Quantum Models for Artificial Neural Network

JEAN FABER[1], GILSON A. GIRALDI[1]

[1]LNCC–National Laboratory for Scientific Computing -
Av. Getulio Vargas, 333, 25651-070 Petropolis, RJ, Brazil
{faber,gilson}@lncc.br

Abstract. There has been a growing interest in artificial neural networks (ANNs) based on quantum theoretical concepts and techniques due to cognitive science and computer science aspects. The so called Quantum Neural Networks (QNNs) is an exciting area of research in the field of quantum computation and quantum information. However, a key question about QNNs is what such an architecture will look like as an implementation on quantum hardware. To look for an answer to this question we firstly review some basic concepts in ANNs and emphasize their inherent non-linearity. Next, we analyze the main algorithms and architecture proposed in this field. The main conclusion is that, up to now, there is no a complete solution for the implementation of QNNs. We found partial solution in models that deal with nonlinear effects in quantum computation. The Dissipative Gate (D-Gate) and the Quantum Dot Neural Network are the focused models in this field. The former is a theoretical one while the later is a device composed by a quantum dot molecule coupled to its environment and subject to a time-varying external field. A discretized version of the Feynman path integral formulation for this system can be put into a form that resembles a classical neural network. Starting from these models, we discuss learning rules in the context of QNNs. Besides, we present our proposals in the field of QNNs.

1 Introduction

In the last two decades we observed a growing interest in Quantum Computation and Quantum Information due to the possibility to solve efficiently hard problem for conventional computer science paradigms. Quantum computation and quantum information encompasses processing and transmission of data stored in quantum states (see [15] and references therein).

On the other hand, Artificial Neural Networks (**ANNs**) is a rapidly expanding area of current research, attracting people from a wide variety of disciplines, mainly due to its capabilities for pattern recognition and classification [2].

Simply stated an ANN is a computing system composed by very specialized units called *neurons* which are linked by *synaptic* junctions. Learning is the fundamental feature of ANNs. Learning occurs when modifications are made to the coupling properties between neurons, at the synaptic junction [2].

From this scenario, emerge the field of artificial neural networks based on quantum theoretical concepts and techniques. They are called Quantum Neural Networks (**QNNs**).

The first systematic examination of quantum theory applied to ANNs was done in Menneers PhD thesis [12]. The basic approach is inspired on the multiple universes view of quantum theory: the neural network is seen as a physical system whose multiple occurrences (*component networks*) are trained according to the set of patterns of interest (see Appendix A). The superposition of the trained components gives the final QNN.

Several works about QNNs have been done since Menneer's thesis. Shafee worked with a quantum neural network with nearest neighbor nodes connected by c-NOT gates [17]. Altaisky [1] proposed a quantum *inspired* version of the perceptron - the basic model for neurons in ANNs.

Associative Memory Networks have been also explored in quantum context [6, 3, 9]. The Hopfield net is a nice example. The network can be seen as a physical system instantiated from a set of physical parameters. The many-dimensional phase space corresponding may have a set of local minima. Each one of these critical points is associated with a particular pattern, which can be said to be "stored" by the network physical parameters. This is a way of doing pattern recognition. The advantage of using a quantum Hopfield net is that the number of stable states can be much larger than the classical counterpart, because of quantum superposition of states (section 3) and because the connectivity is much more complex [3].

Gupta at al. [8] defined a new model for QNNs by introducing a nonlinear and irreversible gate (D-Gate). Authors justify the models as a solution for the *localization* problem, that is, the reflection of the computational trajectory, causing the computation to turn around. In another way, D-Gate would be a run-time device (that means, a gate) sensitive to the probability amplitude.

From the point of view of ANNs most of these works shares the same limitation: from the actual state-of-the-art for quantum computers it is not clear the hardware requirements to implement such models.

The Quantum Perceptrons proposed by Lewestein [11] was an attempt at this goal. In this case, differently from [1] cited above, a unitary operator is used to map inputs to outputs. During training the unitary operator is developed to find the correct mapping. Lewestein discussed what a quantum perceptron can learn in terms of the probability of a unitary operator existing for a typical problem. However, he did not specify the internal workings of a quantum perceptron nor describe how the unitary operator may be developed, that is, how learning occurs.

From the point of view of ANNs we observe that a key problem for implementing QNNs is the need of nonlinearity which is an irreversible operations.

Following this observation, we describe the work of Behrman at al. [4]. They used discretized Feyman path integrals and found that the real time evolution of a quantum dot molecule coupled to the substrate lattice through optical phonons, and subject to a time-varying external field, can be interpreted as a neural network.

Starting from this interpretation, we compare this model with the quantum perceptron of Altaisky [1] and discuss the learning rules and nonlinearity in the context of QNNs. Besides, we present our proposals to explore the D-Gate and quantum dot models.

The paper is organized as follows. Section 2 present some ANNs concepts. In section 3 we present basic concepts for quantum computation and discuss nonlinearity for QNNs. The quantum dot molecule model is described in section 4. We present our analysis and further directions for our work in section 5. Finally, we present the conclusions and future works (section 6). Appendix A outlines the basic idea of Menneer's model.

2 Classical Neural Networks

The first logical neuron was developed by W. S. McCulloch and W.A. Pitts in 1943 [2]. It describes the fundamentals functions and structures of a neural cell reporting that a neuron will fire an impulse only if a threshold value is exceeded.

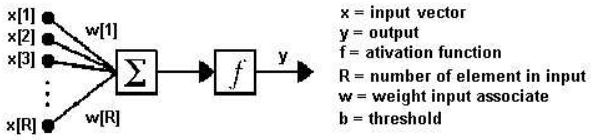

Figure 1: McCulloch-Pitts neuron model.

Figure 1 shows the basic elements of McCulloch-Pitts model: x is the input vector, w are weights associated to the input, y is output, R is number of elements in input and f is the *activation function* that determine the value in output. A simple choice for f is the signal function $sgn(.)$. In this case, if the sum, across all the inputs with its respective weights exceeds the threshold b the output y is 1 else the value of y is -1, that is:

$$y = sgn(\sum_{i=1}^{R} w_i x_i - b). \quad (1)$$

But the McCulloch-Pitts neuron did not have a mechanisms for *learning*. Based on biological evidences, D.O. Hebb suggested a rule to adapt the input weights, which is interpreted as learning rule for the system [2]. This biological inspired procedure can be expressed in the following manner:

$$w_i^{new} = w_i^{old} + \Delta w_i; \quad \Delta w_i = \eta(y^{desired} - y)x_i, \quad (2)$$

where w^{new} and w^{old} are adapted weights and initials weights respectively, η is a real parameter to control the rate of learning and $y^{desired}$ is the desired (know) output. This *learning rule* plus the elements of Figure 1 is called the perceptron model for a neuron.

Then, the learning typically occurs for example through training, or exposure to a know set of input/output data. The training algorithm iteratively adjusts the connection weights $\{w_i\}$ analogous to synapses in biological nervous. These connection weights store the knowledge necessary to solve specific problems.

3 Quantum Computation and Nonliearity

In practice, the most useful model for quantum computation is the Quantum Computational Network also called Deutsch's model [8, 16]. The basic information unit in this model is a *qubit* [14], which can be considered a superposition of two independent states $\mid 0\rangle$ and $\mid 1\rangle$, denoted by $\mid \psi\rangle = \alpha_0 \mid 0\rangle + \alpha_1 \mid 1\rangle$, where α_0, α_1 are complex numbers such that $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

A composed system with n qubits is described using $N = 2^n$ independent states obtained through the tensor product of the Hilbert Spaces associated with the qubits. Thus, the resulting space has a natural basis that can be denoted by:

$$\{\mid i_0 i_1 ... i_{n-1}\rangle; \quad i_j \in \{0, 1\}\}. \quad (3)$$

This set can be indexed by $\mid i\rangle; \quad i = 0, 1, ..., N-1$. Following the Quantum Mechanics Postulates, the system state $\mid \psi\rangle$, in any time t, can be expanded as a *superposition* of the basis states:

$$\mid \psi\rangle = \sum_{i=0}^{N-1} \alpha_i \mid i\rangle; \quad \sum_{i=0}^{N-1} |\alpha_i|^2 = 1. \quad (4)$$

Entanglement is another important concept for quantum computation with no classical counterpart. To understand it, a simple example is worthwhile.

Let us suppose that we have a composed system with two qubits. According to the above explanation, the resulting Hilbert Space has $N = 2^2$ independent states.

Let the Hilbert Space associated with the first qubit (indexed by 1) denoted by H_1 and the Hilbert Space associated with the second qubit (indexed by 2) denoted by H_2. The computational basis for these spaces are given by: $\{\mid 0\rangle_1, \mid 1\rangle_1\}$ and $\{\mid 0\rangle_2, \mid 1\rangle_2\}$, respectively. If qubit 1 is in the state $\mid \psi\rangle_1 = a_{10} \mid 0\rangle_1 + a_{11} \mid 1\rangle_1$ and qubit 2 in the state $\mid \psi\rangle_2 = a_{20} \mid 0\rangle_2 + a_{21} \mid 1\rangle_2$, then the composed system is in the state: $\mid \psi\rangle = \mid \psi\rangle_1 \otimes \mid \psi\rangle_2$, explicitly given by:

$$\mid \psi\rangle = \sum_{i,j \in \{0,1\}} a_{1i} a_{2j} \mid i\rangle_1 \otimes \mid j\rangle_2. \tag{5}$$

Every state that can be represented by a tensor product $\mid \psi\rangle_1 \otimes \mid \psi\rangle_2$ belongs to the tensor product space $H_1 \otimes H_2$. However, there are some states in $H_1 \otimes H_2$ that can not be represented in the form $\mid \psi\rangle_1 \otimes \mid \psi\rangle_2$. They are called *entangled states.* The Bell state (or *EPR pair*) presented next is a very known example:

$$\mid \psi\rangle = \frac{1}{\sqrt{2}} \left(\mid 0\rangle_1 \otimes \mid 0\rangle_2 + \mid 1\rangle_1 \otimes \mid 1\rangle_2 \right). \tag{6}$$

Trying to represent this state as a tensor product $\mid \psi\rangle_1 \otimes \mid \psi\rangle_2$, with $\mid \psi\rangle_1 \in H_1$ and $\mid \psi\rangle_2 \in H_2$, produces an inconsistent linear system without solution.

Entangled states are fundamental for teleportation [7, 14]. In recent years, there has been tremendous efforts trying to better understand the properties of entanglement, not only as a fundamental resource for the Nature, but also for quantum computation and information. In Appendix A, we describe a quantum neural network model inspired in this property.

The computation unit in Deutsch's model consists of quantum gates which are unitary operators that evolves an initial state performing the necessary *computation* to get the desired result. A quantum computing algorithm can be summarized in three steps: (1) Prepare the initial state; (2) A sequence of (universal) quantum gates to evolve the system; (3) Quantum measurements.

From quantum mechanics theory, the last stage performs a *collapse* and only what we know in advance is the probability distribution associated to the measurement operation. So, it is possible that the result obtained by measuring the system should be post-processed to achieve the target (quantum factoring (Chapter 6 of [16]) is a nice example).

Let us return to the perceptron model of section 2. Would it be possible to implement a quantum mechanics device analog to it?

Just as a matter of setting ideas, let's take the quantum *inspired* perceptron model proposed in [1]. In this model a quantum system with n input qubits $\mid x_0\rangle, \mid x_1\rangle, ..., \mid x_{n-1}\rangle$ is considered and an output is derived by the rule:

$$\mid y\rangle = \sum_{j=0}^{n-1} \breve{U}_j \mid x_j\rangle \tag{7}$$

where $\breve{U}_j$, F are 2×2 operators acting on the basis $\{\mid 0\rangle, \mid 1\rangle\}$. In analogy with the classical perceptron, the following learning rule is proposed:

$$\breve{U}_j(t+1) = \breve{U}_j(t) + \eta \left(\mid d\rangle - \mid y(t)\rangle \right) < x_j \mid \tag{8}$$

where $\mid d\rangle$ is the desired output. It can be shown [1] that the above rule drives the system into the desired state $\mid d\rangle$.

From the quantum mechanics point of view, the first problem of the above system is that the learning rule in expression (8) is not an unitary operation in general (the same is true for expression (7)). That is way we call this model *quantum inspired*.

Besides, ANNs need activation functions, which are scalar and nonlinear function, to be implemented. Nonlinearity effects in quantum computation are discussed by Gupta at al. [8] when proposing a new gate, a dissipative one, called *D-Gate*.

The behavior of the D-Gate is the following: given the state system:

$$\mid \psi\rangle = \sum_{i=0}^{N-1} \alpha_i \mid i\rangle, \tag{9}$$

let $A(\mid i\rangle)$ and $A'(\mid i\rangle)$ respectively denote the probability amplitudes before and after the application of the D operator. Then, if $A(\mid 0\rangle) > \delta \Rightarrow A'(\mid 0\rangle) = c$ and $A'(\mid 0\rangle) = 0$ otherwise; where c for probability amplitude denotes some constant used for encoding 1. The parameter δ is a pre-set threshold.

From the point of view of Gupta at al. [8] we could postulate a quantum neural network constructed from Unitary operators and the D-Gate. In the network representation, the quantum gates are interconnected by wires indicating the information flow during the computation (Figure 2). By convention, the computation proceeds from left to right.

However, a quantum mechanics feasible learning rule should be designed. This point is not addressed by Gupta at al. To answer this question we need a more deeply consideration about the D-Gate and its hardware implementation. This is the starting point of this work.

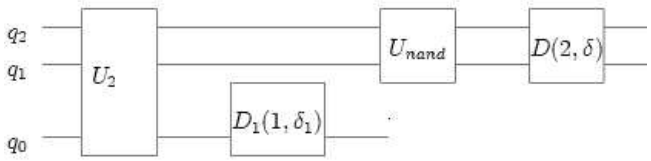

Figure 2: Outputs are connected to Gate inputs in the network.

The D-Gate nonlinearity is due to dissipations. Such irreversible operation can be implemented if full interactions with the environment is taken into account. The behavior of a system can also be nonlinear because of the interactions between its degrees of freedom (see sections 5, 6 of [8]).

But, what kind of physical system in quantum mechanics can perform nonlinear operations?

What about learning rules? The expression (8) gives a rule that adapts operators which evolves the state of the system. However, in the classical perceptron, the quantities affected by the learning rule (2) are system parameters! Quantum mechanics systems have in general a set of predefined parameters. Could be a learning rule that adapt system parameters more feasible in practice? Is there such a rule?

We believe that a possible (may be partial) solution for these questions is the model stated next.

4 A Quantum Dot Neural Network

In [4] we found a mathematical formulation of a quantum neural network through a quantum dot molecule coupled to the substrate lattice through optical phonons, and subject to a time-varying external field. The nonlinearity is a consequence of the real-time propagation of a quantum system coupled to its environment. Dissipation is not considered here, although the general model can incorporate this possibility [10, 4].

Using discretized Feynman path integrals, authors found that the real time evolution of the system can be put into a form which resembles the equations for the *virtual* neuron activation levels of an artificial neural network. The timeline discretization points serve as virtual neurons.

Through the Feynman path integral formulation of quantum mechanics we can write the expression for the time evolution of the quantum mechanical state of a system as:

$$\mid \psi\left(x_f,T\right)\rangle = \int_{(x_0,0)}^{(x_f,T)} D\left[x\left(t\right)\right] \times$$

$$\exp\left(\frac{i}{\hbar}\int_0^T d\tau\left[\frac{1}{2}m\,\dot{x}\,-V\left(x\right)\right]\right)\mid \psi\left(x_0,0\right)\rangle. \qquad (10)$$

Expression (10) is equivalent to the following one:

$$\lim_{N\to\infty} \int_{(x_0,0)}^{(x_{N+1}x_f,T)} dx_1 dx_2 ... dx_N \left(\frac{m}{2\pi i\hbar\Delta t}\right)^{(N-1)/2} \times$$

$$\exp\left(\sum_{j=0}^{N}\left[\frac{im\Delta t}{2\hbar}\left(\frac{x_{j+1}-x_j}{\Delta t}\right)^2 - V\left(x_j\right)\right]\right)\mid \psi\left(x_0,0\right)\rangle \qquad (11)$$

Here $\mid \psi\left(x_0,0\right)\rangle$ is the input state of the quantum system at time $t=0$ and $\mid \psi\left(x_f,T\right)\rangle$ is the output state at time $t=T$. In this equation, m is the mass, $2\pi\hbar$ is Planck's constant, and V is the potential energy. In the second line, the paths are discretized: $N\Delta t = T$, with the number of discretization points, $N\to\infty$.

4.1 Specifying the System

The formulation above is general. A QNN approach comes out when the system is that of a quantum dot molecule with five dots arranged as the pips on a playing card. The dots are close enough to each other that tunneling is possible between any two neighbors. Two electrons are fed into the molecule, which then has a doubly-degenerate ground state (in the absence of environmental potentials). These states can be thought of as the polarization P of the molecule, equal to ± 1, that is, the Pauli matrix operator σ_z. In Equation (10) this would be the value $x\left(t\right)$.

In addition to adjusting or training $V\left(x\right)$, we can obtain an additional trainable nonlinearity by coupling the quantum system to its environment. The environment is modeled by a set of Gaussians, that is, the environment has a quadratic Hamiltonian, or, equivalently, a normal distribution; if the set is taken to be infinite, any desired influence including dissipation can be produced. In this model this would be represented by the coupling between the electronic state of the dot molecules and the lattice through optical phonons. Physically the coupling would have to be weak enough to be represented accurately as linear; for example, GaAs substrate satisfies this, with a (unitless) electron-phonon coupling parameter of $0.08 << 1$.

Instead of taking $N\to\infty$ like in (11), we take N to be finite (quasi-continuum). Equation (11) becomes:

$$\mid \psi\left(\sigma_z\left(N\Delta t\right),T\right)\rangle =$$

$$\sum_{\{\sigma_z(j\Delta t)\}} \exp\left(\frac{i}{\hbar}\sum_j\left[K\sigma_x\left(j\Delta t\right)+\epsilon\left(j\Delta t\right)\sigma_z\left(j\Delta t\right)\right]\right)\times$$

$$I\left[\sigma_z\left(\Delta t\right)\right] \mid \psi\left(\sigma_z\left(0\right),0\right)\rangle \tag{12}$$

where the path integral over possible positions at each time, $x(t)$, has been written as a finite set of sums over states of the polarization, σ_z, at each time slice $j\Delta t$. Also, at each time slice the polarization can be either $+1$ or -1. The potential energy V comes from a time-varying electric field, $\epsilon\left(t\right)$, and the kinetic energy term, in this two-state basis, now has the form $K\sigma_x\left(j\Delta t\right)$, where σ_x is the Pauli matrix. Since σ_x is off-diagonal in the polarization basis, this term contains the (nonlinear) coupling between the states of the quantum dot molecule at successive time slices. The size of this term, given by the parameter K (the tunneling amplitude), is determined by the physics of the dot molecule: how easy it is for the electrons to tunnel from polarization state $+1$ to -1. The effect of the optical phonons is summarized by the influence functional $I\left[\sigma_z\left(t\right)\right]$, given by:

$$I\left[\sigma_z\left(t\right)\right] = \int \prod_k D\left[\alpha_k\left(t\right)\right] \exp\left(\frac{i}{\hbar}\int_0^T d\tau \sum_k S_k\right) \tag{13}$$

where

$$S_k = \frac{m_k}{2}\,\dot{\alpha}_k^2\left(\tau\right) + \frac{m_k\omega_k^2}{2}\alpha_k^2\left(\tau\right) + \lambda_k\alpha_k\left(\tau\right)\sigma_z\left(\tau\right),$$

and α_k is the position variable of the k^{th} harmonic oscillator (phonon), m_k its mass, ω_k its frequency, and λ_k its coupling strength to the system. The advantage of a linearly coupled harmonic bath is that the path integrals over the phonons can be performed immediately, giving us the nonlinear functional:

$$I\left[\sigma_z\left(t\right)\right] = \exp\left(\sum_j\sum_{j'}\sigma_z\left(j\Delta t\right)\chi\left(j\Delta t, j'\Delta t\right)\sigma_z\left(j'\Delta t\right)\right) \tag{14}$$

where $\chi\left(\tau,\tau'\right) = \chi\left(\left|\tau-\tau'\right|\right) = \chi\left(\tau''\right)$ is the influence phase, proportional to the response function of the bath. For the phonon bath,

$$\chi\left(\tau\right) = \sum_k \frac{\lambda_k^2}{2m_k\omega_k}\cosh\left(\frac{\beta\hbar\omega_k}{2}\right)\times$$

$$\left[\cosh\left(\frac{\beta\hbar\omega_k}{2}\right)\cos\left(\omega_k\tau\right) + i\sinh\left(\frac{\beta\hbar\omega_k}{2}\right)\sin\left(\omega_k\tau\right)\right] \tag{15}$$

where it was introduced also a (suitably low) temperature, given by $1/\beta$ in units of Boltzmanns constant.

In [4] authors consider the obtained N intermediate states to be the states of N *virtual* quantum neurons, one at each time slice $j\Delta t$. The nonlinearity necessary for neural computation is inherent in the kinetic energy term, $(x_{j+1}-x_j)$, and in the exponential. Each of the N neurons different possible states contributes to the final measured state; the amount it contributes, can be adjusted by changing the potential energy, $V\left(x\right)$.

The trainable parameters set can be any of those that appear above (λ_k,ω_k) or even the values of the electric field at each time slice j $\{\epsilon\left(j\Delta t\right); j=0,...,N\}$. Combinations of these sets are also possible to be trained. It is important to emphasize that any of these parameters can be controlled physically [4].

4.2 Training the Quantum Network

We now set up a simulation of the quantum neural network. We specify as inputs the initial $(t=0)$ polarizations of each of two quantum dot molecules, far enough from each other spatially that they do not interact directly, but sharing the same substrate.

The system output ($\mid\psi\left(\sigma_z\left(N\Delta t\right),T\right)\rangle$) represents a combination of the basic states of polarization, say $\mid+\rangle$ and $\mid-\rangle$. To define a training rule we have to define a scalar function of the system output whose value is thresholded to decide if the reproduce the *desired* behavior (a quantum logic gate, for example).

In [4] the polarization of the first molecule at the final time is arbitrarily taken. Thus, the probability amplitude for the first molecules final state to be equal to the $\mid+\rangle$ state is computed (give by $\left|<+\mid\psi\left(\sigma_z\left(N\Delta t\right),T\right)\rangle\right|^2$) and the signal of the following expression considered:

$$Out = \left|<+\mid\psi\left(\sigma_z\left(N\Delta t\right),T\right)\rangle\right|^2 - Desired \tag{16}$$

if $Out = 0$ the network is considered to be trained. To achieve this goal, an *Error Function* is defined and a gradient descent algorithm was used for training:

$$Error = \frac{1}{4}Out^2; \quad \lambda_k^{new} = \lambda_k^{old} - \eta\frac{\partial Error}{\partial\lambda_k}, \tag{17}$$

where

$$\frac{\partial Error}{\partial\lambda_k} = \frac{1}{2}Out\left[2Re\left[\frac{\partial<+\mid\psi\rangle}{\partial\lambda_k}<+\mid\psi\rangle^*\right]\right] \tag{18}$$

We shall emphasize that it is possible to train purely quantum gates such as a phase shift, because the network is quantum mechanical.

5 Discussion

Firstly, let us compare equations (7) and (12). We shall observe that instead of saying that we have "N quantum

neurons " we could say that we have a kind of quantum perceptron.

Other point to be considered is the training stage. Basically, the rule given by equation (17) updates the parameters of the system, which is much more closer to a neural network approach than the expression (8).

Despite of these advantages, a doubt about this QNN is that its neural network approach is *virtual* in the sense that it is just a biased interpretation of an approximated model of the quantum system.

Although this argument may be consistent, we do not believe that it discards the model because a neural network should have three basic elements: (1) An *operator* to compose the signal(s) received (equation (12) in the above model); (2) A *test* to decide if the results is the desired one (expression (16)); (3) A rule to adapt parameters if need (equation (17)). These relations are not virtual in the sense that the final result is a system that reproduce the desired behavior (a gate, for example).

Up to our knowledge, this model is the first one to encompass all steps (1)-(3). If compared with the double-slit experiment (Figure 3), which Narayanan and Menneer [13] argued that provides the basis for generic quantum neural networks, we observe that the quantum dot model is much more complete.

The double-slit experiment is described as follows (Figure 3). A photon gun sends photos to a double slit apparatus.. The behavior of the system can be explained in quantum mechanical terms through the duality wave-particle: the particle travels as a wave but collapses to a particle (point) when the detection screen is achieved.

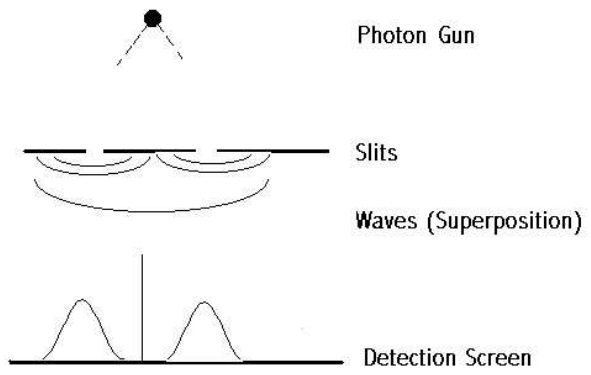

Figure 3: Double slit experiment. A photon gun send particles to the double slit aparatus. Photons travel as waves but collapses to a particle when colliding with the screen detector.

The proposed idea in [13] is that the input pattern replaces the photon , input nodes replace the slits, and the connections between the input nodes and the next layer are the waves created by the superposition of patterns which are evolved by unitary operators. The output nodes act as a detection screen, in which the collapse occurs.

Despite of some isomorphism between the experiment and ANNs, neither the step (2) nor the step (3) are clearly contemplated in this explanation.

In the Appendix A we analyse other aspects of Narayanan and Menneer ideas.

6 Conclusions

QNNs is a promising area in the field of quantum computation and quantum information. Several models have been proposed in the literature but for most of then it was not clear the hardware requirements to implement such models.

In this paper we analyze a physical system -Quantum Dot Molecule - that can be interpreted as an implementation of a QNN. We compare this model with a quantum *inspired* perceptron [1] and analyze it from the neural networks point of view. We believe that quantum dot molecules are promising devices for QNNs.

An important question is if the quantum dot system can be trained to work like a D-Gate as we can incorporate dissipation effects. Besides, we are interested to analyze the potential of quantum dot arrays [3], from the point of view of the above analysis. These are further directions of our work.

7 Acknowledgments

We would like to acknowledge CNPq for the financial support for this work.

References

[1] M. V. Altaisky. Quantum neural network. Technical report, http://arxiv.org/PS_cache/quant-ph/pdf/0107/0107012.pdf, 2001.

[2] R. Beale and T. Jackson. *Neural Computing*. MIT Press, 1994.

[3] E. Behrman, L. Nash, J. Steck, V. Chandrashekar, and S. Skinner. Simulations of quantum neural networks. *Information Sciences*, (128):257–269, 2000.

[4] E. Behrman, J. Niemel, and J. Steck. A quantum dot neural network. Technical report, www.engr.twsu.edu/me/faculty/steck/pubs/ieeenn96.ps, 1996.

[5] H. Everett. Relative state formulation of quantum mechanics. *Reviews of Modern Physics*, 29:454–462, 1957.

[6] A. Ezhov, A. Nifanova, and Dan Ventura. Quantum associative memory with distributed queries. *Information Sciences*, (128):271–293, 2000.

[7] A. Furusawa, J. Sorensen, S. Braunstein, C. Fuchs, H. Kimble, and E. Polzik. Unconditional quantum teleportation. *Science*, 282(706), 1998.

[8] S. Gupta and R. Zia. Quantum neural networks. Technical report, http://www.arxiv.org/PS_cache/quant-ph/pdf/0201/0201144.pdf, 2002.

[9] J. Howell, J. Yeazell, and Dan Ventura. Optically simulating a quantum associative memory. *Physical Review A*, 62, 2000.

[10] C. Lent, D. Tougaw, and W. Porod. Quantum cellular automata: The physics of computing with arrays of quantum dot molecules. In *Proc. of the Workshop on Physics and Computation–PhysComp94. Also available on http://www.nd.edu/∼qcahome/reprints/archive.htm*, pages 5–13, 1994.

[11] M. Lewenstein. Quantum perceptrons. *Journal of Modern Optics*, 41, 1994.

[12] T. Menneer. *Quantum Artificial Neural Networks*. PhD thesis, University of Exeter, 1998.

[13] A. Narayanan and T. Menneer. Quantum artificial neural network architectures and components. *Information Sciences*, (128):231–255, 2000.

[14] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press., December 2000.

[15] M. Oskin, F. Chong, and I. Chuang. A practical architecture for reliable quantum computers. *IEEE Computer*, 35(1):79–87, 2002.

[16] J. Preskill. Quantum computation - caltech notes. Technical report, www.theory.caltech.edu/people/preskill/ph229/#lecture, 2001.

[17] Fariel Shafee. Entangled quantum networks. Technical report, http://arxiv.org/ftp/quant-ph/papers/0203/0203010.pdf, 2002.

8 Appendix A - Menneer Quantum Inspired Model

This QNN model can be summarized as follows.

Training: Given a training set of N patterns, a set of N homogeneous components is set up which later will form the component parts of the network. Each training pattern is channelled into one component, and the set of weights encompassed by this component is changed to learn this, and only this, training pattern.

Testing: After training, a method is required for processing a new input. This method is based on a superposition of the trained networks and collapse.

The following example shows how the method works.

Let us take a single-layer neural network with four input units and one output unit (it is equivalent to a perceptron (section 2)). In this example, we have a set of $N = 3$ input patterns given by: $1111, 1010, 0001$.

Thus, three component networks (three perceptrons) are trained: component *a* for pattern 1111, *b* for input pattern 1010 and component *c* for input pattern 0001.

The input units are numbered from 1 to 4 and the links between the input units and the output unit are labeled according to the rule *Component-Input Unity:* $a1, b1, c1; a2, b2, c2$;etc.

After the component networks have learned their corresponding patterns, their weight values form the superpositional weights in the QNN. For example, in the case of Figure 4, the link from the input unit 1 in the QNN will have a quantum weight of the superpositional composition of the component weights $a1$, $b1$ and $c1$.

	Weights to the output unit from the input unit				Training input to input unit			
	1	2	3	4	1	2	3	4
CN a	a1	a2	a3	a4	1	1	1	1
CN b	b1	b2	b3	b4	1	0	1	0
CN c	c1	c2	c3	c4	0	0	0	1

Figure 4: Training inputs and resulting weights for each component network (CN).

Collapse of the QNN involves to reduce each superposition of weights to a single weight. However, a special care is taken to avoid mixing weights of different components. Now, the idea of entanglement is used: The superposition of weights are entangled such that collapse occurs to one set of weights from one component network.

Collapses occurs during test. In this step, the test input is compared to the training input for each input unit.

For example, if there is an input of 1010:
1 is input to unit 1, giving the superposition 1a1 and 1b1

and 0c1;
0 is input to unit 2, giving 0a2 and 1b2 and 1c2;
1 is input to unit 3, giving 1a3 and 1b3 and 0c3;
0 is input to unit 4, giving 0a4 and 1b4 and 0c4.

The component weight-set that is used to process the test input is the one with the greatest summed coefficient (component b).

From the point of view of ANN, the above proposal has the advantage of avoiding the known *Catastrophic Forgetting* because interference between the learning patterns do not occur due to the fact that each training pattern has its own component.

Recently, Ezhov [6] argues that this approach to quantum neural networks is consistent with the Everett [5] parallel universe interpretation of quantum mechanics.

However, from the viewpoint of quantum computation, we shall say that the above model is only quantum-inspired because: (1) There is no an explicit superpositional composition of the component weights; (2) The entanglement stated is really a lookup table (component,training pattern) instead of a quantum mechanics operation.

To be more specific, let us take the known problem of finding the period r of a periodic function $f : Z_N \rightarrow Z$, where Z_N denotes the additive group of integers modulo N.

In this case, the quantum solution found in the literature [14] starts from a hardware with two registers (a register is a composite system made of a set of qubits) in the following entangled state:

$$\mid \Psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \mid x\rangle \otimes \mid f(x)\rangle. \tag{19}$$

Thus, when measuring the value in the second register, giving, say, a value y_0, then the first register 's state will collapse to an uniform superposition of all those $\mid x\rangle's$ such that $f(x) = y_0$; that is:

$$\mid \Psi\rangle_{after} = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} \mid x_0 + kr\rangle, \tag{20}$$

where x_0 is such a x and $N = Kr$.

In this case, expression (19) is an *explicit* superposition of states. $\mid \Psi\rangle$ is an entangled state because it can not be expressed as tensor product (see section 3). The desired effect in this case is to cause each integer values of x to be entangled with the corresponding value $f(x)$. Thus, by the postulates of the quantum mechanics, the collapse occurs after measuring.

Certainly, it is exactly the desired behavior when linking the component network and the training pattern. However, it is not clear how to put the algorithm proposed by Menneer in a quantum mechanical formulation, like expressions (19)-(20).