W, Li L, Yang J, Liu Y (2010) A modified gradient-based neuro-fuzzy learning algorithm and its convergence. Inf Sci 180:1630–1642

Wu W, Fan QW, Zurada JM et al (2014) Batch gradient method with smoothing $L_{1/2}$ regularization for training of feedforward neural networks. Neural Netw 50:72–78

Xu Z, Zhang H, Wang Y, Chang X, Liang Y (2010) $L_{1/2}$ regularizer. Sci China Inf Sci 53:1159–1169

Zhang NM, Wu W, Zheng GF (2006) Convergence of gradient method with momentum for two-layer feedforward neural networks. IEEE Trans Neural Netw 17(2):522–525

Zhang H, Wu W, Liu F, Yao M (2009) Boundedness and convergence of online gradient method with penalty for feedforward neural networks. IEEE Trans Neural Netw 20(6):1050–1054

Zhang HS, Zhang Y, Xu DP, Liu XD (2015) Deterministic convergence of chaos injection-based gradient method for training feedforward neural networks. Cogn Neurodyn 9(3):331–340