International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.9 (2014), pp.151-158
http://dx.doi.org/10.14257/ijmue.2014.9.9.17

Convergence of Gradient Descent Algorithm with Penalty Term For Recurrent Neural Networks

*1 Xiaoshuai Ding and 2 Kuaini Wang

*1 *College of Education, Tibet University for Nationalities, Xianyang 712082, China*
2 *College of Science, China Agricultural University, Beijing 100083, China*
*1 *missdxss@163.com*, 2 *wangkuaini1219@sina.com*

Abstract

This paper investigates a gradient descent algorithm with penalty for a recurrent neural network. The penalty we considered here is a term proportional to the norm of the weights. Its primary roles in the methods are to control the magnitude of the weights. After proving that all of the weights are automatically bounded during the iteration process, we also present some deterministic convergence results for this learning methods, indicating that the gradient of the error function goes to zero(weak convergence) and the weight sequence goes to a fixed point(strong convergence), respectively. A numerical example is provided to support the theoretical analysis.

Keywords: *recurrent neural networks, gradient descent algorithm, penalty term, boundedness, convergence*

1. Introduction

Recurrent neural networks (RNN) are a kind of networks that include one or more feedback connections. These feedback mechanisms allow RNN to learn to recognize and generate not only temporal patters, but also spatial patterns [3]. As in the case of feedforward neural networks, the gradient methods are proposed for training RNN [2] due to its simplicity, either in an off-line(batch) or an online(incremental) manner. In batch training, weight changes are accumulates over an entire presentation of the training data before being applied, while online training updates weights after the presentation of each training example. There have many convergence results for the gradient methods for RNN. Ku and Lee [4] investigated the Lyapunov convergence of the training process for a diagonal RNN with infinitely many training samples by using an online gradient training algorithm. Kuan *et al.* [5] use stochastic process theory to establish some convergence results of probability nature for the online gradient training algorithm, based on the assumption that a very large number of(or infinitely many in theory) training samples. In particularly, the deterministic convergence of the off-line gradient descent algorithm for the RNN with only finite number of training samples have been analyzed in [6]and [8], under the condition that all of the weights in the learning process are bounded. But generally, this condition is not easy to check in practice. During the training iteration process, the weights might, though not necessarily, become very large or even unbounded.

To overcome this difficulty, we introduce a penalty term into the error function, so as to prevent the norm of network weights unbounded. For simplicity, we concentrate our attention to a simple recurrent neuron. By the assumption that only the weights of the recurrent neuron, rather than all the weights of the whole network are bounded, we show that the weights of the network are actually bounded deterministically in the gradient descent learning process by adding a usual penalty term, which is proportional to the norm of the weights. In addition, we give a convergence result of the gradient learning process, in which the above boundedness result is applicable. Some techniques in Refs [9, 10] for the gradient method for training feedforward neural networks have been exploited in the proof.

The remainder of this paper is organized as follows. The network architecture and the gradient descent algorithm with a penalty term are described in the next section. Section 3 presents some lemmas and a convergence theorem. The detail proof of the theorem is provided in section 4. Section 5 is devoted to a numerical example to support our theoretical findings.

2. Network Structure and Learning Method with Penalty

As shown in Figure 1, we consider a recurrent neuron with P external input nodes and 1 output node. Denote the weight vector of the network by $w=(w_0,w_1,\cdots,w_P)^T\in R^{P+1}$. Let $\xi^j=(\xi_1^j,\cdots,\xi_P^j)\in R^P$ be external input signal at time $j(1\le j\le J)$, and $\zeta^j\in R$ be the output at time j. For convenience, we concatenate ζ^{j-1} and ξ^j to form a $(P+1)$ dimensional vector u^j as follows:

$$u_i^j=\begin{cases}\zeta^{j-1}, i=0\\ \xi_i^j \quad, i=1,2,\cdots,P\end{cases} \tag{1}$$

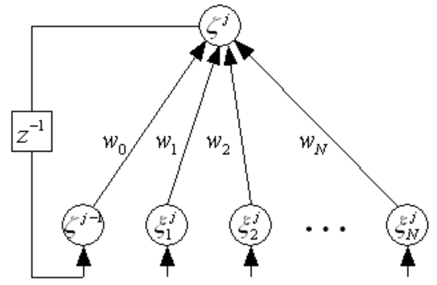

Figure 1. Architecture of P-1-1 Recurrent Neural Network

Let

$$S^j=w\cdot u^j=\sum_{i=1}^{P}w_i\xi_i^j+w_0\zeta^{j-1} \tag{2}$$

be the input to the output node at time j. Then the outputs of the network are

$$\zeta^j=g(S^j),\quad j=1,2,...,J \tag{3}$$

where $g(t)$ is the given activation function. The initial condition is

$$\zeta^0=0 \tag{4}$$

We now describe the gradient descent algorithm for training this network. Let O^j denote the target output of the network at time j. Our error function with a penalty term has the form

$$E(w)=\frac{1}{2}\sum_{j=1}^{J}[O^j-g(S^j)]^2+\lambda\|w\|^2\equiv\sum_{j=1}^{J}g_j(S^j)+\lambda\|w\|^2 \tag{5}$$

Where $g_j(S^j) = \frac{1}{2}[O^j - g(S^j)]^2$, $\lambda > 0$ is the coefficient of the penalty term, and $\|\cdot\|$ in this paper stands for the Euclidean norm. Then starting with any initial value w^0, the weights $\{w^m\}$ are updated iteratively by the gradient method as follows:

$$w^{m+1} = w^m + \Delta w^m \tag{6}$$

$$\Delta w^m = -\eta E_w(w^m) = -\eta\left[\sum_{j=1}^{J} g_j'(S^{m,j})(u^{m,j} + w_0^m p^{m,j-1}) + 2\lambda w^m\right], \quad m = 0,1,\cdots \tag{7}$$

where $E_w(w^m) = \left.\frac{\partial E(w)}{\partial w}\right|_{w=w^m}$, η is a given learning rate, and

$$S^{m,j} = S^j\big|_{w=w^m}, \qquad u^{m,\,j} = u\,\big|^j_{w=w^m} \tag{8}$$

$$p^{m,j} = \left.\frac{d\zeta^j}{dw}\right|_{w=w^m} = g'(S^{m,j})[u^{m,j} + w_0^m p^{m,j-1}] \tag{9}$$

With initial conditions

$$p^{m,0} = 0 \tag{10}$$

3. Main results

To analysis the convergence of the method we shall need the following assumptions.

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

(A2) $\{w_0^m\}(m = 0,1,2,\cdots)$ are bounded.

Remark: We note that from (5) and Assumption (A1) that $|g_j(t)|$, $|g_j'(t)|$ and $|g_j''(t)|$ are also uniformly bounded for any $t \in R$. Assumption (A2) says that $|w_0^m|$ keep bounded during the training process. This is often used in literature for a nonlinear iteration procedure to guarantee the convergence. (See, *e.g.* [7])

Theorem 1. Suppose that the error function $E(w)$ is given by (5), that the weight sequence $\{w^m\}$ is generated by the algorithm (6) and (7) for any initial value $w^0 \in R^{P+1}$, and that Assumptions (A1) and (A2) are valid, then there holds the following results:

(a) $E(w^{m+1}) \le E(w^m)$, $\quad m = 0,1,2,\cdots$;

(b) There exists a constant $E^* \ge 0$, such that

$\lim_{m\to\infty} E(w^m) = E^*$, $\qquad \lim_{m\to\infty} \|E_w(w^m)\| = 0$;

(c) There exists a constant $M > 0$, such that $\|w^m\| \le \sqrt{\frac{E(w^m)}{\lambda}} \le M$, $\quad m = 0,1,2,\cdots$;

(d) Furthermore, if there exists a closed bounded region $D \subset R^{P+1}$ satisfy $\{w^m\}_{m=0}^{\infty} \subset D$, and the set $D_0 = \{w \in D \mid E_w(w) = 0\}$ contains only finite points. Then there exists $w^* \in D_0$, such that

$$\lim_{m\to\infty} w^m = w^*.$$

The monotonicity of the error function sequence $\{E(w^m)\}$ is shown in Conclusion (a). Conclusion (c) confirms that the weight sequence $\{w^m\}$ is bounded and its bound can be controlled by λ, which is an important outcome of adding the penalty. Conclusion (b) indicates that the convergence of $E(w^m)$ and $E_w(w^m)$, which are called weak convergence. The strong convergence of $\{w^m\}$ itself is guaranteed in Conclusion (d).

4. Proofs

We first present three useful lemmas for the convergence analysis. It indicates some properties of the error function in the next lemma, which can be directly proved by induction arguments and thus is omitted. Others, for convenience, we use C for a generic constant, which may be different value even in the same equation. To simplify matters, we denote

$$\Delta S^{m,j} = S^{m+1,j} - S^{m,j}, \qquad \Delta w^m = w^{m+1} - w^m \tag{11}$$

Lemma 1. Suppose that (9)-(10) are satisfied, then

$$p^{m,j} = \sum_{k=0}^{j-1}\left[\prod_{l=j-k}^{j} g'(S^{m,l})\right](w_0^m)^k u^{m,j-k} \tag{12}$$

Lemma 2. If Assumption (A1)-(A2) are valid, then there is $C>0$ such that

$$|\Delta S^{m,j}|^2 \le C\|\Delta w^m\|^2, \qquad m=0,1,\dots;\ j=1,2,\dots,J \tag{13}$$

And there holds the following formula

$$\Delta S^{m,j} = \Delta w^m\cdot[u^{m,j} + w_0^m p^{m,j-1}] + \delta_1^{m,j} + \delta_2^{m,j} \tag{14}$$

where

$$\delta_1^{m,j} = \frac{1}{2}\sum_{k=1}^{j-1}\left[\prod_{l=j-k+1}^{j-1} g'(S^{m,l})\right](w_0^{m+1})^k g''(\theta^{m,j-k})(\Delta S^{m,j-k})^2 \tag{15}$$

$$\delta_2^{m,j} = \Delta w^m\cdot\sum_{k=1}^{j-1}\left[\prod_{l=j-k}^{j-1} g'(S^{m,l})\right][(\mathrm{w}_0^{m+1})^k - (\mathrm{w}_0^m)^k]\,u^{m,j-k} \tag{16}$$

and $\theta^{m,j-k}$ is a real number between $S^{m+1,j-k}$ and $S^{m,j-k}$.

Proof. By (11), (2) and (3), we have

$$\begin{aligned}\Delta S^{m,j} &= S^{m+1,j} - S^{m,j} = \Delta w^m\cdot u^{m,j} + w^{m+1}[u^{m+1,j} - u^{m,j}]\\ &= \Delta w^m\cdot u^{m,j} + w_0^{m+1}[\varsigma^{m+1,j-1} - \varsigma^{m,j-1}] = \Delta w^m\cdot u^{m,j} + w_0^{m+1}[g(S^{m+1,j-1}) - g(S^{m,j-1})]\end{aligned} \tag{17}$$

The use of the first order Taylor expansion and (17) to show

$$\Delta S^{m,j} = \Delta w^m\cdot u^{m,j} + w_0^{m+1}[g'(\theta_1^{m,j-1})\Delta S^{m,j-1}]$$

where $\theta_1^{m,j-1}$ lies between $S^{m+1,j-1}$ and $S^{m,j-1}$. It results from Assumption (A1), (A2) and the Cauchy-Schwarz inequality that

$$|\Delta S^{m,j}|^2 = |\Delta w^m\cdot u^{m,j} + w_0^{m+1}[g'(\theta_1^{m,j-1})\Delta S^{m,j-1}]|^2 \le C\|\Delta w^m\|^2 + C|\Delta S^{m,j-1}|^2$$

Noting

$$|\Delta S^{m,1}|^2 = |\Delta w^m\cdot u^{m,1}|^2 \le C\|\Delta w^m\|^2$$

We can prove by induction on j that (13) holds.

Next, we begin to prove (14). Applying the second order Taylor expansion for (17), we get the recursion formula of $\Delta S^{m,j}$ about j,

$$\Delta S^{m,j} = \Delta w^m\cdot u^{m,j} + w_0^{m+1}g'(S^{m,j-1})\Delta S^{m,j-1} + \frac{1}{2}w_0^{m+1}g''(\theta^{m,j-1})|\Delta S^{m,j-1}|^2 \tag{18}$$

Where $\theta^{m,j-1}$ is a real number between $S^{m+1,j-1}$ and $S^{m,j-1}$. A combination of (12), (15) and (16) leads to

$$\Delta S^{m,j} = \Delta w^m\cdot[u^{m,j} + w_0^m p^{m,j-1}] + \delta_1^{m,j} + \delta_2^{m,j}$$

The following lemma is a crucial tool for proving the strong convergence, which is basically the same as Lemma 3.5.10 in [1]. Its proof is thus omitted.

Lemma 3. Let $F: R^m \to R^n (m, n \geq 1)$ be a continuous differentiable function on a bounded closed region $\Omega \subset R^m$, and $\Omega_0 = \{z \in \Omega \mid F(z) = 0\}$ contains finite points. If the sequence $\{z^k\}_{k=1}^{\infty} \subset \Omega$ satisfy

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

Then there exists $z^* \in \Omega_0$, such that

$$\lim_{k\to\infty} z^k = z^*.$$

Now, we embark on the proof of our main result.

Proof of Theorem 1. From (5) and the Taylor expansion, we have

$$\begin{aligned} & E(w^{m+1}) - E(w^m) \\ &= \sum_{j=1}^{J} [g_j(S^{m+1,j}) - g_j(S^{m,j})] + \lambda(\| w^{m+1} \|^2 - \| w^m \|^2) \\ &= \sum_{j=1}^{J} g_j'(S^{m,j}) \Delta S^{m,j} + \frac{1}{2}\sum_{j=1}^{J} g_j''(\theta_2^{m,j}) | \Delta S^{m,j} |^2 + \lambda(2w^m + \Delta w^m) \cdot \Delta w^m, \end{aligned}$$

where $\theta_2^{m,j}$ lies between $S^{m+1,j}$ and $S^{m,j}$. It follows from Lemma 2 that

$$\begin{aligned} & E(w^{m+1}) - E(w^m) \\ &= \sum_{j=1}^{J} g_j'(S^{m,j})[\Delta w^m \cdot (u^{m,j} + w_0^m p^{m,j-1}) + \delta_1^{m,j} + \delta_2^{m,j}] + \lambda(2w^m + \Delta w^m) \cdot \Delta w^m + \sum_{j=1}^{J} \delta_3^{m,j} \\ &= \sum_{j=1}^{J} g_j'(S^{m,j}) \Delta w^m \cdot (u^{m,j} + w_0^m p^{m,j-1}) + \lambda(2w^m + \Delta w^m) \cdot \Delta w^m + \sum_{j=1}^{J} [g_j'(S^{m,j})(\delta_1^{m,j} + \delta_2^{m,j}) + \delta_3^{m,j}] \\ &= \Delta w^m \cdot \left[\sum_{j=1}^{J} g_j'(S^{m,j})(u^{m,j} + w_0^m p^{m,j-1}) + 2\lambda w^m \right] + \lambda \| \Delta w^m \|^2 + \sum_{j=1}^{J} [g_j'(S^{m,j})(\delta_1^{m,j} + \delta_2^{m,j}) + \delta_3^{m,j}] \end{aligned} \tag{19}$$

where $\delta_3^{m,j} = \frac{1}{2} g_j''(\theta^{m,j}) | \Delta S^{m,j} |^2$.

Employing (7), we conclude that

$$\sum_{j=1}^{J} g_j'(S^{m,j})(u^{m,j} + w_0^m p^{m,j-1}) + 2\lambda w^m = E_w(w^m) = -\frac{1}{\eta} \Delta w^m \tag{20}$$

By virtue of Lemma 2 and Assumption (A1), (A2), we obtain that

$$\sum_{j=1}^{J} \delta_3^{m,j} = \sum_{j=1}^{J} \frac{1}{2} g_j''(\theta^{m,j}) | \Delta S^{m,j} |^2 \leq \frac{1}{2} C_0 J C \| \Delta w^m \|^2 \leq C \| \Delta w^m \|^2 \tag{21}$$

$$\sum_{j=1}^{J} g_j'(S^{m,j})(\delta_1^{m\ j} + \delta_2^{m\ j,}) \leq C \| \Delta w^m \|^2 \tag{22}$$

Then, a combination of (19)-(22) leads to

$$E(w^{m+1}) - E(w^m) \leq -(\frac{1}{\eta} - C - \lambda) \| \Delta w^m \|^2 \tag{23}$$

If the learning rate η is chosen to satisfy

$$0 < \eta < \frac{1}{C + \lambda}, \tag{24}$$

We can derive

$$E(w^{m+1}) \le E(w^m), \quad m = 1, 2, ... \tag{25}$$

Hence, Conclusion (a) has been proved.

Since the nonnegative sequence $\{E(w^m)\}$ monotonously decrease and is bounded below, there must be a limit value $E^* \ge 0$, such that

$$\lim_{m\to\infty} E(w^m) = E^* \tag{26}$$

Let $\beta = \frac{1}{\eta} - C - \lambda$, according to (23), it suffices to show

$$0 < E(w^{m+1}) \le E(w^m) - \beta \| \Delta w^m \|^2 \le \cdots \le E(w^0) - \beta \sum_{k=0}^{m} \| \Delta w^k \|^2 \tag{27}$$

Note that $E(\mathrm{w}^m) \ge 0$, for any $m \in N$, we have

$$0 \le \sum_{k=0}^{m} \left\| \Delta w^k \right\|^2 \le \frac{1}{\beta}[E(w^0) - E(w^{m+1})] \le \frac{1}{\beta} E(w^0) < +\infty \tag{28}$$

This implies the series of positive term $\sum_{k=0}^{\infty} \| \Delta w^k \|^2$ convergence. Thus

$$\lim_{m\to\infty} \| \Delta w^m \| = 0 \tag{29}$$

The equality together with (7) derives

$$\lim_{m\to\infty} \| E_w(w^m) \| = \lim_{m\to\infty} \frac{1}{\eta} \| \Delta w^m \| = 0 \tag{30}$$

Thus Conclusion (b) holds.

By (5) and (25), we see that

$$\| w^m \|^2 = \frac{1}{\lambda}\left(E(w^m) - \sum_{j=1}^{J} g_j(S^{m,j}) \right) \le \frac{E(w^m)}{\lambda} \le \frac{E(w^0)}{\lambda} \tag{31}$$

Then let $M = \sqrt{\frac{E(w^0)}{\lambda}}$, Conclusion (c) is proved.

Conclusion (d) in Theorem 1 immediately results from a combination of Conclusion (b), equation (29) and Lemma 3.

5. Numerical Experiments

To illustrate the capacity of the learning algorithm used in this paper, a 2-dimensional linear classification problem is considered. The training examples are

$$\xi^1 = \{0, 0\},\ O^1 = 0; \qquad \xi^2 = \{0, 1\},\ O^2 = 1;$$
$$\xi^3 = \{1, 1\},\ O^3 = 1; \qquad \xi^4 = \{1, 0\},\ O^4 = 1.$$

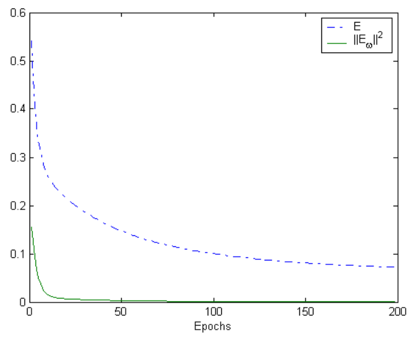

Figure 2. Error function and norm of gradient with penalty, with parameter $\eta = 0.5, \beta = 0.8$

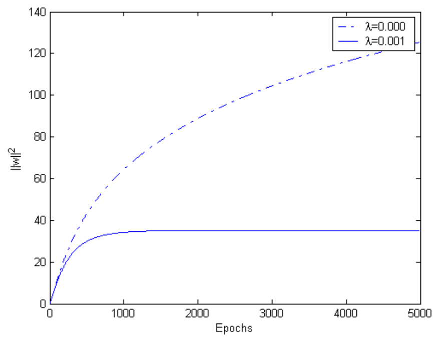

Figure 3. Norms of weight, with and without penalty, with parameter $\eta = 0.5, \beta = 0.8$

We chose a RNN with 2-1-1 structure to training. Take the activation function be $g(x) = \dfrac{1}{1+e^{-\beta x}}$, the initial weight w be stochastic chosen within $[-0.5, 0.5]$. The stop condition is when iteration steps are 5000, or the error is small than 0.001. Others, we use different value of λ to training the network and compare them with.

From Figure 2, we can see that the square error decreases monotonically and the corresponding gradient tends to zero. The effectiveness of the algorithm in controlling the weights is shown in Fig 3. Without the penalty term, the weights become larger and larger during the iteration. After adding the penalty term, the magnitude of the weight become smaller obviously and finally tends to keep steady.

Table 1. Effect of λ on Error and Weight

$\eta=0.8, \beta=0.5$	E	$\|\|w\|\|^2$	$\eta=0.5, \beta=0.8$	E	$\|\|w\|\|^2$
$\lambda=0.000$	0.0024	267.0193	$\lambda=0.000$	0.0013	125.3634
$\lambda=0.001$	0.1003	49.5700	$\lambda=0.001$	0.0558	35.1176
$\lambda=0.002$	0.1404	29.6707	$\lambda=0.002$	0.0876	23.5590
$\lambda=0.003$	0.1668	21.1546	$\lambda=0.003$	0.1083	18.0018
$\lambda=0.004$	0.1864	16.3868	$\lambda=0.004$	0.1247	14.6104
$\lambda=0.005$	0.1383	12.2963	$\lambda=0.005$	0.2019	13.3425
$\lambda=0.006$	0.1499	10.6067	$\lambda=0.006$	0.2146	11.2362
$\lambda=0.007$	0.1599	9.3162	$\lambda=0.007$	0.2254	9.6970
$\lambda=0.008$	0.1688	8.2975	$\lambda=0.008$	0.2348	8.5262
$\lambda=0.009$	0.1768	7.4728	$\lambda=0.009$	0.2430	7.6077
$\lambda=0.010$	0.2504	6.8693	$\lambda=0.010$	0.1840	6.7916

Table 1 shows that the larger the coefficient λ is, the smaller the weight becomes. Hence, our approach provides a mechanism to effectively control the magnitude of the weight, which is important for the neural networks.

References

[1] Y. Yaxiang and S. Wenyu, "Optimization Theory and Methods", Science Press, China, (**2001**).
[2] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural Networks", Neural Computation, vol. 1, (**1989**), pp. 270-280.
[3] W. Deliang, L. Xiaomei, S. C. Ahalt, "On temporal generalization of simple recurrent networks", Neural Networks, vol. 9, (**1996**), pp. 1099-1118.
[4] C. C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control", IEEE Transaction on Neural Networks, vol. 6, no. 1, (**1995**), pp. 144-156.
[5] C. M. Kuan, K. Hornik and H. White, "A convergence results for learning in recurrent neural networks", Neural Computation, vol. 6, (**1994**), pp. 420-440.
[6] X. Dongpo, L. Zhengxue and W. Wei, "Convergence of gradient descent algorithm for a recurrent neuron", Lecture Notes in Computer Science, ISNN 2007, PartⅢ, LNCS 4493, (**2007**), pp. 117-122.
[7] M. Gori and M. Maggini, "Optimal convergence of an online back propagation", IEEE Transaction on Neural Networks, vol. 7, no. 1, (**1996**), pp. 251-254.
[8] X. Dongpo, L. Zhengxue and W. Wei, "Convergence of gradient method for a fully recurrent neural networks", Soft Computing, vol. 14, no. 3, (**2010**), pp. 245-250.
[9] G. E. Hiton, "Connectionist learning procedure", Artificial Intelligence, no. 40, (**1989**), pp.185-234.
[10] S. Hongmei, W. Wei and L. Lijun, "Convergence of online gradient method with penalty for BP neural networks", Communications in Mathematical Research, vol. 26, no. 1, (**2010**), pp. 67-75.
[11] S. Haykin, Translated in Chinese by Y. Shiwei and S. Zhongzhi, "Neural Networks: A Comprehensive Foundation", 2nd edition. China Machine Press, China, (**2004**).

Authors

Xiaoshuai Ding, she received the B.S. degree in information and computation science from Jilin University, China, 2005, the M.S. degree in computational mathematics from Dalian University of Technology, 2007. She is currently a lecturer in the College of Education, Tibet University for Nationalities. Her present research interest includes design and analysis of recurrent neural networks for constrained optimization.

Kuaini Wang, she has received the B.S. degree in information and computation science from Jilin University, China, 2005, the M.S. degree in mathematics from China Agricultural University, 2009. She is currently pursuing the Ph.D. degree in operations research and management science at China Agricultural University. Her research interest includes machine learning and support vector machines.