Single-shot learning algorithm for quantum weightless neural networks

Adenilton J. da Silva and Teresa B. Ludermir
Centro de Informática
Universidade Federal de Pernambuco
Recife/PE, Brazil 50740-560
Email: {ajs3,tbl}@cin.ufpe.br

Wilson R. de Oliveira
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco
Recife/PE, Brazil 52171-900
Email: wrdo@gmail.com

***Abstract*—In this paper we present a new quantum learning algorithm for quantum weightless neural networks that is based on quantum superposition principle. The main characteristic of this quantum learning algorithm is its capacity to train a neural network with only one execution of the neural network model. The proposed algorithm is denoted Single Shot quantum Learning Algorithm (SSLA). SSLA is based in the GSN learning algorithm that also requires a single presentation of the train data to the network and in a superposition quantum based learning algorithm.**

I. Introduction

Weightless neural networks(WNNs) were firtly proposed by Aleksander [1], [2] and are a simple model of artificial neural network. Inputs and outputs of WNNs are normally Boolean values and WNNs are also known as Boolean neural networks. Besides its simplicity, WNNs are powerful classifiers and have been used in several applications [3], [4], [5], [6]. Theoretical results show that one can simulate a probabilistic automata with weightless neural networks [7] and with an auxiliary data structure one can simulate Turing machines. In this paper we work with a quantum version of a WNN.

Quantum computation was firstly proposed by Feynman in [8]. He showed that a quantum computer can simulate quantum systems exponentially faster than a classical computer. Factorization and search are two others algorithms that have quantum versions more efficient than any classical known algorithm [9], [10].

In [11] weightless neural networks and quantum computing are used to firstly propose a quantum weightless neural network. Two types of quantum neural networks qPLN and qMPLN were presented, but their learning algorithms does not use quantum computing properties. A model of quantum neuron based in the RAM node is proposed in [12] but the learning algorithm is not presented in the paper. In [13] a quantum network composed by quantum RAM nodes or qRAM nodes is proposed. In this work a learning algorithm that uses principles of quantum computation is presented. The main characteristic of this learning algorithm, called Superposition based Learning (SL) algorithm, is its capacity to put all the patterns in training set in superposition and present then to the neural network simultaneously.

The idea of training a weightless neural network composed by goal seeking neurons (GSN) [14] with only one presentation of the training set is not a new one. Here we propose a quantum algorithm to train weightless neural networks where one need to run the neural network only once. The proposed algorithm is based in the superposition based learning algorithm, where one can present all the patterns in the training set simultaneously to a weightless neural network. Here we use a nonlinear quantum circuit and perform some modifications in the SL algorithm to show that one can train a weightless neural network with only two neural network runs.

The remainder of this paper is organized in four sections. In Section II we present some weightless networks and its learning algorithms. In Section III the principles of quantum computation necessary to understand this work are presented. In Section VI the main result of this paper is presented. We present a quantum learning algorithm to train a qRAM network where (like in GSN networks) one need of only one presentation of train data to the neural network. Finally Section VIII is the conclusion.

II. Weightless neural networks

The RAM based neural networks were proposed by Igor Aleksander [1] and do not have weights associated in their connections.

A RAM node with n inputs has 2^n memory locations, addressed by the n-bit string $a = (a_1 a_2 \dots a_n)$. A binary signal $x = (x_1 x_2 \dots x_n)$ on the input lines will access only one of these locations resulting in $y = C[x]$ [15]. In Figure 1, s and d are respectively the learning strategy and the desired output to be learned.

Learning in weightless neural networks takes place simply by writing into the corresponding look-up table entries. This learning process is much simpler than the adjustment of weights. In spite of the simplicity of the RAM based nodes, RAM based networks have good generalisation capabilities [16] and computational power [17].

Goal Seeking Neuron (GSN) is based on the RAM node. The main characteristic of a GSN node is that its learning requires only one presentation of patterns in the training set (one-shot learning), so its training step is very fast. The GSN is a RAM node where one can store a two bit numbers (00, 11 or 10) that can be interpreted respectively as 0,1 and u.

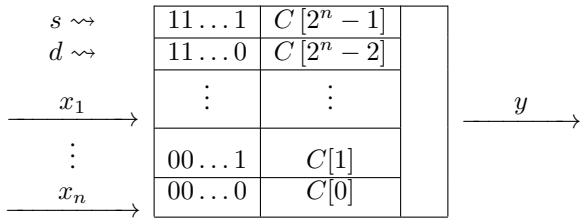

Fig. 1. RAM node

That GSN also can receive as input and produce outputs the values 0,1 and u.

If at least one input has the value u the GSN will access more than one memory position simultaneously. For instance if a GSN with 3 inputs receives the input signal $I = \{0, u, 0\}$, because of the u value in the second position of the input signal I, two memory locations with address 000 and 010 will be accessed. The output of a GSN will be 0 if the majority of accessed memory locations store 0, 1 if the majority of the memory locations store 1, and u otherwise.

The GSN node can be in 3 different operation states: Validation state, where the neuron verifies if a new pattern can be learnt; the learning state, where changes in the node memory positions occurs; and the utilization state, where the node is used to produce outputs for new patterns.

Algorithm 1 presents GSN learning algorithm. In the first step all the memory content are initialized with the undefined value u. When a pattern can be learnt the output neuron searchs for an accessed memory position that has the same value than the desired output. If this value is not found a memory position with the u value has its content changed to store the desired value $d(p)$. When the memory position is choosed, the desired output of the previous layer is defined.

Algorithm 1: GSN learning algorithm

```
Initialize the content of all memory positions with the value u
foreach pattern p in the training set do
    network state ← validation
    The network receives the pattern p and produces the output s
    if s = u or s = d(p) then
        network state ← learning
        Allow that the network learns the desired output d(p)
    end
end
```

III. Quantum computation

The fundamental unit of information in quantum computation is the quantum bit (or qubit). A qubit is represented by a bideminsional, unitary, complex vector in a complex vectorial space $\mathbb{V}$. A qubit is represented in Equation 1, where $\alpha, \beta \in \boldsymbol{C}$, $|\alpha|^2 + |\beta|^2 = 1$, $|0\rangle = [0,1]^T$ and $|1\rangle = [1,0]^T$. The notation $|\cdot\rangle$ is called Dirac notation and is used to represent vectors.

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

To represent systems with multiple qubits we use the tensor product $\otimes$. The tensor product of two qubits $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|\theta\rangle = \vartheta|0\rangle + \gamma|1\rangle$ is described in Equation 2. This process can easily be generalized for realizes a product between m and n dimensional vectors.

$$|\psi\rangle \otimes |\theta\rangle = |\psi\rangle|\theta\rangle = \begin{bmatrix} \alpha \cdot \begin{bmatrix} \vartheta \\ \gamma \end{bmatrix} \\ \\ \beta \cdot \begin{bmatrix} \vartheta \\ \gamma \end{bmatrix} \end{bmatrix} \tag{2}$$

Given a basis, a unitary operator in a vector space $\mathbb{V}$ can be represented by a matrix U such that $\mathsf{U}\mathsf{U}^\dagger = \mathsf{I}$. Quantum operators of qubits in a vector space $\mathbb{V}$ are unitary transformations $\mathsf{U} : \mathbb{V} \to \mathbb{V}$. In Equation 3 some quantum operators are presented.

$$\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \mathsf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \tag{3}$$

The CNOT-gate is an example of a two-qubit controlled operation. It also goes under the name (quantum) XOR. Its matrix representation in the computational basis is

$$\mathsf{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4}$$

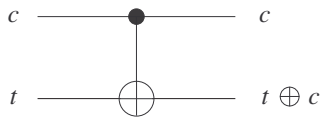

Fig. 2. Controlled NOT gate.

The CNOT gate performs a NOT (i.e. an X) operation on the target qubit t conditioned on the control bit c being 1.

In a quantum computer one can work with a quantum bit in a continuous linear combination of $|0\rangle$ and $|1\rangle$. A qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is a state in superposition and can be seen as part $|0\rangle$ and part $|1\rangle$. One property of quantum computation is the parallelism. Applying a quantum operator U_f, such that $\mathsf{U}_f|x, 0\rangle = |x, f(x)\rangle$ in a state in superposition $\sum_{i=0}^{n-1} \alpha_i|x_i, 0\rangle$, the value of $f(x)$ will be calculated for all x_i. We will use the quantum parallelism to put neural networks configurations in superposition.

One cannot direct see all the results $f(x_i)$ calculated in superposition.In quantum physics if a system is in a superposition $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ measuring the system will

probabilistically collapses it to one basis state $|i\rangle$ as described in Equation 5.

$$p\left(|i\rangle\right) = \frac{|\alpha_i|^2}{|||\psi\rangle||^2} = \frac{|\alpha_i|^2}{\sum_j |\alpha_j|^2} \tag{5}$$

After the first measurement the state collapses to a basis state, others measurements will give always the same result. Because of this property one cannot see all the results calculated in a state in superposition with quantum parallelism. Quantum algorithms needs to explore parallelism before a measurement.

IV. Grover's algorithm

Grover proposed in [10] a quantum algorithm for search in one unordered data set which is quadratically faster than any classical algorithm. To perform a search in an unordered data set with N items one will need in the worst case of N classical steps. In the quantum case one will need of $T = \frac{\pi}{4}\sqrt{\frac{N}{M}}$ steps. Algorithm 2 presents the Grover's algorithm, it is based on the one shown in [18] and M is a squared matrix whose each entry is $1/2^n$.

Algorithm 2: Grover's algorithm

1. Initialize the system $|\psi\rangle = |0\rangle_n$
2. Apply the Hadamard transform $|\psi\rangle = H^{\otimes n}|\psi\rangle$
3. Apply the phase inversion operation: $U_f(I \times H)$
4. Apply the inversion about the mean operation: $-I + 2M$
5. Repeat steps 3 and 4, $T = \sqrt{2^n}$ times
6. Measure the state $|\psi\rangle$

V. Related work

A quantum perceptron has proposed by Altaysky in [19]. In Altaisky definition a quantum perceptron $\mathbf{N}$ as described as in Equation 6, where $\hat{F}$ is quantum operator over 1 qubit representing the activation function, $\hat{w}_j$ is a quantum operator over a single qubit representing the j-th weight of the neuron and $|x_j\rangle$ is one qubit representing the input associated with $\hat{w}_j$.

$$|y\rangle = \hat{F}\sum_{j=1}^{n} \hat{w}_j |x_j\rangle \tag{6}$$

In [19] a learning rule for the Altaisky quantum perceptron has been proposed and it is showed that the learning rule drives the the perceptron to the desired state $|d\rangle$. In learning rule described in Equation 7 it is suposed that $\hat{F} = I$. This learning rule described in 7 does not preserve unitary operators.

$$\hat{w}_j(t+1) = \hat{w}_j(t) + \eta \cdot \left(|d\rangle - |y(t)\rangle\right)\langle x_j| \tag{7}$$

Another model of quantum perceptron has been proposed in [20]. The main difference between Zhou's quantum perceptorn and Altaisky's quantum perceptron is the weight representation. In Zhou's quantum perceptron the weights of a neuron with n inputs is represented by a quantum operator over n qubits. So a Zhou quantum percetron with n inputs necessarily has n^2 weights. The ZQP is described in equation 8, where $\mathbf{Z}$ is any quantum operator. The weights of the ZQP are the elements of matricial representation of $\mathbf{Z}$ operator.

$$|y\rangle = \mathbf{Z}|x_1 \cdots x_j\rangle \tag{8}$$

The main problem with this strategy is that the neuron output $|y\rangle$ has the same number of qubits of the neuron input. In a example presented in [20] the measure of the last qubit of state $|y\rangle$ is used as neuron output, but one can also use another qubit or several qubits on the $|y\rangle$ state. Another question is high number of weights, a quantum operator over n qubits is represented by a complex matrix with n^2 elements, then the number of weights in ZQP is quadratic in relation to the number of inputs. One can easily see in [20] that ZQP learning rule also do not preserve unitary operators.

In [21] a quantum neural network with a quantum architecture is proposed. The quantum neuron proposed has the structure to be trained with a quantum algorithm, but it was not shown how to train the quantum model with classical algorithms. As in this work, the learning algorithm proposed uses nonlinear quantum operators and maybe cannot be practically used even if a one can construct a quantum computer.

VI. qRAM

A quantization of the RAM node is presented in [12] and in [13] a learning algorithm based in the principle of quantum superposition, denoted Superposition based Learning Algorithm (SLA), is presented. The main drawback in SLA is the use of a nonlinear quantum operator in the learning process.

The memory positions of a RAM node normally stores the values 0 or 1. In a qRAM network one will store only a single qubit. Instead of direct store a qubit, we use the A operator (described in Equation 9) and selectors to produce the desired qubit. This strategy avoid loss of information after measurements.

$$\mathsf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{9}$$

A. Definition

In Definition 1 a qRAM node is formally defined and in Fig. 3 the quantum circuit of an example of a qRAM node is shown. The output $|o\rangle$ of a qRAM node can be used as the input of another neuron forming layers as showed in Fig. 4.

Definition 1: A qRAM node with n inputs is represented by the operator N described in equation (10). The inputs, selectors and outputs of N are organized in three quantum registers $|i\rangle$ with n qubits, $|s\rangle$ with 2^n qubits and $|o\rangle$ with 1 qubit. The quantum state $|i\rangle$ describes qRAM input, and quantum state $|s\rangle|o\rangle$ describes qRAM state.

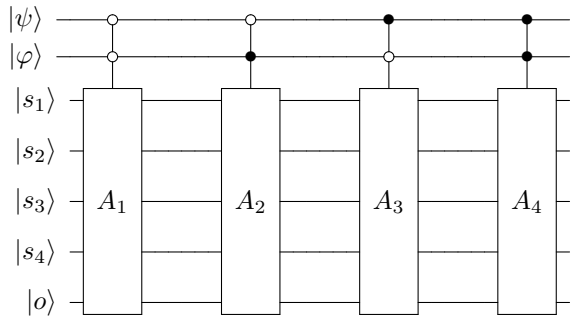

Fig. 3. qRAM node

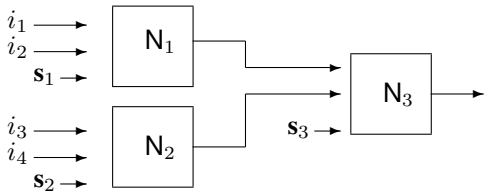

Fig. 4. Two layer qRAM network

$$\mathsf{N} = \sum_{i=0}^{2^n-1} |i\rangle_n\langle i|_n A_{s_i,o} \tag{10}$$

In next subsection we shown how a qRAM network works showing an example of execution. The qRAM neuron acts like a GSN neuron when a input $H|0\rangle$ is received.

B. Simulation

Let Net be a quantum network with architecture described in Fig. 4 and node circuits described in Fig. 3, Equation 11 describe the circuit of the nodes in the network. One can present inputs simultaneously as in Equation (12) to qRAM network. We will see that when a network receives inputs in superposition the outputs of each inputs are calculated simultaneously in the superposition.

$$\begin{aligned}\mathsf{N} = |00\rangle\langle 00| \otimes \mathsf{A}_{s_1,o} + |01\rangle\langle 01| \otimes \mathsf{A}_{s_2,o} + \\ |10\rangle\langle 10| \otimes \mathsf{A}_{s_3,o} + |11\rangle\langle 11| \otimes \mathsf{A}_{s_4,o}\end{aligned} \tag{11}$$

$$|i\rangle = \frac{1}{2}\left(|0000\rangle + |0001\rangle + |0010\rangle + |0011\rangle\right) \tag{12}$$

The neuron N_1 receives as input the two first qubits of state $|i\rangle$, $|i_1 i_2\rangle = |00\rangle$. The action of neurons of network is described by the operator in equation (11). The action of the neuron N_1 and N_2 are described in equation (13) and (14).

$$\begin{aligned}\mathsf{N}_1|i_1 i_2\rangle|\mathbf{s}_1\rangle|o_1\rangle = (|00\rangle\langle 00| \otimes A_{s_1,o} + |01\rangle\langle 01| \otimes A_{s_2,o} + \\ |10\rangle\langle 10| \otimes A_{s_3,o} + |11\rangle\langle 11| \otimes A_{s_4,o})\,|00\rangle|0110\rangle|0\rangle = \\ |00\rangle\langle 00| \otimes A_{s_1,o}\,(|00\rangle|0110\rangle|0\rangle) = |00\rangle|0110\rangle|0\rangle\end{aligned} \tag{13}$$

$$\begin{aligned}\mathsf{N}_2|i_3 i_4\rangle|\mathbf{s}_2\rangle|o_2\rangle = (|00\rangle\langle 00| \otimes A_{s_1,o} + |01\rangle\langle 01| \otimes A_{s_2,o} + \\ |10\rangle\langle 10| \otimes A_{s_3,o} + |11\rangle\langle 11| \otimes A_{s_4,o})\,\frac{1}{2}\,(|00\rangle + \\ |01\rangle + |10\rangle + |11\rangle)\,|00\rangle|0110\rangle|0\rangle = \\ \frac{1}{2}\,(|00\rangle\langle 00| \otimes A_{s_1,o}\,(|00\rangle|0110\rangle|0\rangle)) + \\ |01\rangle\langle 01| \otimes A_{s_2,o}\,(|00\rangle|0110\rangle|0\rangle)) + \\ |10\rangle\langle 10| \otimes A_{s_3,o}\,(|00\rangle|0110\rangle|0\rangle)) + \\ |11\rangle\langle 11| \otimes A_{s_4,o}\,(|00\rangle|0110\rangle|0\rangle))) = |00\rangle|0110\rangle|0\rangle + \\ |01\rangle|0110\rangle|1\rangle + |10\rangle|0110\rangle|1\rangle + |11\rangle|0110\rangle|0\rangle\end{aligned} \tag{14}$$

The outputs of neurons N_1 and N_2 are $|o_1\rangle = |0\rangle$ and $|o_2\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$. These outputs will be used as inputs of neuron N_3. Equation (15) shows the action of N_3 and the network calculates the outputs of all the inputs in superposition.

$$\begin{aligned}\mathsf{N}_3|o_1 o_2\rangle|\mathbf{s}_3\rangle|o_3\rangle = \frac{1}{\sqrt{2}}\,(|00\rangle\langle 00|A_{s_1 o_3}|00\rangle|0110\rangle|0\rangle + \\ |01\rangle\langle 01|A_{s_2 o_3}|01\rangle|0110\rangle|0\rangle) = \\ \frac{1}{\sqrt{2}}\,(|00\rangle|0110\rangle|0\rangle + |01\rangle|0110\rangle|1\rangle)\end{aligned} \tag{15}$$

The action of the qRAM network Net can be summarized as in equation (16). The networhk calculates the output of each input in superposition.

$$\begin{aligned}\mathsf{Net}\left(\frac{1}{2}\left(|0000\rangle + |0001\rangle + |0010\rangle + |0011\rangle\right)\right)|0\rangle = \\ \frac{1}{2}\,(\mathsf{Net}|0000\rangle|0\rangle + \mathsf{Net}|0001\rangle|0\rangle + \\ \mathsf{Net}|0010\rangle|0\rangle + \mathsf{Net}|0011\rangle|0\rangle) \\ = \frac{1}{2}\,(|0000\rangle|0\rangle + |0001\rangle|1\rangle + |0010\rangle|1\rangle + |0011\rangle|0\rangle)\end{aligned} \tag{16}$$

In the next section we show how one can train a qRAM network with only two network runs. To perform this task we explore the quantum properties of superposition and parallelism.

VII. Single Shot Quantum Learning Algorithm

Without quantum computation, the idea of training a neural network with a single execution is impracticable. In first SSLA steps, we will present all the patterns to the neural network simultaneously and compute the output for each input and each neural network configuration simultaneously.

To perform this idea classically one will need to create several copies of the neural network to receive all the inputs and compute in parallel the corresponding outputs. After to calculate the output of each pattern for each neural network configuration one can search the neural configuration with best performance. Yet classically the idea of SSLA learning is presented in Figure 5. For a given neural network architecture, all the patterns in the training set $P = \{p_1, p_2, \cdots, p_k\}$ are presented for each of the n neural network configurations,

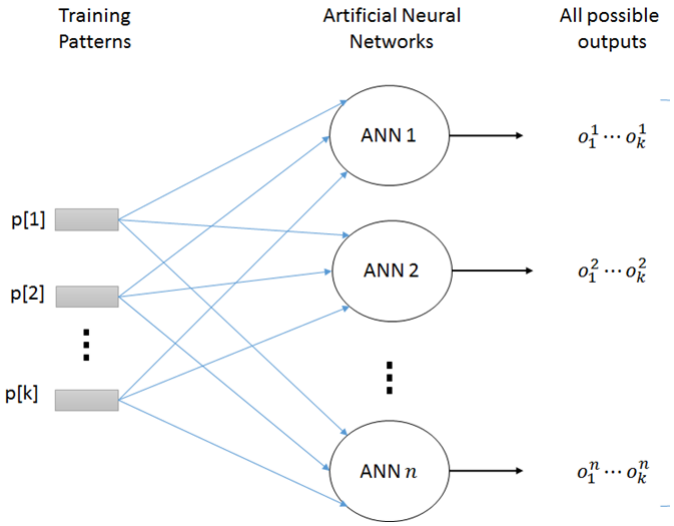

Fig. 5. Superposition based learning framework

possible in a parallel computer. The output are calculate and then one can search the best neural network parameters.

In a quantum computer, one can implement this improbable strategy without the hardware limitations showed in the classical strategy. Algorithm 1 presents the SSLA. It is a quantum-learning algorithm for quantum neural networks such that output o, parameters p and desired output d are represented in separated quantum registers. SSLA is based on the Superposition based Learning Algorithm (SLA) proposed in [1]. The main difference between SLA and SSLA is that in SSLA one needs only 1 run of the quantum neural network to perform the learning task.

Algorithm 3: Single Shot Quantum Learning Algorithm

1 Initialize all the qubits in register **s** with the quantum state $H|0\rangle$.
2 Initialize the register **p**, **o** , **d** with the quantum state $|p\rangle = \sum_{i=1}^{p} |p_i, 0, d_i\rangle$.
3 $|\psi\rangle = N|\psi\rangle$, where N is a quantum operator representing the action of the neuron.
4 Set registrator **e** to state $|e\rangle = |1\rangle$ where the registers **p,o,d** $= \sum_{i=1}^{p} |p_i, d_i, d_i\rangle$
5 Apply the phase inversion operation in state $|\psi\rangle$ where the register **e** $= |1\rangle$
6 Apply the inversion about the mean $-I + 2A$ in register **s**
7 Repeat the steps 6 and 7 $T = \frac{pi}{4}\sqrt{n}$ times, where n is the number of possible selectors to the network.
8 Measure the register **s** to obtain the desired parameters.

Algorithm 3 describes SSLA. In the first SSLA step the free parameters of the network are initialized in a superposition of all possible values. To perform this we initialize each neuron with the value $H|0\rangle$, this value can be interpreted as the u value in Algorithm 1. If the number of selectors is n, then the state of register s is described in Equation (17). This superposition of all this values means that we have, for a given architecture, all neural networks in superposition.

$$|s\rangle = \frac{1}{2^n} \sum_{i=0}^{2^n-1} |i\rangle \tag{17}$$

In step 2 we initialize the quantum registers p, o and d. The register o state is initialized in computation basis state . Registers p and d are initialized with an entangled superposition of patterns and desired output. This step can be performed with the algorithm proposed in [3]. Equation (18) describes state of quantum registers p, o and d after step 2.

$$|p, o, d\rangle = \frac{1}{\sqrt{k}} \sum_{i=1}^{k} |p_i, 0, d_i\rangle \tag{18}$$

At this moment $|\psi\rangle = |s, p, d, o\rangle$ is in state described in equation (3), where all the networks are in superposition and all the training patterns and desired outputs are prepared to be presented to the network.

$$|\psi\rangle = |s, p, o, d\rangle = \frac{1}{2^n\sqrt{k}} \sum_{i=0}^{2^n-1} \sum_{j=1}^{k} |i, p_j, 0, d_j\rangle \tag{19}$$

In step 3 we apply the neural network $\mathbf{N}$ to the quantum the state $|\psi\rangle$. After this step the outputs o_{ij} of each selector i and pattern j will be calculated. Equation (4) describes the resultant state.

$$|\psi\rangle = \frac{1}{2^n\sqrt{k}} \sum_{i=0}^{2^n-1} \sum_{j=1}^{k} |i, p_j, o_{ij}, d_j\rangle \tag{20}$$

In step 4 we use an auxiliary quantum register . After step 4 the state $|\psi\rangle|e\rangle$ is as in Equation (5), where $e_{ij} = 1$ if $o_{ij} = d_j$ for each j. This step requires the application of a nonlinear quantum algorithm proposed in [22].

$$|\psi\rangle|e\rangle = \frac{1}{2^n\sqrt{k}} \sum_{i=0}^{2^n-1} \sum_{j=1}^{k} |i, p_j, o_{ij}, d_j, e_{ij}\rangle \tag{21}$$

In steps 5, 6 and 7 we perform a quantum search looking for parameters such that $o_{ij} = d_j$, i.e. $e_{ij} = 1$, in the quantum register **s**. This step will increase amplitude probability of the part in superposition $|\psi\rangle$. In step 8 we measure the quantum register s to obtain the desired parameters. The algorithm will return the set of parameters that best fits the train set. SSLA search the space of free parameters of the network and return the parameters that make the network fits the train set.

VIII. Conclusion

In [13] has been shown that weightless neural networks has quantum versions that can be trained with the classical learning algorithms and allow quantum learning. The quantum weightless neural networks can be defined in a natural way, preserving theoretical as the capacity to simulate probabilistic

automata and Turing machines and practical results that show the capacity of weightless networks to learn and generalize.

In this paper we proposed a quantum Single-Shot Learning Algorithm inspired in the learning algorithm of GSN network and the superposition based learning algorithm. The main characteristic of the SSLA is that it requires only a single execution of the neural network. The main problem in SSLA is that it requires the use of nonlinear quantum algorithms. The same problem occurs in several learning algorithms for neural networks [13], [21], [23].

Compared with the GSN learning algorithm the SSLA algorithm has the advantage of consider the performance of the network for all patterns and in GSN a greed strategy is used, where the pattern presentation order will influence the final classifier. The single shot procedure used in SSLA is different of the strategy used in classical WNN. Here the networks runs only once with all data in superposition. SSLA does not divide the data set in train, validation and test sets, so one can investigate in how to use a validation set to guide the learning process.

One possible future work is to develop a single shot learning algorithm using only linear quantum operators. This may be done presenting each pattern to the neural network iteratively and storing the performance of each network in superposition as in [21]. In this way one will be capable of search the space of performances and setting the performance value may avoid overfitting.

Theoretical analyses of weightless neural networks have been done in some works. The main results are the capacity of simulate probabilistic automata and Turing machines with weightless neural networks. One can verify if this also can be performed in the quantum case and verify if quantum automata and quantum Turing machine can be simulated with quantum weightless neural networks.

Another possible future work is to develop a quantum weighted neural network with the properties of the quantum weightless neural networks: 1) direct implementation in quantum circuits; 2) capacity to simulate classical algorithms 3) with a structure that allows the utilization of quantum algorithms. One way to achieve a quantum perceptron with this characteristics is to associate the quantum neuron with a field, so if the field is enough large to approximate the rational numbers one may simulate the classical weighted neural networks.

ACKNOWLEDGMENT

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

REFERENCES

[1] I. Aleksander, "Self-adaptive universal logic circuits," *Electronics Letters*, vol. 2, no. 8, pp. 321–322, 1966.

[2] I. Aleksander, M. de Gregorio, F. França, P. Lima, and H. Morton, "A brief introduction to weightless neural systems," in *European Symposium on Artificial Neural Networks, 2009. Proceedings.*, April 2009, pp. 299–305.

[3] S. Yong, W. Lai, and G. Goghill, "Weightless neural networks for typing biometrics authentication," in *Knowledge-Based Intelligent Information and Engineering Systems*, ser. Lecture Notes in Computer Science, 2004, vol. 3214, pp. 284–293.

[4] A. F. D. Souza, F. Pedroni, E. Oliveira, P. M. Ciarelli, W. F. Henrique, L. Veronese, and C. Badue, "Automated multi-label text categorization with vg-ram weightless neural networks," *Neurocomputing*, vol. 72, no. 1012, pp. 2209 – 2217, 2009.

[5] B. P. Grieco, P. M. Lima, M. D. Gregorio, and F. M. Frana, "Producing pattern examples from mental images," *Neurocomputing*, vol. 73, no. 79, pp. 1057 – 1064, 2010.

[6] D. Cardoso, M. de Gregorio, P. Lima, J. Gama, and F. Frana, "A weightless neural network-based approach for stream data clustering," in *Intelligent Data Engineering and Automated Learning - IDEAL 2012*, ser. Lecture Notes in Computer Science, 2012, vol. 7435, pp. 328–335.

[7] M. de Souto, T. Ludermir, and W. de Oliveira, "Equivalence between ram-based neural networks and probabilistic automata," *Neural Networks, IEEE Transactions on*, vol. 16, no. 4, pp. 996–999, july 2005.

[8] R. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, pp. 467–488, 1982.

[9] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[10] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, ser. STOC '96. ACM, 1996, pp. 212–219.

[11] W. de Oliveira, A. J. Silva, T. B. Ludermir, A. Leonel, W. Galindo, and J. Pereira, "Quantum logical neural networks," *Neural Networks, 2008. SBRN '08. 10th Brazilian Symposium on*, pp. 147–152, oct. 2008.

[12] W. de Oliveira, "Quantum ram based neural networks," in *European Symposium on Artificial Neural Networks, 2009. Proceedings.*, April 2009, pp. 331–336.

[13] A. J. da Silva, W. R. de Oliveira, and T. B. Ludermir, "Classical and superposed learning for quantum weightless neural networks," *Neurocomputing*, vol. 75, no. 1, pp. 52–60, 2012.

[14] E. Filho, M. Fairhurst, and D. Bisset, "Adaptive pattern recognition using goal seeking neurons," *Pattern Recognition Letters*, vol. 12, no. 3, pp. 131 – 138, 1991.

[15] T. B. Ludermir, A. Carvalho, A. P. Braga, and M. C. P. Souto, "Weightless neural models: A review of current and past works," *Neural Computing Surveys*, vol. 2, pp. 41–61, 1999.

[16] A. F. Souza, F. Pedroni, E. Oliveira, P. M. Ciarelli, W. F. Henrique, L. Veronese, and C. Badue, "Automated multi-label text categorization with VG-RAM weightless neural networks," *Neurocomputing*, vol. 72, no. 10-12, pp. 2209–2217, 2009.

[17] W. R. de Oliveira, M. C. P. de Souto, and T. B. Ludermir, "Turing's analysis of computation and artificial neural networks," *J. Intell. Fuzzy Syst.*, vol. 13, no. 2-4, pp. 85–98, 2002.

[18] N. S. Yanofsky and M. A. Mannucci, *Quantum Computing for Computer Scientists*. New York, NY, USA: Cambridge University Press, 2008.

[19] M. V. Altaisky, "Quantum neural network," Joint Institute for Nuclear Research, Russia, Technical report, 2001, available online at the Quantum Physics repository:arXiv:quant-ph/0107012v2.

[20] R. Zhou and Q. Ding, "Quantum m-p neural network," *Int J Theor Phys*, vol. 46, pp. 3209–3215, 2007.

[21] M. Panella and G. Martinelli, "Neural networks with quantum architecture and quantum learning," *International Journal of Circuit Theory and Applications*, vol. 39, pp. 61–77, july 2011.

[22] D. S. Abrams and S. Lloyd, "Nonlinear quantum mechanics implies polynomial-time solution for np-complete and $\#p$ problems," *Phys. Rev. Lett.*, vol. 81, no. 18, pp. 3992–3995, Nov 1998.

[23] R. Zhou, H. Wang, Q. Wu, and Y. Shi, "Quantum associative neural network with nonlinear search algorithm," *International Journal of Theoretical Physics*, vol. 51, no. 3, pp. 705–723, 2012.