Neural Networks based on Quantum Gated Nodes

Jianping Li
School of Computer & Information Technology
Northeast Petroleum University
Daqing, Heilongjiang, China
Email: leejp [AT] 126.com

Siyuan Zhao
School of Computer & Information Technology
Northeast Petroleum University
Daqing, Heilongjiang, China

Haiyan Zhang
Daqing Oilfield Logistic Administration
Daqing, Heilongjiang, China

Yinpeng Liu
Daqing Oilfield Information Technology
Company IOT Branch
Daqing, Heilongjiang, China

Abstract—**On the basis of analyzing the principles of the quantum rotation gates and quantum controlled-NOT gates, a novel quantum neural networks model is proposed in our paper. In this model, the input information is expressed by the qubits, which, as the control qubits after rotated by the rotation gate, control the qubits in the hidden layer to reverse. The qubits in the hidden layer, as the control qubits after rotated by the rotation gate, control the qubits in the output layer to reverse. The networks output is described by the probability amplitude of state $|1\rangle$ in the output layer. The learning algorithm is derived and discussed based on the gradient descent algorithm. It has been shown in two application examples of pattern recognition and function approximation that this new quantum neural network model is superior to the standard BP networks with regard to their convergence rate, number of iterations, approximation ability, and robustness.**

Keywords- quantum computing, quantum neural networks, quantum gate, learning algorithm

I. INTRODUCTION (HEADING 1)

In 1980s, Benioff firstly proposed the concept of quantum computation[1]. Shor discussed the first quantum algorithm of very large integer factorization[2] in 1994. In 1996, Grover explored an important quantum algorithm, which can search for a marked state in an unordered list[3]. Although the quantum machines are not yet technologically feasible, the quantum algorithms that can be applied on the quantum computers are indeed interesting and significantly different from the classical computing. As we know, fuzzy logic, evolutionary computation, and neural networks are regarded as intelligent computing (soft computing), and also have some comparability with the quantum computation[4]. Therefore, combination of these computing methods is emerging. Different from the Hebbian learning, a quantum neural network can be used for the enriched learning of neural networks. Proposed by Penrose in 1989[5], the idea of quantum information processing in the human brain is still a controversial theory, which has not been experimentally proved. However, the exploration of quantum information devices is a promising research topic, because of the enhanced capacity and speed from the quantum mechanism. With such characteristics as smaller size of quantum devices, larger capacity of quantum networks and faster information processing speed, a quantum neural network can mimic some distinguishing properties of the brain better than the classical neural networks, even if the real human brain does not even have any quantum element. In a word, the quantum neural networks have important research significance in both theory and engineering.

Since Kak firstly proposed the concept of quantum neural computation [6] in 1995, the quantum neural networks have attracted great attention during the past decade, and a large number of novel techniques have been studied for the quantum computation and neural networks. For example, Ref. [7] proposed the model of quantum neural networks with multi-level hidden neurons based on the superposition of quantum states in the quantum theory. In 1998, a new neural network model with quantum circuit was developed for the quantum computation, and was proven to exhibit a powerful learning capability [8]. Matsui et al. develop a quantum neural networks model using the single bit rotation gate and two-bit controlled-NOT gate. They also investigate its performance in solving the four-bit parity check and the function approximation problems [9]. Ref. [10] proposes the neural networks with the quantum gated nodes, and indicates that such quantum networks may contain more advantageous features from the biological systems than the regular electronic devices. In our previous work [11], we have proposed a quantum BP neural networks model with learning algorithm based on the single-qubit rotation gates and two-qubit controlled-NOT gates.

In this paper, we study a new neural networks model with the quantum gated nodes. Our scheme is a three-layer model with a hidden layer, which employs the gradient descent principle for learning. The input-output relationship of this model is derived based on the physical meaning of the quantum gates. The convergence rate, number of iterations, and approximation error of the quantum neural networks are examined with different learning coefficients. Two application

examples demonstrate that this quantum neural network is superior to the standard BP networks (SBP).

II. Quantum Bits and Quantum Gates

A. Quantum bits

In the quantum computers, the 'qubit' has been introduced as the counterpart of the 'bit' in the conventional computers to describe the states of the circuit of quantum computation. The two quantum physical states labeled as $|0\rangle$ and $|1\rangle$ express 1 bit information, in which $|0\rangle$ corresponds to the bit 0 of classical computers, while $|1\rangle$ bit 1. Notation of "$|\ \rangle$" is called the Dirac notation, which is the standard notation for the states in the quantum mechanics. The difference between bits and qubits is that a qubit can be in a state other than $|0\rangle$ and $|1\rangle$. It is also possible to form the linear combinations of the states, namely superpositions:

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle\ , \tag{1}$$

where α and β are complex numbers, called probability amplitudes. That is, the qubit state $|\phi\rangle$ collapses into either $|0\rangle$ state with probability $|\alpha|^2$, or $|1\rangle$ state with probability $|\beta|^2$, and we have

$$|\alpha|^2 + |\beta|^2 = 1. \tag{2}$$

Hence, the qubit can be described by the probability amplitudes as $[\alpha, \beta]^T$.

Suppose we have n qubits, and correspondingly, a n qubits system has 2^n computational basis states. Similar to the case of a single qubit, the n qubits system may form the superpositions of 2^n basis states:

$$|\phi\rangle = \sum_{x\in(0,1)^n} a_x |x\rangle, \tag{3}$$

where a_x is called probability amplitude of the basis states $|x\rangle$, and '$\{0,1\}$' means 'the set of strings of length two with each letter being either zero or one'. The condition that these probabilities can sum to one is expressed by the normalization condition:

$$\sum_{x\in(0,1)^n} |a_x|^2 = 1. \tag{4}$$

B. Quantum gate and its representation

In the quantum computation, the logic function can be realized by applying a series of unitary transform to the qubit states, which the effect of the unitary transform is equal to that of the logic gate. Therefore, the quantum services with the logic transformations in a certain interval are called the quantum gates, which are the basis of performing the quantum computation.

The definition of a single qubit rotation gate is given:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}. \tag{5}$$

Let the quantum state $|\phi\rangle = \begin{bmatrix} \cos\theta_0 \\ \sin\theta_0 \end{bmatrix}$, and $|\phi\rangle$ can be transformed by $R(\theta)$ as follows:

$$R(\theta)|\phi\rangle = \begin{bmatrix} \cos(\theta_0 + \theta) \\ \sin(\theta_0 + \theta) \end{bmatrix}. \tag{6}$$

It is obvious that $R(\theta)$ shifts the phase of $|\phi\rangle$. The prototypical multi-qubit quantum logic gate is the controlled-NOT or CNOT gate. This gate has two input qubits, namely the control qubit and target qubit, respectively..

The action of this gate can be described as the following. If the control qubit is set to 0, the target qubit is left alone. If the control qubit is set to 1, the target qubit is flipped [12].

In the two-qubit controlled-NOT gate, how to condition on a single qubit set is obvious. This condition can be generalized to the multi-qubits controlled-NOT gate. Suppose we have $n+1$ qubits, and X is a single qubit NOT gate. We define the multi-qubits controlled-NOT operation $C^n(X)$ by [12]

$$C^n(|x_1x_2\cdots x_n\rangle|\phi\rangle) = |x_1x_2\cdots x_n\rangle|X^{x_1x_2\cdots x_n}|\phi\rangle, \tag{7}$$

where $x_1x_2\cdots x_n$ in the exponent of X is the product of the bits $x_1, x_2, \cdots, x_n$. That is, the operator X is applied to the last one qubit (the target qubit), if the first n qubits (the control qubits) are all equal to one. Otherwise, no action is taken.

Assume that the control qubits are given by $|x_i\rangle = a_i|0\rangle + b_i|1\rangle$, where $i = 1,2,\cdots,n$. When the target qubit $|\phi\rangle = |0\rangle$, the output of the multi-qubits controlled-NOT gate can be described:

$$C^n(|x_1\rangle\otimes\cdots\otimes|x_n\rangle\otimes|0\rangle)$$
$$=|x_1\rangle\otimes\cdots\otimes|x_n\rangle\otimes|0\rangle-b_1b_2\cdots b_n\,|\overbrace{11\cdots1}^{n}\,0\rangle. \quad (8)$$
$$+b_1b_2\cdots b_n\,|\overbrace{11\cdots1}^{n}\,1\rangle$$

It is observed from Eq.(8) that the output of $C^n(X)$ is in the entangled state of $n+1$ qubits, and the probability of the target qubit state, in which $|1\rangle$ is observed, equals to $(b_1b_2\cdots b_n)^2$.

III. Neural Networks Base on Quantum Gated Nodes

In this paper, a new Quantum Gate Neural Networks model (QGNN) is proposed, as illustrated in Fig.1, where $|x_1\rangle,|x_2\rangle,\cdots,|x_n\rangle$ are the network input, $|h_1\rangle,|h_2\rangle,\cdots,|h_p\rangle$ are the output of the hidden layer, and $|y_1\rangle,|y_2\rangle,\cdots,|y_m\rangle$ are the output of the output layer.

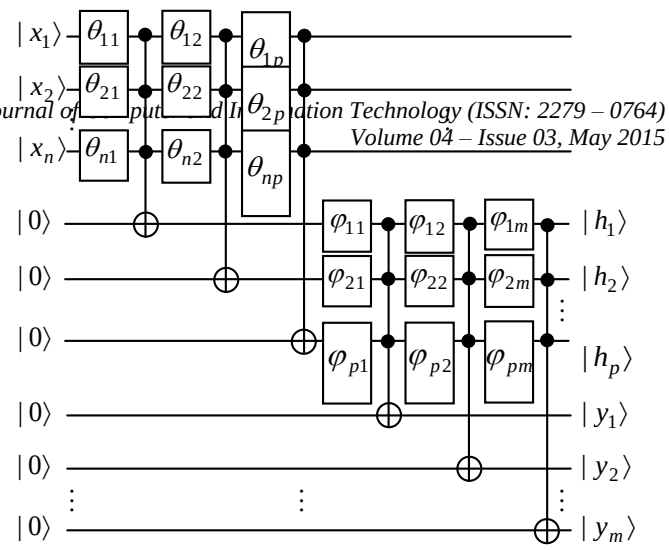

Figure 1. Networks model with quantum gated nodes

A. Quantum state description of training samples

For the training samples in the *n*-dimension Euclid-space $\bar{X}=(\bar{x}_1,\bar{x}_2,\cdots,\bar{x}_n)^T$, where $\bar{x}_i\in[a_i,b_i]$, the transformation to realize the quantum state description of the training samples is defined as:

$$|X\rangle=[|x_1\rangle,|x_2\rangle,\cdots,|x_n\rangle]^T, \quad (9)$$

where

$$|x_i\rangle=\cos\left(\frac{2\pi(\bar{x}_i-a_i)}{b_i-a_i}\right)|0\rangle+\sin\left(\frac{2\pi(\bar{x}_i-a_i)}{b_i-a_i}\right)|1\rangle$$
$$=\left[\cos\left(\frac{2\pi(\bar{x}_i-a_i)}{b_i-a_i}\right)\quad\sin\left(\frac{2\pi(\bar{x}_i-a_i)}{b_i-a_i}\right)\right]^T.$$

B. Output of nodes in every layer

Let $|x_i\rangle=\cos\theta_i|0\rangle+\sin\theta_i|1\rangle$. According to Eqs. (6)-(8), the output of each layer of our networks can be written:

$$|h_j\rangle=\cos(\varphi_j)|0\rangle+\prod_{i=1}^{n}\sin(\theta_i+\theta_{ij})|1\rangle, \quad (10)$$
$$=\cos(\varphi_j)|0\rangle+\sin(\varphi_j)|1\rangle$$

$$|y_k\rangle=\cos(\xi_k)|0\rangle+\prod_{j=1}^{p}\sin(\varphi_j+\varphi_{jk})|1\rangle, \quad (11)$$
$$=\cos(\xi_k)|0\rangle+\sin(\xi_k)|1\rangle$$

where $i=1,2,\cdots,n$, and $j=1,2,\cdots,p$, $k=1,2,\cdots,m$. $\varphi_j=\arcsin\left(\prod_{i=1}^{n}\sin(\theta_i+\theta_{ij})\right)$, $\xi_k=\arcsin\left(\prod_{j=1}^{p}\sin(\varphi_j+\varphi_{jk})\right)$.

In this paper, we define the output of the nodes in each layer as the probability amplitude of the corresponding state, in which $|1\rangle$ is observed. Thus, the actual output of our networks is rewritten as follows:

$$h_j=\sin(\varphi_j)=\prod_{i=1}^{n}\sin(\theta_i+\theta_{ij}), \quad (12)$$

$$y_k=\prod_{j=1}^{p}\sin(\varphi_j+\varphi_{jk})$$
$$=\prod_{j=1}^{p}\sin\left(\arcsin\left(\prod_{i=1}^{n}\sin(\theta_i+\theta_{ij})\right)+\varphi_{jk}\right). \quad (13)$$

C. Parameters update in every layer

In this QGNN model, the parameters to be updated are the rotation angles of the quantum rotation gates in every layer. Suppose the desired normalized outputs are $\tilde{y}_1,\tilde{y}_2,\cdots,\tilde{y}_m$. The evaluation function is defined:

$$E=\sum_{k=1}^{m}(\tilde{y}_k-y_k)^2/2. \quad (14)$$

According to the gradient descent algorithm, the gradient of the rotation angles of the quantum rotation gates in every layer can be calculated as follows:

$$-\frac{\partial E}{\partial\theta_{ij}}=\frac{\sum_{k=1}^{m}(\tilde{y}_k-y_k)y_k ctg(\varphi_j+\varphi_{jk})h_j ctg(\theta_i+\theta_{ij})}{\sqrt{1-h_j^2}}, \quad (15)$$

where $h_j=\prod_{i=1}^{n}\sin(\theta_i+\theta_{ij})$.

$$-\frac{\partial E}{\partial \varphi_{jk}} = (\tilde{y}_k - y_k) y_k ctg(\varphi_j + \varphi_{jk}) . \quad (16)$$

Hence, the rotation angles of the quantum rotation gates in each layer are updated by following Eqs. (17)-(18):

$$\theta_{ij}(t+1) = \theta_{ij}(t) - \eta \frac{\partial E}{\partial \theta_{ij}} , \quad (17)$$

$$\varphi_{jk}(t+1) = \varphi_{jk}(t) - \eta \frac{\partial E}{\partial \varphi_{jk}} , \quad (18)$$

where t is the current iteration, and η is the learning coefficient. It can be seen from Eq.(13) that there exist many global optimum solutions for the iteration sequences $\{\theta_{ij}(t)\}$ and $\{\varphi_{jk}(t)\}$.

IV. Algorithm Description

The training algorithm of our QGNN is described in details as follows.

Step 1. Represent the training samples by their quantum states.

Step 2. Initialize the QGNN networks parameters, i.e., the nodes number of each layer, rotation angles θ_{ij} and φ_{jk}, learning coefficient η, target error ε, and maximum of iterations Max_N. Set the current iteration $t = 0$.

Step 3. For each sample, calculate the actual output of the network according to Eq.(13), and update the network parameters using Eqs.(15)-(18).

Step 4. For all M samples, calculate the output error E_i by $E_i = \underset{1 \le k \le m}{Max} | \tilde{y}_{ik} - y_{ik} |$,where $i = 1, \cdots, M$. Let $E_{\max} = \max(E_1, \cdots, E_M)$. If $E_{\max} < \varepsilon$ or $t > Max\_N$, go to Step 5. Otherwise, $t = t + 1$, and return back to **Step 3**.

Step 5. Save θ_{ij} and φ_{jk}.

V. Simulations

To examine the effectiveness of the proposed QGNN, two examples are used to compare it with the SBP in this section. Our QGNN has the same structure and parameters as the SBP in the simulations. Some relevant concepts are defined as follows.

Definition 1 (Approximation error). Suppose $\tilde{y}_{ij}$ and y_{ij} denote the desired output and actual output, respectively. The approximation error is defined:

$$E = \underset{1 \le i \le M}{Max} (\underset{1 \le j \le m}{Max} | \tilde{y}_{ij} - y_{ij} |) , \quad (19)$$

where M denotes the number of the training samples, and m denotes the dimension of the output vector.

Definition 2 (Average approximation error). Suppose $E_1, E_2, \cdots, E_N$ denote the approximation error of the N-th of the networks training, respectively. The average approximation error is defined:

$$E_{avg} = \frac{1}{N} \sum_{i=1}^{N} E_i . \quad (20)$$

Definition 3 (Convergence). Suppose E denotes the approximation error after all the iterations, and ε denotes the target error. If $E < \varepsilon$, the networks training is considered to have converged.

Definition 4 (Convergence rate). Suppose N denotes the total number of training trials, and C denotes the number of convergent training trials. The convergence rate is:

$$\lambda = C / N . \quad (21)$$

A. Pattern recognition

The XOR problem is a typical benchmark for testing neural networks learning algorithms, which is a nonlinear classification problem. There are some local minima on its irregular optimization surface. In our simulations, the XOR problem has nine input patterns to classify, as shown in Fig.2, where the circles belong to the 1st class, and rectangles the 2nd class.

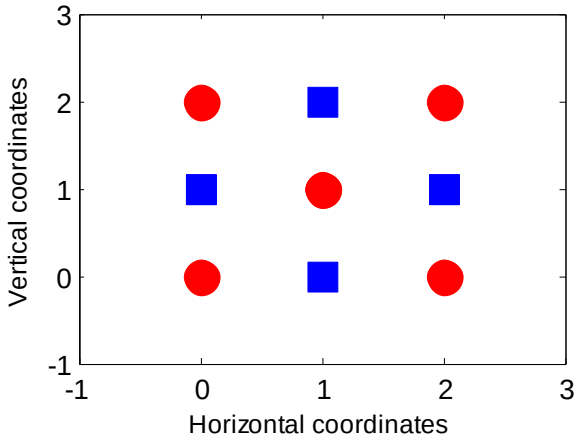

Figure 2. XOR problem in the simulations

Firstly, we investigate how the convergence rate changes with regard to the learning coefficient. The networks structure is set to be 2-10-1, the target error 0.1, and the maximum number of iterations 15,000. The learning coefficient is from

{0.1, 0.2, …, 1.0 }. This example is run for 100 times for each learning coefficient of the QGNN and SBP. The convergence rate of the QGNN is always at 100%, no matter how the learning coefficient changes. However, the convergence rate of the SBP significantly changes. When the learning coefficient is 0.1, the convergence rate of the SBP is 22%, and when the learning coefficient is greater than 0.1, the maximum of the convergence rate is only 69%. The comparison results of the convergence rate are shown in Fig.3.

Next, we examine how the number of the iterations changes, if the learning coefficient changes. The same example is used and run for 100 times for each learning coefficient of the QGNN and SBP. When the learning coefficient changes, for the average iterations of the QGNN, the maximum and minimum numbers are 687.70 and 275.01, respectively. The difference between these two values is only 412.69. However, in case of the SBP, the maximum and minimum numbers are 12,335.45 and 1,697.24, respectively. The difference is 10,638.21. The comparison results are depicted in Fig.4, which demonstrate that the QGNN needs considerably less number of iterations than the SBP.

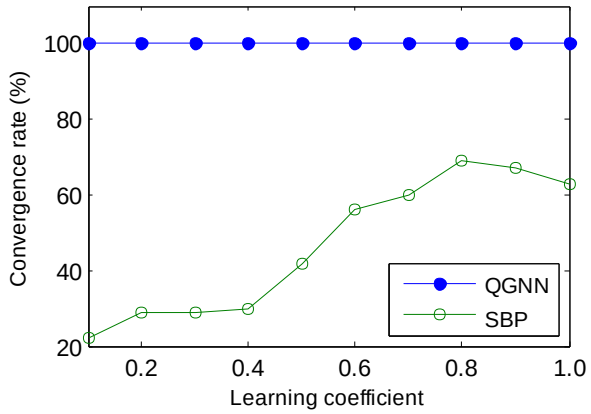

Figure 3. The convergence rate and learning coefficient

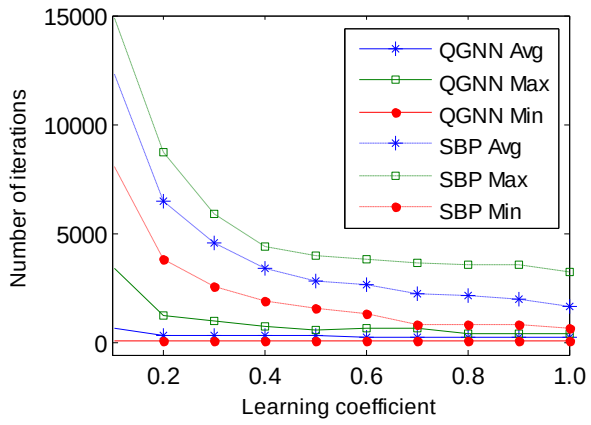

Figure 4. Example of a figure caption. (figure caption)

B. *Function approximation*

In this simulation, the following function is considered:

$$y = (1.0 + \sqrt{x_1} + \frac{1}{x_2} + x_3^{-1.5})^2 \ , \qquad (22)$$

where x_1 , x_2 , and x_3 are the integers in {1, 2, 3, 4, 5} . To approximate this nonlinear function, we use 20 groups of sample data, as in Table 1.

Firstly, we investigate how the approximation error changes with the learning coefficient. The networks structure is chosen to be 3-8-1, and the maximum of iteration 2,000. The learning coefficient set is {0.1, 0.2, … , 1.0 }. This example is run for 100 times for each learning coefficient of the QGNN and SBP. When the learning coefficient changes, the maximum and minimum of the average approximation error of the QGNN are 0.0267 and 0.0083, respectively. The range of the maximum and minimum is 0.0184, which is insensitive to the change of the learning coefficient. However, for the average approximation error of the SBP, the maximum and minimum are 0.1958 and 0.0660, respectively. The range of change is 0.1298. The comparison results are shown in Fig.5, which indicate that the QGNN is superior to the SBP concerning both the approximation ability and robustness.

TABLE I. SAMPLE DATA FOR FUNCTION APPROXIMATION

x_1	x_2	x_3	y		x_1	x_2	x_3	y
1	3	1	11.111		3	2	1	17.910
1	3	2	7.219		3	2	2	12.857
1	3	3	6.380		3	2	3	11.727
1	3	4	6.043		3	2	4	11.270
1	3	5	5.870		3	2	5	11.032
2	5	1	13.063		5	3	1	20.879
2	5	2	8.808		5	3	2	15.390
2	5	3	7.877		5	3	3	14.152
2	5	4	7.503		5	3	4	13.649
2	5	5	7.310		5	3	5	13.387

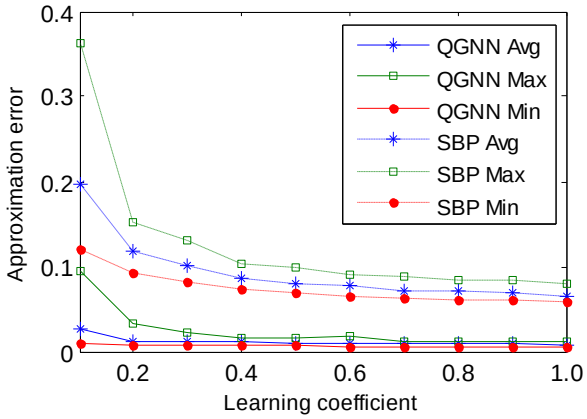

Figure 5. Relationship between approximation error and learning coefficient

Secondly, we explore the relationship between the number of the iterations and learning coefficient. The networks structure is 3-8-1, the maximum of iterations is 50,000, and the target error is 0.05. The learning coefficient set is {0.1, 0.2, … , 1.0 } again. 100 independent trials are run for each learning coefficient of the QGNN and SBP. When the learning coefficient changes, the maximum average number of the iterations of the QGNN is 2,684.91, and the minimum is only 111.64. The range of these two values is 2,573.27. However, for the SBP, the maximum and minimum are 29,846.60 and 3,767.81, respectively, and the corresponding range is 26,078.79. The QGNN is compared with the SBP in Fig.6. It is clearly visible that the QGNN is better than the SBP with regard to the number of iterations.

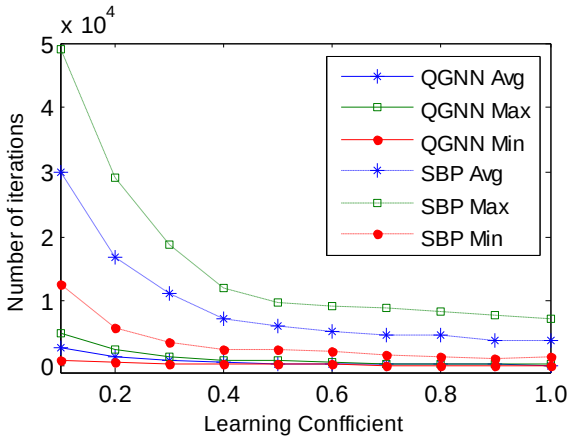

Figure 6. Relationship between number of iterations and learning coefficient

We also investigate how the convergence rate behaves when the learning coefficient changes. The networks structure is 3-8-1, the target error is 0.05, and the maximum number of iterations is 2,000. The learning coefficient is still from {0.1, 0.2, … , 1.0 }. This example has been run for 100 separate times for each learning coefficient of the QGNN and SBP. The convergence rate of the QGNN has always been at 100%, independent of the change of the learning coefficient. Nevertheless, the change of the convergence rate of the SBP is obvious. When the learning coefficient is smaller than 0.2, the convergence rate of the SBP is 0.0%, and when the learning coefficient is greater than 0.2, the maximum convergence rate is only 64%. The simulation results are provided in Fig.7.

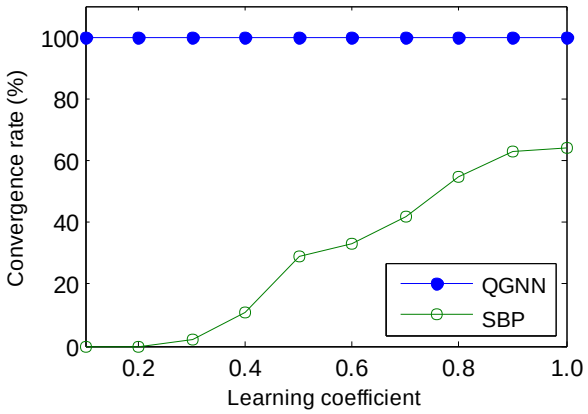

Figure 7. Relationship between convergence rate and learning coefficient

It is worth pointing out that the slow convergence caused by the well-known BP learning algorithm is a serious problem. The learning efficiency of the BP algorithm has been investigated in many ways. By analyzing the error saturation, Ref.[13] discusses an error saturation handling technique to prevent the nodes in the output layer from being saturated, which can not only improve the learning efficiency but also maintain the semantic meaning of the energy function. In Ref.[14], a dynamic and adaptive BP neural networks model is proposed by hybridizing the genetic algorithms, simulated annealing and error back propagation to overcome their disadvantages. It is demonstrated that the proposed model can enhance the generalization ability. Ref.[15] studies the ability of learning automata-based schemes in escaping from the local minima, since the standard error back propagation usually fails to find the global optimum. These learning automata-based schemes have a better convergence capability than the conventional methods. Ref.[16] proposes two novel approaches, back propagation with magnified gradient function and deterministic weight modification, to speeding up the convergence rate and improving the global convergence capability of the standard BP learning algorithm. Our QGNN is different from the aforementioned techniques. With the embedded quantum computation mechanism, multiple attractors are generated in the QGNN. The existence of these inbuilt periodic attractors leads to an efficient convergence of the QGNN.

VI. CONCLUSIONS

In this paper, we propose a novel neural networks model with the quantum gated nodes, and analyze the effects of the learning coefficients on its performance. Our neural network has three layers including a hidden layer, which employs the gradient descent method for learning. With regard to the convergence rate, classification and approximation ability,

number of iterations, and robustness, it has been shown that the proposed QGNN model is better than the SBP. We are going to study more effective quantum neural networks and algorithms based on the continuous-time evolution and Schrodinger's equations.

REFERENCES

[1] Benioff P. Quantum mechanical hamiltonian models of Turing machines. Journal of Statistical Physics, 1982, (29)3: 515-546.

[2] Shor P. W. Algorithms for quantum computation: Discrete logarithms and factoring [A], In: Proc of the 35th Annual Symp on Foundations of Computer Science[C]. New Mexico: IEEE Computer Society Press, 1994, 124-134.

[3] Grover L. K. A fast quantum mechanical algorithm for database search [A], In: Proceedings of the 28th Annual ACM Symposium on the Theory of Computing[C]. Pennsylvania, 1996, 212-221.

[4] Penrose R. Shadows of the Mind: A Search for the Missing Science of Consciousness. London: Oxford University Press, 1994.

[5] Penrose R. The Emperor's New Mind: Concerning Computers, Minds, and The Laws of Physics. London: Oxford University Press, 1989.

[6] Kak S. On quantum neural computing. Information Sciences, 1995, 83: 143-160.

[7] Gopathy P., Nicolaos B. K. Quantum Neural networks (QNN's): Inherently fuzzy feedforward neural networks. IEEE Transactions on Neural Networks, 1997, 8 (3): 679-693.

[8] Matsui N., Takai M., Nishimura H. A network model based on qubit-like neuron corresponding to quantum circuit, Trans. IEICE, J81-A (1998): 1687-1692.

[9] Matsui N., Kouda N., Nishimura H. Neural network based on QBP and its performance. Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural networks. 2000, 3 (24-27): 247-252.

[10] Shafee F. Neural networks with quantum gated nodes. Engineering Applications of Artificial Intelligence. 2007, 20(4): 429-437.

[11] Li P. C., Li S. Y. Learning algorithm and application of quantum BP neural networks based on universal quantum gates. Journal of Systems Engineering and Electronics. 2008, 19(1): 167-174.

[12] Nielsen M. A., Chuang M. Quantum Computation and quantum Information. London: Cambridge University Press, 2000.

[13] Lee H. M., Chen C. M., Huang T. C. Learning efficiency improvement of back-propagation algorithm by error saturation prevention method. Neurocomputing, 2001, 41: 125-143.

[14] Yu S. W., Zhu K., Diao F. Q. A dynamic all parameters adaptive BP neural networks model and its application on oil reservoir prediction. Applied Mathematics and Computation, 2008, 195: 66–75.

[15] Meybodi M. R., Beigy H. A note on learning automata-based schemes for adaptation of BP parameters. Neurocomputing, 2002, 48: 957-974.

[16] Ng S. C., Cheung C. C., Leung S. H. Magnified Gradient Function With Deterministic Weight Modification in Adaptive Learning. IEEE TRANSACTIONS ON NEURAL NETWORKS, 2004, 15(6): 1411-1423.