Int J Theor Phys
DOI 10.1007/s10773-014-2393-1

Comments on "Quantum M-P Neural Network"

Adenilton J. da Silva · Wilson R. de Oliveira · Teresa B. Ludermir

Received: 12 June 2014 / Accepted: 23 October 2014

Abstract In a paper on quantum neural networks, Zhou and Ding (Int. J. Theor. Phys. 46(12):3209-3215 (2007)) proposed a new model of quantum perceptron denoted quantum M-P neural network and showed its functionality by an example. In this letter, we show that the proposed learning algorithm does not follow an unitary evolution and the proposed neuron can be efficiently simulated by a classical single layer neural network.

Keywords Quantum computation · Neural networks · Quantum learning

1 Introduction

In [2] Kak proposed an idea of quantum neural computation. Since then, several researchers have proposed quantum neural networks [6–8] and quantum inspired neural networks [3, 4]. In [8] Zhou and Ding proposed a quantum perceptron; they called the new neuron model as quantum M-P neural network (qMPN). The weights of qMPN are stored in a squared matrix W. Let $|x\rangle$ be an input vector, the output $|y\rangle$ can be calculated as in (1).

$$|y\rangle = W|x\rangle \tag{1}$$

Representing a neural network as a quantum operator can bring new possibilities to neural computation. A quantum neural network can receive any superposition of quantum states

A. J. da Silva (✉) · T. B. Ludermir
Centro de Informática, Universidade Federal de Pernambuco, Recife, Brazil
e-mail: ajs@deinfo.ufrpe.br

T. B. Ludermir
e-mail: tbl@cin.ufpe.br

A. J. da Silva · W. R. de Oliveira
Departamento de Estatística e Informática, Universidade Federal Rural de Pernambuco, Recife, Brazil

W. R. de Oliveira
e-mail: wilson.rosa@gmail.com

Published online: 19 November 2014

at its inputs which can be seen as untrained data. The network operator acts linearly. For instance, if a neuron with weight matrix W receives the input $|\alpha\rangle = \sum_{k=1}^{n} \alpha_k |k\rangle$ it will act linearly in each value in the superposition and its output will be $|y\rangle = \sum_{k=1}^{n} \alpha_k W|k\rangle$.

In Section 4.2 of [8] is presented a prepocessing step to work with non-orthogonal states. This prepossessing step changes the input representation from qubits (or vectors) to a matrix of inner products. Any quantum bit can be represented as a superposition of orthogonal vectors and this step is not used in any other section of [8]. This prepocessing is a nonquantum operation for it accesses the amplitudes since inner product is employed freely. Calculating the inner product with a basis element results the amplitude corresponding to that basis element in a superposition.

In this letter we first show that the proposed learning algorithm for qMPN does not preserve unitary operators and it can produce non unitary weight matrices. After these steps we show that any qMPN can be efficiently simulated by a single layer neural network composed of classical perceptrons and that the learning algorithm of the qMPN is exactly the classical perceptron learning rule.

2 Learning Algorithm

The learning algorithm proposed in [8] for qMPN is described in Algorithm 1. The weights update rule of Algorithm 1 is described in (2), where w_{ij} are the entries of the $n \times n$ matrix W, $1 \leq i, j, \leq n$, η is a learning rate, $|d\rangle_i$ and $|y\rangle_i$ corresponds to the ith probability amplitude of n-dimensional qubits $|d\rangle$ and $|y\rangle$, and $|x\rangle_j$ is the jth probability amplitude of the n-dimensional qubit $|x\rangle$.

Algorithm 1 Learning algorithm qMPN

1 Let $W(0)$ be a weight matrix
2 Given a set of quantum examples in the form $(|x\rangle, |d\rangle)$, where $|x\rangle$ is an input and $|d\rangle$ is the desired output
3 Calculate $|y\rangle = W(t)|x\rangle$, where t is the iteration number
4 Update the weights following the learning rule described in (2).
5 Repeat steps 3 and 4 until a stop criterion is met.

$$w_{ij}(t+1) = w_{ij}(t) + \eta \left(|d\rangle_i - |y\rangle_i\right) |x\rangle_j \tag{2}$$

If $|d\rangle = \alpha|0\rangle + \beta|1\rangle$, then $|d\rangle_1 = \alpha$ and $|d\rangle_2 = \beta$ are the probability amplitudes of the state $|d\rangle$. In quantum computation, one cannot direct access a probability amplitude [5]. Measuring a qubit returns only $|0\rangle$ with probability $|\alpha|^2$ or $|1\rangle$ with probability $|\beta|^2$. Therefore, the learning rule described in (2) is not a quantum learning rule and one cannot use a quantum computer to train the qMPN with Algorithm 1.

One alternative is to use a classical computer to train the qMPN with Algorithm 1. However this strategy can lead to non-unitary neurons configurations, as we show in (3), where $W(0) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$, $|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $|d\rangle = \frac{1}{\sqrt{2}}(|0\rangle + 1)$, and $\eta = 1$. After the first learning algorithm iteration the qMPN will be represented by the matrix $W(1)$. Clearly $W(1)$ is a non unitary operator. Quantum gates must be unitary operators [5], so the qMPN

trained with this algorithm is not necessarily a quantum neuron. We conclude this paragraph with Theorem 1.

$$
\begin{aligned}
|y\rangle = W(0)|x\rangle &= \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \cdot \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\
w_{00}(t+1) &= w_{00}(t) + \eta \left(|d\rangle_0 - |y\rangle_0\right) |x\rangle_0 = \\
&= 0 + \left(\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}}\right) \cdot \frac{1}{\sqrt{2}} = 0 \\
w_{01}(t+1) &= w_{01}(t) + \eta \left(|d\rangle_0 - |y\rangle_0\right) |x\rangle_1 = \\
&= 1 + \left(\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}}\right) \cdot \left(\frac{1}{\sqrt{2}}\right) = 1 \\
w_{10}(t+1) &= w_{10}(t) + \eta \left(|d\rangle_1 - |y\rangle_1\right) |x\rangle_0 = \\
&= -1 + \left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right) \cdot \left(\frac{1}{\sqrt{2}}\right) = 0 \\
w_{11}(t+1) &= w_{11}(t) + \eta \left(|d\rangle_1 - |y\rangle_1\right) |x\rangle_1 = \\
&= 0 + \left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right) \cdot \left(\frac{1}{\sqrt{2}}\right) = 1 \\
W(1) &= \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}
\end{aligned} \tag{3}
$$

Theorem 1 *qMPN learning rule does not preserve unitary operators.*

We verified that Algorithm 1 is not a quantum algorithm. The main question in this letter is about the differences between the perceptrons and the qMPN and if the Algorithm 1 presents some advantage when compared with classical neural networks learning algorithm. Before start this analysis we define a classical neuron.

A neuron with inputs $x_1, x_2, \cdots, x_n$ and weights $w_1, w_2, \cdots, w_n$ and linear activation function f has its output y described in (4). One possible weight update rule for an artificial neuron is the Least Mean Square (LMS) rule [1] described in (5).

$$
y = f\left(\sum_{i=1}^{n} x_i \cdot w_i\right) \tag{4}
$$

$$
w_i(t+1) = w_i(t) + \eta \cdot (d - y) \cdot x_i \tag{5}
$$

Now we present an example with two artificial neurons, where the weights of the first neuron are w_{11} and w_{12} and the weights of the second neuron are w_{21} and w_{22}. If the neurons receive an input $\begin{bmatrix} x_1 & x_2 \end{bmatrix}$, then the output of the first neuron will be $y_1 = x_1 w_{11} + x_2 w_{12}$ and the output of the second neuron will be $y_2 = x_1 w_{21} + x_2 w_{22}$.

One can organize the weights in a matrix $W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$, the inputs in a vector $|x\rangle = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$, and the neurons outputs can be calculated exactly as in (1), where $|y\rangle = \begin{bmatrix} y_1 & y_2 \end{bmatrix}^T$. The desired outputs for $|x\rangle$ can be represented with a vector $|d\rangle = \begin{bmatrix} d_1 & d_2 \end{bmatrix}$, where d_i is the desired output of ith neuron, when $|x\rangle$ is presented. We rewrite (5) using a matrix notation and we obtain exactly the learning rule in (2) that was proposed in [8].

The authors also shown the capacity of the qMPN to solve non linearly separable patterns. But to perform this task the values 00, 01, 10, 11 were associated with qubits in the computational basis that are linearly independents.

3 Conclusion

We conclude claiming that for any qMPN one can create an equivalent classical single layer neural network. We also verified that the learning algorithm proposed by Zhou and Ding does not present any advantage over the classical ones and works exactly as the LMS rule.

Acknowledgments This work is supported by research grants from CNPq, CAPES and FACEPE (Brazilian research agencies).

References

1. Haykin, S.: Neural networks: a comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River, NJ (1999)
2. Kak, S.C.: On quantum neural computing. Inf. Sci. **83**(3), 143–160 (1995)
3. Kouda, N., Matsui, N., Nishimura, H., Peper, F.: Qubit neural network and its learning efficiency. Neural Comput. Appl. **14**(2), 114–121 (2005)
4. Li, P., Xiao, H., Shang, F., Tong, X., Li, X., Cao, M.: A hybrid quantum-inspired neural networks with sequence inputs. Neurocomputing **117**, 81–90 (2013)
5. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
6. Panella, M., Martinelli, G.: Neural networks with quantum architecture and quantum learning. Int. J. Circ. Theor. Appl. **39**, 61–77 (2011). doi:10.1002/cta.619
7. da Silva, A.J., de Oliveira, W.R., Ludermir, T.B.: Classical and superposed learning for quantum weightless neural networks. Neurocomputing **75**(1), 52–60 (2012)
8. Zhou, R., Ding, Q.: Quantum m-p neural network. Int. J. Theor. Phys. **46**(12), 3209–3215 (2007)