UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CLÁUDIO LUIS ALVES MONTEIRO

QUANTUM NEURONS WITH REAL WEIGHTS FOR DIABETES PREDICTION

Recife
2021

CLÁUDIO LUIS ALVES MONTEIRO

QUANTUM NEURONS WITH REAL WEIGHTS FOR DIABETES PREDICTION

A M.Sc. Dissertation presented to the Centro de Informática of Universidade Federal de Pernambuco in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Concentration Area: Computational Intelligence
Advisor: Fernando Maciano de Paula Neto

Recife
2021

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

M775q Monteiro, Cláudio Luis Alves
Quantum neurons with real weights for diabetes prediction / Cláudio Luis Alves Monteiro. – 2021.
51 f.: il., fig., tab.

Orientador: Fernando Maciano de Paula Neto.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2021.
Inclui referências.

1. Inteligência computacional. 2. Aprendizado de máquina. I. Paula Neto, Fernando Maciano de (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2021 – 189

CLÁUDIO LUIS ALVES MONTEIRO

“QUANTUM NEURONS WITH REAL WEIGHTS FOR DIABETES PREDICTION”

A M.Sc. Dissertation presented to the Centro de Informática of Universidade Federal de Pernambuco in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Approved in: 02/09/2021.

Advisor: Fernando Maciano de Paula Neto

EXAMINING COMISSION

I dedicate this work to my mother and to the scientists that try to make the world a better place through knowledge.

ACKNOWLEDGEMENTS

I thank to all beings that passed through my life and directly or indirectly contributed for this work. I highlight the contribution of my advisor professor Fernando who helped me through all the processes of this degree and my mother who always believed in my capacity and supported my education among all adversities.

I'm also grateful to friends and colleagues that make life lighter and to the professors and other workers of Centro de Informática who make the best environment possible for us to learn and develop our work. Finally, this work was supported by the CAPES scholarship and I thank the ones who made it possible for me to receive this fund.

ABSTRACT

Parametric models with real numbers valued parameters have greater performance than its counterparts with binary valued weights, due to the gain in representing information with real values, and therefore having a larger space for memory association. In this work, is proposed a quantum neuron capable of store real weights and preserve the gain of the superposition property, encoding the information in the probability amplitudes of the quantum system, the Real Weights Quantum Neuron. Its performance is compared with other quantum neurons to analyze the application of the quantum neurons on real-world problems, i.e diabetes classification. The results of the experiments shows that a single quantum neuron is capable of achieving an accuracy rate of 100% in the XOR problem and an accuracy rate of 100% in a non-linear dataset, demonstrating that the quantum neurons with real weights are capable of modeling non-linearly separable problems. In the problem of diagnosing diabetes, quantum neurons achieved an accuracy rate of 76% and AUC-ROC of 88%, while its classic version, the perceptron, reached only 63% accuracy and the artificial neural network reached 80% AUC-ROC. These results indicate that a single quantum neuron performs better than its classical version and even the artificial neural network for AUC-ROC, demonstrating potential for use in healthcare applications in the near future. This work is also a contribution to the field of quantum neural networks, which can be further advanced from the quantum neuron proposed.

Keywords: machine learning; diabetes mellitus; healthcare; quantum computing; quantum machine learning.

RESUMO

Este trabalho apresenta resultados sobre a aplicação de algoritmos de aprendizagem de máquina quântica no setor de saúde. Foi desenvolvida e testada uma proposta de neurônio quântico capaz de armazenar pesos reais em comparação com outros neurônios quânticos. Esse modelos podem transportar uma quantidade exponencial de informação para um número linear de unidades de informação quântica (qubits) usando a propriedade quântica de superposição. Foi comparado o desempenho desses algoritmos nos seguintes problemas: simular o operador XOR, resolver um problema não linear genérico e previsão de diabetes em pacientes. Os resultados dos experimentos mostraram que um único neurônio quântico é capaz de atingir uma acurácia de 100% no problema XOR e 100% de acurácia em um conjunto de dados não linear, demonstrando que neurônios quânticos com pesos reais são capazes de classificar corretamente problemas não linearmente separáveis. No problema de classificação de diabetes, os neurônios quânticos alcançaram uma acurácia de 76% e AUC-ROC de 88%, enquanto sua versão clássica, o perceptron, atingiu apenas 63% de acurácia e a rede neural artifical atingiu 80% AUC-ROC. Esses resultados indicam que um único neurônio quântico tem um desempenho maior que sua versão clássica e até mesmo que a rede neural artifical na AUC-ROC, demonstrando seu potencial para uso em aplicações para o setor de saúde. Este trabalho é também uma contribuição ao campo das redes neurais quânticas, que pode ser avançada a partir do neurônio quântico proposto.

Palavras-chaves: aprendizado de máquina; diabetes mellitus; saúde; computação quântica; aprendizado de máquina quântico.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

The growing development of information systems for the management of healthcare institutions enabled the storage of patient data such as medical diagnosis, personal characteristics, events and procedures during hospitalization and hospital infrastructure. Although this data has been historically stored in hard copies, the use of Electronic Health Records leads to an expanding digital storage, such that the U.S. healthcare system has reached 150 exabytes in 2011 (RAGHUPATHI; RAGHUPATHI, 2014). All this information can be used to develop machine learning applications to enhance the care services, e.g. predict hospitalization and intervening before the patient reaches a critical point, predict the risk of developing a wide range of diseases and promote the preventive care.

Recent advances in computing hardware made it possible to achieve better results in machine learning and artificial intelligence. Although there are models capable of storing large and complex patterns such as Artificial Neural Networks (ANNs), much still has to be done to develop systems with higher performance for speech and image recognition, industrial optimisation, financial forecast and other areas. As an example, the OpenAI company launched in 2020 the state of the art GPT-3 language model (BROWN et al., 2020), with 174 billion parameters and trained by more than 440 billion text-encoded tokens. There is a growing need for models capable of high-dimensional representations of knowledge and can be efficiently trained with large volumes of data to develop intelligent systems capable of doing more complex activities.

Quantum Computing may be used in the near future to address this need due to its quantum mechanical properties of storing large complex-valued vectors and matrices and performing linear operations on these vectors. 2^N classical N-bits of information can be stored in a N-qubits quantum state (e.g. a 40-qubit system stores more than 1 trillion quantum states). Furthermore, quantum states amplitudes can represent the weights of a neural model, and the inner product between input data and weights can be performed.

Neural models such as Rosenblatt's perceptron (ROSENBLATT, 1958) were proposed and investigated at the earlier stages of machine learning (artificial intelligence). Perceptrons works with binary weights, such that given m inputs $x_k \in \{-1, 1\}$, $k = 0, ..., m-1$ and m weights, $w_k \in [-1, 1)$, the output of the perceptron is the inner product of the inputs and weights. Tacchino *et al.* (TACCHINO et al., 2019) develops a quantum neuron based on the perceptron, exploiting the superposition to encode binary input and weights in quantum states. Recently, a quantum artificial neural network (QANN) has been proposed using the perceptron with binary input and weight (TACCHINO et al., 2020). However, in the task that ANNs are involved, the weight stores the network information. Siegelmann *et al.* show that the type of weights of classical neural networks is intrinsically related to their computational power (SIEGELMANN; SONTAG, 1991; SIEGELMANN, 1995). For example, neural networks with rational weights are computationally equivalent to Turing Machines whilst arbitrary real-valued weights have super-Turing computing power.

One can use existing quantum algorithms that solve optimisation problems (ORUS; MUGEL; LIZASO, 2019; JONES; MOSCA; HANSEN, 1998; SHENVI; KEMPE; WHALEY, 2003; HAN; KIM, 2000; BOYER et al., 1998) at an acceptable cost and to solve weight search problems in ANN or QANN. Altaysky's QANN (ALTAISKY, 2001) acts over a single input qubit, returning as output a state of a qubit, described by a learning rule that adapts the weight matrix to the expected result. The quantum m-p neural network proposed by Zhou (ZHOU; DING, 2007) uses a neuron representation given by a weight matrix that is updated from a training set. Although closer to the functioning of classical neural networks, it does not have a nonlinear activation function (SILVA; OLIVEIRA; LUDERMIR, 2015). Wiebe (WIEBE; KAPOOR; SVORE, 2016) exploited the Grover Search Algorithm (GROVER, 1996) to propose quantum perceptron training algorithms that are significantly faster than its classical counterpart, while also making training less sensitive to small errors. Kristensen *et al* (KRISTENSEN et al., 2020) develops an artificial spiking quantum neuron using the spins of qubits to generate a model capable of returning the similarity between two bell states.

The quantum learning model developed by Panella (PANELLA; MARTINELLI, 2011) uses the superposition of states to perform a search for the optimal network through nonlinear operators. The qRAM from de Oliveira (OLIVEIRA, 2009) operates with 2^n selectors where n is the number of qubits as inputs, one qubit as output and 2^n matrices of controlled NOT gates, which allows one to learn classically but not uses a nonlinear function as an activation function. Lamata (LAMATA, 2020) reviews the field of quantum biomimetics as a possible approach to solve optimisation problems, building quantum computing algorithms that mimic the behaviour of biological systems. Approaches to performing nonlinear operations on quantum neurons have been proposed (YAN; QI; CUI, 2020; PANELLA; MARTINELLI, 2011; NETO et al., 2019; SCHULD; SINAYSKIY; PETRUCCIONE, 2014), demonstrating the capacity of solving nonlinear separable problems with quantum computing. Nevertheless, studies of the applications of these algorithms in real-world problems (BIAMONTE et al., 2017) are lacking.

Chapter 2 introduces quantum notation and the fundamentals of quantum operations necessary to understand this work. In Chapter 3, is proposed a new quantum neuron with real-valued weights, combining the hypergraph states generation subroutine (TACCHINO et al., 2019) for data input and the uniformly controlled rotations proposed in (MöTTöNEN et al., 2005) for encoding the weights. The proposed model substantially saves the use of auxiliary quantum registers, since it encodes information in the amplitude of quantum states, a technique known as *information encoding.* Experiments indicate that, when coupling bias in the neuron to address the symmetry problem, the quantum neuron with real weights has higher predictive performance than the binary weights quantum neuron. Experiments on noisy intermediate-scale quantum (NISQ) processors (PRESKILL, 2018) were performed and the results indicate that the proposed quantum neuron has the potential to be executed in real-world applications. Therefore, contributing to the development of machine learning algorithms that exploit the properties of quantum computing.

To analyse the suitability of quantum neurons for healthcare problems, In Chapter 4 the performance of the Real Weights Quantum Neuron proposed in chapter 3, the Binary Quantum Neuron proposed by (TACCHINO et al., 2019) and the Continuously Valued Quantum Neuron proposed by (MANGINI et al., 2020) are compared, models which are capable to carry an exponential amount of information to a linear number of quantum information units (qubits) using the quantum property of superposition, and there is also a proposal of preprocessing strategies based on the notion of hypercomplex numbers. These algorithms are tested on the problems of simulating the XOR operator, solving a generic nonlinear problem and a novel test on a healthcare problem, the prediction of patients with diabetes, whereas in the original and related works this was not explored (MANGINI et al., 2021; TACCHINO et al., 2020). Experiments indicate that the quantum neurons have the potential to be executed in real-world applications in the healthcare industry and that it can be further advanced in order to improve the performance in classification tasks.

2 QUANTUM COMPUTING

In this chapter, quantum computing basic concepts are reviewed to serve as a basis for the understanding of quantum neurons. Further material can be found in (NIELSEN; CHUANG, 2000). Quantum computing uses the principles of quantum mechanics to perform information processing. There are properties underlying quantum behaviour not found directly in the classical macroscopic world, such as entanglement and superposition, which are believed to be accelerators of information processing (JOZSA; LINDEN, 2003).

In quantum computing, the minimum unit of information is a *qubit*. A qubit represents a system that can be in two states simultaneously. Mathematically, the qubit can be represented as a complex two-dimensional vector in $\mathbb{C}^2$. Using the computational basis formed by the vectors $|0\rangle = [1, 0]^T$ and $|1\rangle = [0, 1]^T$, is possible to generate any vector in this space by linear combination (or superposition), such as shown in Equation 2.1. The Dirac notation represents quantum vectors with the symbol $|\cdot\rangle$.

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \tag{2.1}$$

The values α and β are complex numbers, also called *probability amplitudes*, and obey the normalisation condition $|\alpha|^2 + |\beta|^2 = 1$. These values also indicate that there is a chance to measure or observe the quantum states $|0\rangle$ and $|1\rangle$ with the probability $|\alpha|^2$ and $|\beta|^2$, respectively. Quantum computing then becomes the application of quantum operators that change the probability amplitudes of the qubits involved, represented as a complex vector space. A complex vector is a nonempty set $\mathbb{V}$ whose elements are called vectors, with the operations addition, negation and scalar multiplication, which obey to the following properties: for all V, W, $X \in \mathbb{V}$, and for a, b, c, $\in \mathbb{C}$, (i) commutativity of addition denotes that $V + W = W + V$; (ii) associativity of addition denotes that $(V + W) + X = V + (W + X)$; (iii) zero is an additive identity, $V + 0 = V = 0 + V$; (iv) every vector has an inverse, $V + (-V) = 0 = (-V) + V$; (v) scalar multiplication has a unit, $1 \cdot V = V$; (vi) scalar multiplication respects complex multiplication, $a \cdot (b \cdot V) = (a \times b) \cdot V$; (vii) scalar multiplication distributes over addition, $c \cdot (V + W) = c \cdot V + c \cdot W$; and finally, (viii) scalar multiplication distributes over complex addition, $(a + b) \cdot V = a \cdot V + b \cdot V$ (YANOFSKY; MANNUCCI, 2008).

It is possible to compose many qubits in a single quantum system through the operation of a *tensor product* $\otimes$. The tensor product is used to represent quantum systems with two or more qubits $|ij\rangle = |i\rangle \otimes |j\rangle$. Let S and T be two vector spaces, the tensor product of S and T, denoted by $S \otimes T$, is the vector space generated by the tensor product of all vectors $|s\rangle \otimes |t\rangle$, with $|s\rangle \in S$ and $|t\rangle \in T$. For example, the operation $|\psi_1\rangle \otimes |\psi_2\rangle$ from two states $|\psi_1\rangle = \alpha_1 |0\rangle + \alpha_2 |1\rangle$ and $|\psi_2\rangle = \beta_1 |0\rangle + \beta_2 |1\rangle$ is defined as $|\psi_1\rangle \otimes |\psi_2\rangle = |\psi_1\psi_2\rangle = (\alpha_1 |0\rangle + \alpha_2 |1\rangle) \otimes (\beta_1 |0\rangle + \beta_2 |1\rangle) = \alpha_1\beta_1 |00\rangle + \alpha_1\beta_2 |01\rangle + \alpha_2\beta_1 |10\rangle + \alpha_1\beta_2 |11\rangle$.

Some states $|\psi\rangle \in S \otimes T$ cannot be written as a product of states of its component systems S and T. States with this property are called *entangled* states. For instance, two entangled qubits are the Bell states described in Equation (2.2).

$$\begin{aligned} \left|\Phi^+\right\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\ \left|\Phi^-\right\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}} \\ \left|\Psi^+\right\rangle &= \frac{|01\rangle + |10\rangle}{\sqrt{2}} \\ \left|\Psi^-\right\rangle &= \frac{|01\rangle - |10\rangle}{\sqrt{2}} \end{aligned} \tag{2.2}$$

To transform the information stored on qubits, we use *quantum operators*. In this work, they essentially are *unitary matrices* or *gates* that can act on single or multiple qubits. Quantum operator $\mathbf{U}$ over n qubits is a unitary complex matrix of order $2^n \times 2^n$. We specify the gates: Identity $\mathbf{I}$, which generates the output identical to the input; the Pauli $\mathbf{X}$ operator, which works like the classic NOT on the computation basis, flipping the value of the qubit; the Hadamard $\mathbf{H}$ gate which produces a superposition of quantum states; and the Pauli $\mathbf{Z}$ gate, that applies a negative in the quantum state when the qubit is not zero.

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{matrix} \mathbf{I}\,|0\rangle = |0\rangle \\ \mathbf{I}\,|1\rangle = |1\rangle \end{matrix} \qquad \mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \begin{matrix} \mathbf{X}\,|0\rangle = |1\rangle \\ \mathbf{X}\,|1\rangle = |0\rangle \end{matrix}$$

$$\mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad \begin{matrix} \mathbf{H}\,|0\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle) \\ \mathbf{H}\,|1\rangle = 1/\sqrt{2}(|0\rangle - |1\rangle) \end{matrix}$$

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \begin{matrix} \mathbf{Z}\,|0\rangle = |0\rangle \\ \mathbf{Z}\,|1\rangle = -\,|1\rangle \end{matrix}$$

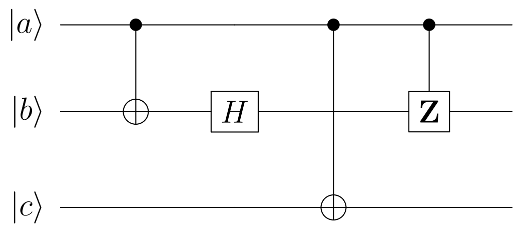
Furthermore, there are the controlled gates that perform transformations on multiple qubits, acting with conditional structures (*i.e.* if / else). The Controlled NOT gate **CNOT** operates on two qubits, controlled by the first and applying the **NOT** gate in the second one. The matrix representation of this operator is defined below:

$$\mathbf{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \begin{matrix} \mathbf{CNOT}\,|00\rangle = |00\rangle \\ \mathbf{CNOT}\,|01\rangle = |01\rangle \\ \mathbf{CNOT}\,|10\rangle = |11\rangle \\ \mathbf{CNOT}\,|11\rangle = |10\rangle \end{matrix}$$

In the **CNOT** operator, the **X** operator is conditionally applied, depending on the control qubits. However, it is possible to define a controlled operation for any quantum operator, Controlled-U gate, or just **C-U**, defined as $\mathbf{C\text{-}U}\,|00\rangle = |00\rangle$, $\mathbf{C\text{-}U}\,|01\rangle = |01\rangle$, $\mathbf{C\text{-}U}\,|10\rangle = |1\rangle\, U\,|0\rangle$ and finally $\mathbf{C\text{-}U}\,|11\rangle = |1\rangle\, U\,|1\rangle$.

It is possible to represent quantum operations in a circuit representation. In this representation, qubits are considered wires and operators as boxes or a few special symbols. Figure 1 has an example of a quantum circuit composed of a **CNOT**, where the control qubit is depicted by a filled circle, and the **X** operator represented as an encircled plus signal $\oplus$. The order of execution of the operators is the usual order of execution, from left to right.

Figure 1 – Example of quantum circuit with two **CNOT** operators, a **H** operator and a Controlled-Z operator

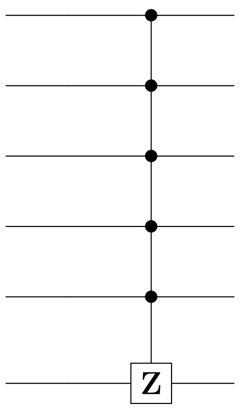


Source: The author (2021)

The **CNOT** and **C-U** operators can have more than one control qubit. These are qubits that are involved in the operation and all are conditioning factors for the application of an operator. $\mathbf{C\text{-}U}^N$ to generalise to N-ary **C-U** having $N-1$ control qubits and the last qubit being the target. An example of $\mathbf{C\text{-}Z}^6$ gate is given in Figure 2.

Figure 2 – Example of a **C-Z** operator with N=6

.



Source: The author (2021)

Quantum computing systems must stay isolated from the environment. Available quantum devices are still subjected to environmental interference that generates decoherence, a loss of the ordering of probability amplitudes due to variations in the phases of the quantum states. This interference also generates dissipation, the destruction of the population of quantum states when parts of the quantum system behave as classical systems

(SCHULD; SINAYSKIY; PETRUCCIONE, 2014). This implies that the expected output from quantum measurements is subject to higher variations over the execution time. Nevertheless, alternative approaches for quantum computing that incorporate the interaction of quantum systems with the environment have been proposed and can be further explored for the development of quantum neural models, e.g. dissipative quantum computing (VERSTRAETE; WOLF; CIRAC, 2009), measurement-based quantum computing (BRIEGEL et al., 2009) and adiabatic quantum computing (FARHI et al., 2000). E.g. Torrontegui and García-Ripoll (TORRONTEGUI; GARCíA-RIPOLL, 2019) proposed a quantum perceptron with the implementation of the perceptron operator behaving as a quasiadiabatic passage on an Ising-type spin model, which scales favourably with the network size and the total circuit error.

A possible approach for minimising the effects of decoherence and dissipation in larger space vectors, is to choose a modelling strategy that has fewer qubits and fewer iterations over time. This can be achieved by using an ensemble of quantum neurons for predicting different perspectives of the features, where each neuron is responsible for predicting one perspective and combining the decisions classically, or training one quantum neuron for each input (CAO; GUERRESCHI; ASPURU-GUZIK, 2017). Therefore the qubits required to execute the quantum neuron is reduced as each quantum neuron trained for subsets of data is smaller than the quantum neuron trained with full data features.

3 QUANTUM NEURON WITH REAL WEIGHTS

This chapter proposes a new quantum neuron, with real weights, exploiting the so-called quantum parallelism which allows for an exponential speedup of computations. The quantum neurons were trained in a classical-quantum approach, considering the delta rule to update the values of the weights in an image database of three distinct patterns. We performed classical simulations and also executed experiments in an actual small-scale quantum processor. The results of the experiments show that the proposed quantum neuron, with real weights has a good generalisation capacity, demonstrating better accuracy than the traditional binary quantum perceptron model. The paper proposing this new quantum neuron model was accepted and published in August 2021 in the journal Neural Networks (MONTEIRO et al., 2021).

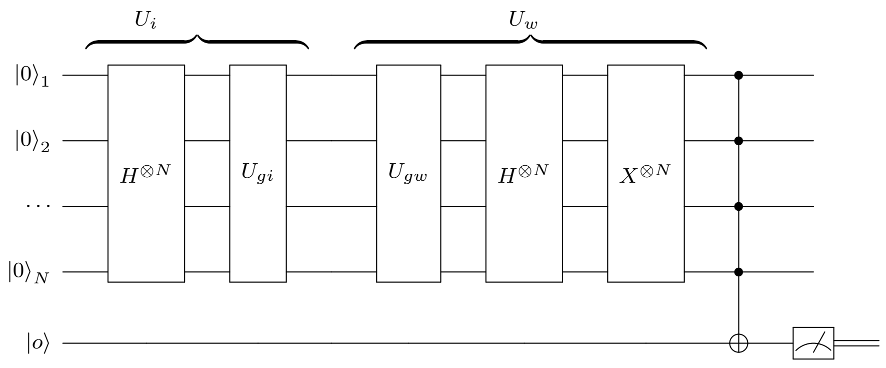
3.1 QUANTUM NEURONS

Schuld *et al* (SCHULD; SINAYSKIY; PETRUCCIONE, 2014) reviewed different strategies of quantum neural networks, stating that for a QNN to be considered valid it should satisfy the requirements: (1) The initial state of the quantum system encodes any binary string of length N; (2) the model reflects neural computing behaviors (e.g. synaptic connections, integrate and fire, training rule, etc.); and (3) the evolution is based on quantum effects such as superposition, entanglement and interference. The authors identified that most of the proposed models do not incorporate these requirements and therefore they do not fully exploit the advantages of quantum computing for neural computing applications. Nevertheless, they state that quantum perceptrons and quantum measurements proposals are viable modelling solutions but still must be further developed into mature Quantum Neural Networks (SCHULD; SINAYSKIY; PETRUCCIONE, 2015).

Tacchino *et al.* (TACCHINO et al., 2019) proposed a quantum neuron based on Rosenblatt's perceptron, exploring the property of superposition between qubits to generate 2^N possible entries for N qubits. The information is encoded in the amplitudes of quantum states in the system, employed by the Brute Force algorithm or the Hypergraph States Generation Subroutine (HSGS), detailed in Algorithm 21.

This quantum neuron calculates the inner product of two vectors $\vec{i}$ and $\vec{w}$ both of size $m = 2^N$. This computation is performed firstly using quantum operators that carry the information of these vectors in the amplitudes of quantum states $|\psi_i\rangle$ e $|\psi_w\rangle$. In this

Figure 3 – Circuit of quantum neuron for N input qubits proposed in (TACCHINO et al., 2019)



Source: Tacchino *et al* (2019)

model, the possible values for information input and weights are binary (e.g. $\{-1, 1\}$).

$$\vec{i} = \begin{pmatrix} i_0 \\ i_1 \\ \vdots \\ i_{m-1} \end{pmatrix}, \vec{w} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{m-1} \end{pmatrix} \text{Source: The author (2021)} \tag{3.1}$$

$$|\psi_i\rangle = \frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} i_j |j\rangle$$

$$|\psi_w\rangle = \frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} w_j |j\rangle$$

To generate the quantum state $|\psi_i\rangle$, we use a quantum operator U_i, i.e,

$$|\psi_i\rangle = U_i |0\rangle^{\otimes N} = U_{gi} H^{\otimes N} |0\rangle^{\otimes N} \tag{3.2}$$

To generate the state $|\psi_w\rangle$, an operator U_{gw}:

$$|\psi_w\rangle = U_{gw} H^{\otimes N} |0\rangle^{\otimes N} \tag{3.3}$$

After that, the circuit of the quantum neuron responsible for computing the inner product is made with the operators U_i and U_w, where $U_i = U_{gi} H^{\otimes N}$ and $U_w = X^{\otimes N} H^{\otimes N} U_{gw}$, followed by a Controlled-NOT (**CNOT**) operator with N control qubits and an auxiliary qubit as the target. The quantum Perceptron circuit is described in Figure 3.

The quantum neuron calculates the inner product $\langle\psi_w| |\psi_i\rangle$, considering that,

$$U_w |\psi_w\rangle = |1\rangle^{\otimes N} = |m-1\rangle \tag{3.4}$$

and

$$U_w |\psi_i\rangle = \sum_{j=0}^{m-1} c_j |j\rangle = |\phi_{i,w}\rangle \tag{3.5}$$

The operation $U_w |\psi_i\rangle$ returns the inner product output, which is coded at the last amplitude of the quantum state involved in the operation:

$$\langle\psi_w| |\psi_i\rangle = \langle\psi_w| U_w^\dagger U_w |_p si_i\rangle = \langle m-1| |\phi_{i,w}\rangle = c_{m-1} \tag{3.6}$$

Although, to consider that

$$U_w |\psi_w\rangle = |1\rangle^{\otimes N} = |m-1\rangle \tag{3.7}$$

other assumptions are needed about the operator U_{gw} that were not made explicit in (TACCHINO et al., 2019). For example, if $U_{gw} H^{\otimes N} |0\rangle^{\otimes N} = |\psi_w\rangle$, $U_w = X^{\otimes N} H^{\otimes N} U_{gw}$, and $U_w |\psi_w\rangle = |1\rangle^{\otimes N} = |m-1\rangle$ are true, then:

$$U_w |\psi_w\rangle = (X^{\otimes N} H^{\otimes N} U_{gw})(U_{gw} H^{\otimes N} |0\rangle^{\otimes N}) \tag{3.8}$$

It is possible to reduce the equation above **only** if $U_{gw} U_{gw} = I$, where $U_{gw} = U_{gw}^\dagger$ and I is the identity matrix:

$$\begin{aligned} U_w |\psi_w\rangle &= (X^{\otimes N} H^{\otimes N} U_{gw})(U_{gw} H^{\otimes N} |0\rangle^{\otimes N}) \\ &= X^{\otimes N} H^{\otimes N} H^{\otimes N} |0\rangle^{\otimes N} = X^{\otimes N} |0\rangle^{\otimes N} = |m-1\rangle \end{aligned} \tag{3.9}$$

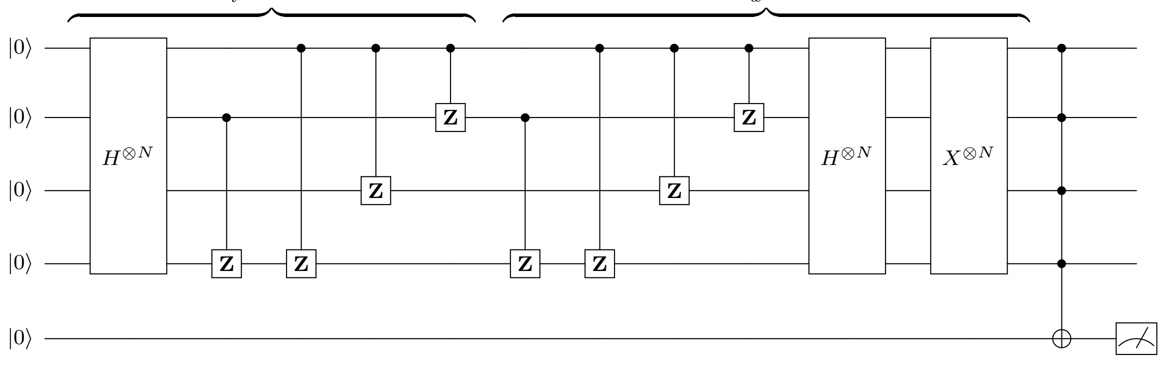
The HSGS algorithm transforms the probability amplitudes of the quantum states such that they represent the binary information of input and weights as 1 and -1 values. The first step is to check whether the value on the first position of the vector is -1. If the condition is satisfied, flip the sign of every element on the vector $\mathbf{v}$ (line 4). The next step is to verify if there is any quantum state with only one qubit in the state $|1\rangle$ (e.g. $|0100\rangle$) that has amplitude -1, if so, a **Z** gate is applied to that position. Then, for $p \in [2, \cdots, N]$., the elements of the quantum system in the computational basis with p qubits in state $|1\rangle$ are identified. For each of these cases, check whether it is necessary to insert a **Z** gate.

In Figure 4 it is possible to observe an example of the binary weights quantum neuron implemented using the HSGS algorithm to input data and weights in the quantum circuit.

Algoritmo 1: HSGS Algorithm proposed in (TACCHINO et al., 2019).

```
   Result: Quantum circuit QC;
1  Input: a given $\mathbf{v} = \{-1, 1\}^{2^N}$ input vector;
2  Auxiliary variables: vAux = $\{1\}^{2^N}$ vector;
3  if v[0] is -1 then
4      v = [amp.(-1) for amp in v] // Flip all the binary values of v;
5  end
6  for $i = 0; i < len(v); i++$ do
7      if v[i] is -1 and there exist only one qubit is in IntToBin(i) string then
8          Put Z gate on the circuit QC in the position of the one qubit of the state
            IntToBin(i);
9          Update vAux vector, considering that a signal change was applied in the
            position $i$;
10     end
11 end
12 for $p = 2, p < N, p++$ do
13     for $i = 0; i < len(\boldsymbol{v}); i++$ do
14         if IntToBin(i) has p bits equal to one then
15             if vAux[i] is not equal to v[i] then
16                 Put $C^pZ$ gate on the circuit QC between the $p$ qubits that have
                    values 1 in IntToBin(i);
17                 Update vAux vector considering that a signal changed was applied
                    in $j$ positions of vAux which $p$ qubits that have 1 value in
                    IntToBin(i) are in the same position in IntToBin(j);
18             end
19         end
20     end
21 end
```

Figure 4 – Example of quantum circuit for the binary weights quantum neuron



Input vector $v_i = [1, 1, 1, 1, 1, -1, 1, 1, 1, -1, -1, 1, -1, 1, 1, -1]$ and weights $w_i = [-1, -1, -1, -1, -1, 1, -1, -1, -1, 1, 1, -1, 1, -1, -1, 1]$.

Source: The author (2021)

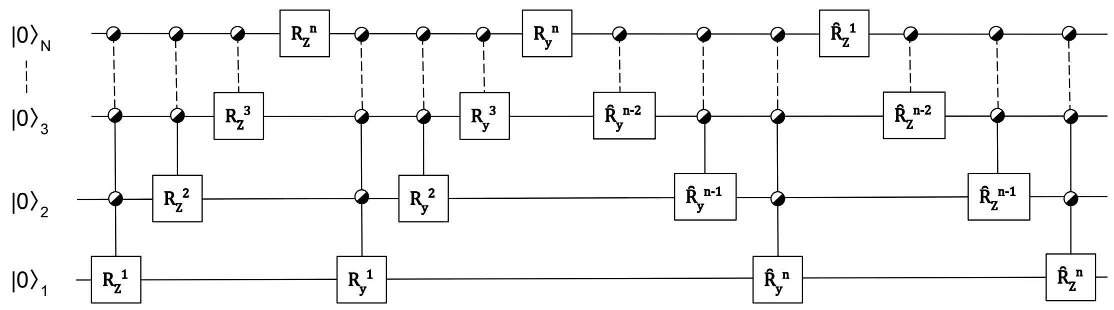
3.2 QUANTUM NEURON WITH REAL WEIGHTS

Parametric models with real numbers valued parameters have greater performance than its counterparts with binary valued weights. This is due to the gain in representing information with real values, and therefore having a larger space for memory association. To develop a quantum neuron capable of store real weights and preserve the gain with the superposition property, we need to encode the information in the probability amplitudes of the quantum system.

For the *HSGS* algorithm to generate U_{gi} and U_{gw} used in (TACCHINO et al., 2019), the condition $U_{gw}U_{gw} = I$ is true because is formed only by **Z** gates and controlled-**Z** gates. The restriction mentioned in 3.9 is applied only for the operator U_{gw}. The operator that generates the quantum state $|\psi_i\rangle$ does not have such restrictions. Therefore, we can use any other operator to load information in the quantum system.

Möttönen *et al.* (MöTTöNEN et al., 2005) proposed an algorithm to store real values in the amplitudes of quantum states. Here we named this circuit as the **Encoding Operator (EO)**, detailed in Figure 5. This operator is not obviously equal to its adjoint due to the rotation operations that it produces, i.e. the operator U_{gw} has to be Hermitian since Equation 3.9 dictates that $U_{gw}U_{gw} = \mathbf{I}$. Despite this limitation, as described in Section 3.1, we can then use it to generate the operator U_{gi} to represent the real weights, and use the HSGS algorithm to generate U_{gw} to represent the binary inputs, since the internal product is a commutative operation, where the order of the operands does not change the result of the operation.

Figure 5 – Circuit of uniformly controlled rotations presented in (MöTTöNEN et al., 2005)



The rotation angles $\alpha_{j,k}^q$ for the uniformly controlled rotations are given in Equations 3.10 and 3.11.
Source: The author (2021)

The uniformly controlled rotation $F_t^k(a, \alpha)$ can be represented as a quantum gate where: k is the number of controlled qubits; t is the target qubit; a, the rotation axis and α_i as the angles. Therefore, the **EO** is defined as sequence of controlled $R_a(\alpha_i)$ gates, where the algorithm has an exponential computational cost to find the sequence of gates (MöTTöNEN et al., 2005).

The first step is to equalize the phases w_i using a sequence of uniformly controlled rotations in cascade, transforming the vector up to the global phase. This is executed using a diagonal N-qubit quantum gate (BULLOCK; MARKOV, 2003). The rotation angles are given in Equation 3.10, where $j = 1, 2, ..., 2^{N-k}$ and $k = 1, 2, ..., N$:

$$\alpha^z_{j,k} = \sum_{l=1}^{2^{k-1}} (w_{(2j-1)2^{k-1}+l} - w_{(2j-2)2^{k-1}+l})/2^{k-1}, \tag{3.10}$$

After applying the uniformly controlled rotation $F^{n-1}_n(y, \alpha)$ with the angles found, we find the desired decomposition (take the product of the non-commuting matrices found from left to right) and then, the rotation angles have the values:

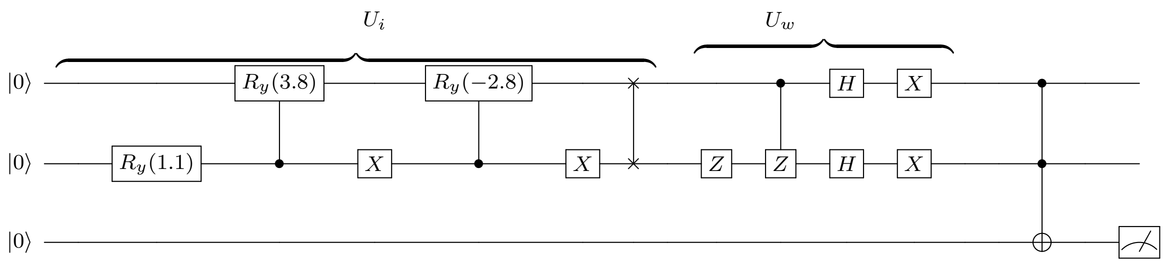
$$\alpha^y_{j,k} = 2\arcsin(\sqrt{\sum_{l=1}^{2^{k-1}} |a_{(2j-1)2^{k-1}+l}|^2} / \sqrt{\sum_{l=1}^{2^k} |a_{(j-1)2^k+l}|^2}), \tag{3.11}$$

where $j = 1, 2, ..., 2^{N-k}$ and $k = 1, 2, ..., N$.

This procedure is used to build the **EO**, and therefore manipulate the probability amplitudes of the quantum system, allowing for the representation of real values on the quantum neuron. The model still uses the HSGS algorithm to input data, but now the inner product is achieved with more information gain than its binary counterpart.

Figure 6 shows an example of quantum neuron with real weights for a 2×2 image represented by the vector $v_i = [1, -1, 1, 1]$ and real weights $w_i = [0.117, -0.77, -0.177, 0.5]$. Due to the restriction pointed in Equation 3.9, for the real weights quantum neuron, the operator U_i is used to input the weights using the **EO** operation and U_w is used to encode the input vector using the HSGS algorithm.

Figure 6 – Example of quantum circuit for the real weights quantum neuron



Input vector $v_i = [1, -1, 1, 1]$, encoded in the circuit by the operator U_i, and weights $w_i = [0.117, -0.77, -0.177, 0.5]$ encoded by U_w.
Source: The author (2021)

Algoritmo 2: Amplitude Encoding Algorithm proposed in (MöTTöNEN et al., 2005) .

```
   Result: Quantum circuit QC;
1  Input: a given vector v with dimension $m = 2^N$ ;
2  Auxiliary variables: vAux and vBeta vectors;
3  Procedure: recursiveComputeBetas(v, vBeta):
4  for $i = 0; i < len(\boldsymbol{v}); i + 2$ do
5      compute square-root of the sum of $v[i]^2$ and $v[i+1]^2$ and store result norm in
         vAux;
6      if norm is 0 then
7          append 0 to vBeta
8      else
9          if $v[i]$ less than 0 then
10             compute $2\pi - 2\arcsin(v[i+1]/norm)$ and store result in vBeta ;
11         else
12             compute $2\arcsin v[i+1]/norm$ and store result in vBeta
13         end
14     end
15     recursiveComputeBetas(vAux, vBeta)
16 end
17 Procedure: generateCircuit(vBeta):
18 for $i = 0; i < len(\boldsymbol{vBeta}); i++$ do
19     if n of controlled qubits is 0 then
20         apply a RY rotation in the qubit with the angle vBeta[i], put the qubit in
             the list of controlled qubits and update the number of controlled qubits.
             Return final quantum circuit QC for vector v.
21     end
22 end
```

3.3 BIAS AND THE SYMMETRY PROBLEM

The quantum perceptron proposed in (TACCHINO et al., 2019) has the property of learning a pattern and its negative counterpart, due to the inner product behavior in the quantum system. The data here is represented by 4×4 images with black (-1) and white (1) pixels. When training for a pattern X, the quantum neuron spontaneously learns the opposite pattern $\hat{X}$ (see Figure 10). Accordingly to (TACCHINO et al., 2019), this behavior reflects the invariance of the encoding states $|\psi_i\rangle$ and $|\psi_w\rangle$ under the addition of a global -1 factor.

We show that this property can be undesirable for classification tasks. The quantum neuron can deceive the target pattern and return a prediction probability that is greater than expected for an input pattern that is far from the original target, but close to its negative.

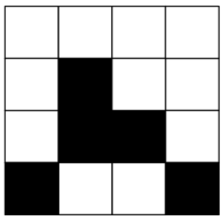
To evidence this problem we trained the patterns 1 and 2 together (see Figure 10) for the same neuron (quantum neuron A), and another quantum neuron for the pattern

3 (quantum neuron B). The optimization procedure used to train the weights of the neurons is the Delta Rule, applied for all quantum neurons for reliability and described in (WIDROW; HOFF, 1960). The delta rule is executed with real values and the weights are binarized for the execution of the HSGS operator only, but continues to be real valued in the neurons using the Encoding algorithm. Although the optimization returns the trained weights as real numbers, for the binary neurons we apply the deterministic binarization $-1 \ \textit{if} \ x < 0 \ \textit{else} \ 1$.

In Equations 3.12 and 3.13 it is possible to observe the weights found by the Delta Rule algorithm considering the quantum neurons A and B. The weights are trained with real numbers but the binarization is applied to generate the states $|\psi_i\rangle$ and $|\psi_w\rangle$. When executing the trained quantum neurons with the test example Y, Fig 7, we find that $B \cdot Y$ has a greater probability output than $A \cdot Y$ (0.14 for quantum neuron A and 0.38 for quantum neuron B). Therefore, although the pattern Y is much closer to the patterns 1 and 2, due to the symmetric behavior, the negative pattern $\hat{3}$ increase the probability activation for the quantum neuron B.

Figure 7 – Example of binary image pattern (left) and its representation in binary matrix (right)

Y =



$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{bmatrix}$$

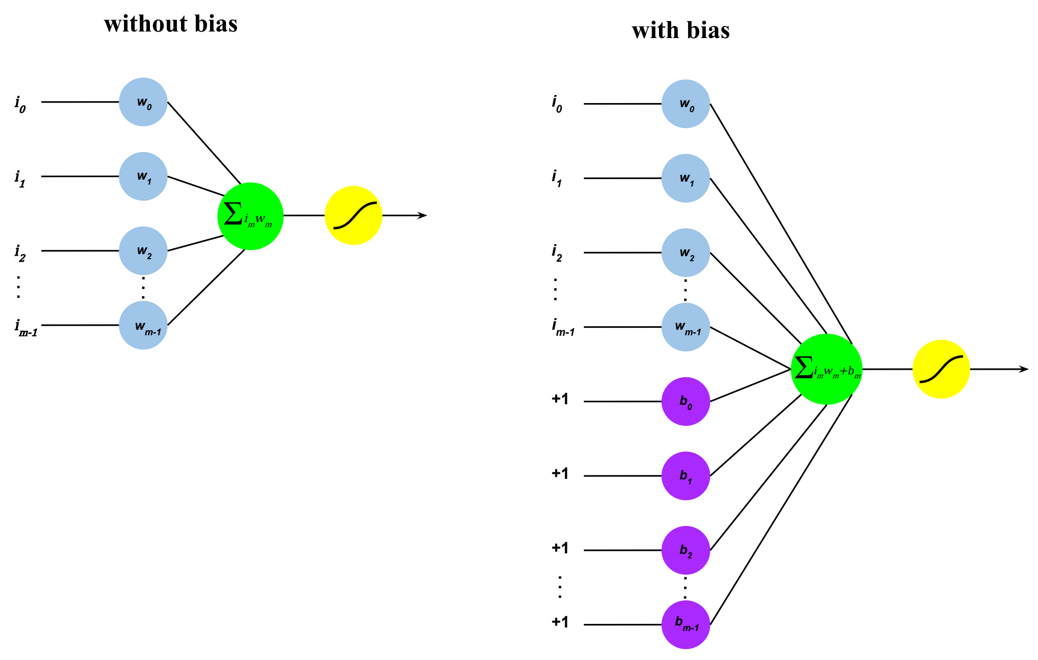
Source: The author (2021)

$$A = \begin{bmatrix} -0.064 & 0.064 & 0.064 & -0.064 \\ 0.064 & 0.487 & 0.487 & 0.064 \\ 0.064 & 0.487 & 0.487 & 0.064 \\ -0.064 & 0.064 & 0.064 & -0.064 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \end{bmatrix} \tag{3.12}$$

$$B = \begin{bmatrix} 0.118 & 0.353 & 0.118 & 0.118 \\ 0.353 & -0.118 & -0.353 & 0.353 \\ 0.353 & -0.118 & -0.353 & 0.118 \\ 0.118 & 0.353 & 0.118 & 0.118 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.13}$$

To address this problem we perform a preprocessing stage in the quantum neuron, adding bias to the input vectors, with the fixed cost of adding only one more qubit in the system. This procedure doubles the number of quantum states, inserting a set of a constant bias of +1, which is the same size of the original input (see Figure 8).

Figure 8 – Quantum neuron without bias (left) and with bias (right)



Source: The author (2021)

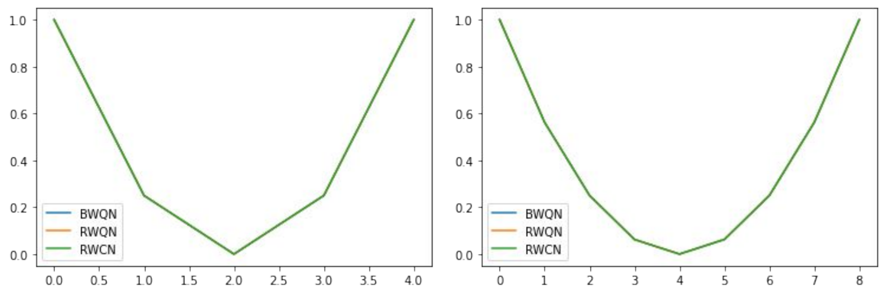
We also test the inner product behavior of the quantum neurons for a range of difference between input and weights, generating random states input vector and then testing from 0 to $2^N + 1$ number of differences, where N is the number of qubits of the quantum system. Figure 9a exhibits the results of this test, showing the symmetry relation between inner product output and the level of difference in input U_i and weights U_w. A more detailed analysis of this behavior can be found in (NETO; FILHO; MONTEIRO, 2020). The output of the inner product decreases when the difference increases until it reaches the point of maximum orthogonality, then the inner product output increases as the difference becomes a mirror of the opposite target. This behavior happens due to the result of the neuron being a measure of probability that can only be positive, so internal products with negative values will be normalized and squared.

In Figure 9b, the variance of the difference between input and weights with bias follow a linear pattern, instead of the symmetry observed before, without bias. Therefore, we can verify the distance from the input pattern and the stored weight without considering the barred input or barred weight as the same pattern.

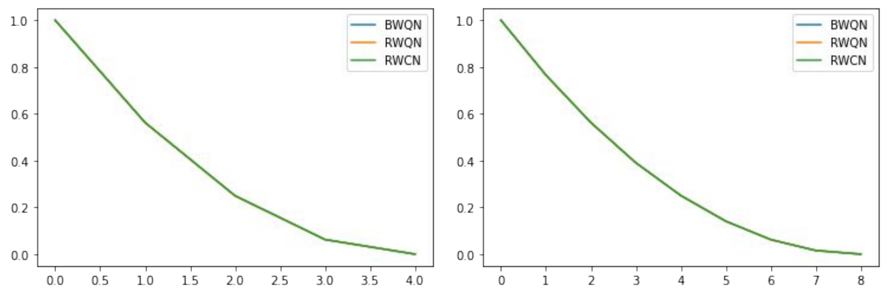
With the weights trained with bias, when testing, the pattern learned is related only to the real target and not its negative counterpart. For each of the quantum neurons A and B, we also trained the biased version quantum neurons $\hat{A}$ and $\hat{B}$. To this extent, we

Figure 9 – Test of the inner product behavior of the quantum neurons for a range of difference between input and weights

(a) Without bias, the quantum perceptron proposed in (TACCHINO et al., 2019) is stimulated maximally to the pattern exactly equal to weight and to the pattern exactly opposite to weight (its negative counterpart), being unable to differentiate both.



(b) With bias, the quantum proposed neuron is able to differentiate the input patterns as they become more and more different from their weight.



The graphs show the neuron output (vertical axis) as a function of the difference between the input vector and the weight vector (horizontal axis) for the quantum neuron without bias (top) and with bias (below), considering $N = 2$ qubits (left) and N = 3 (right). In the quantum neuron without bias, the neuron preserves the symmetry of the output, showing its inability to differentiate a pattern from its completely reverse pattern (i.e. its negative counterpart). When the quantum neuron has bias, the neuron has an ever-decreasing curve as an output, which indicates that the degree of the difference between inputs and weights is inverse to the output stimulus, therefore being able to differentiate a pattern from its completely opposite pattern. To generate these graphs, 1000 completely different binary input values and weights were generated for each of the possible distances. The output value of the neuron on the graph is the average output of these iterations. It is possible to verify that the theoretical quantum neurons BWQN, RWQN and RWCN have average output values exactly the same.
Source: The author (2021)

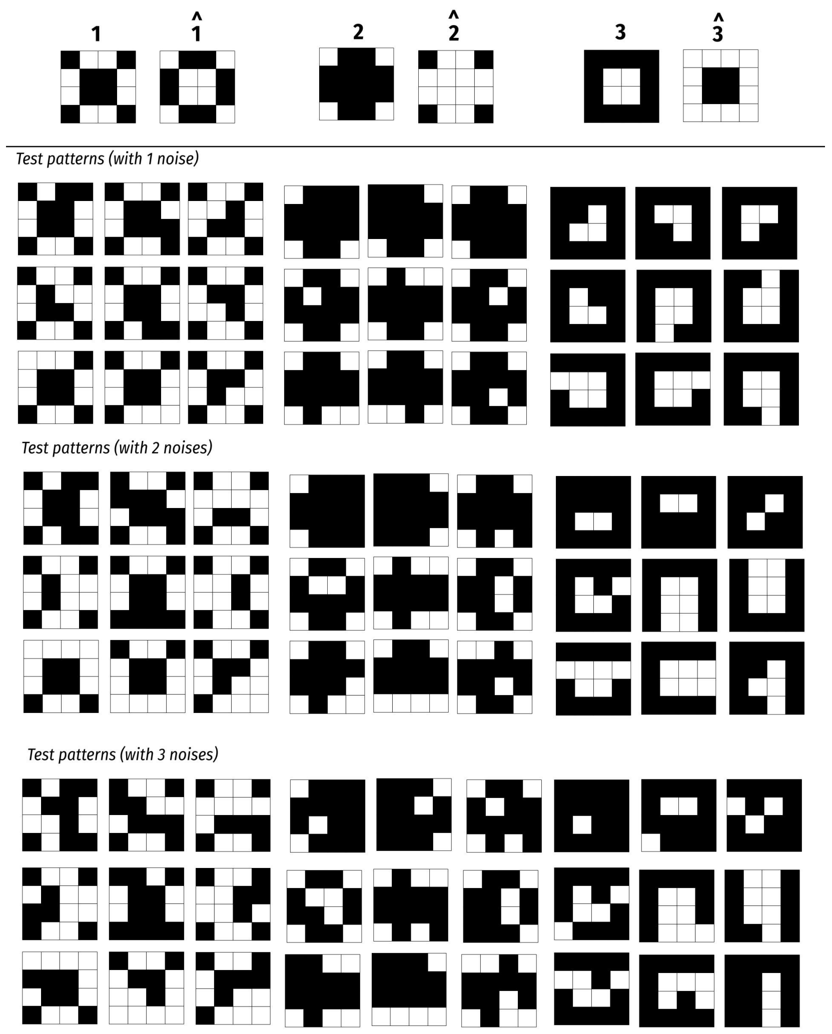
find that the biased approach indeed has an error correction behavior, denoted by the output of these neurons where $\hat{A} \cdot Y = 0.10$ and $\hat{B} \cdot Y = 0.03$.

These experiments have demonstrated the advantage of using the biased approach for the quantum neuron, in order to correct the symmetry problem. Nevertheless, for this approach, the HSGS algorithm shows an issue related to cases that have a value of -1 in the first position of the input or weight vectors, where the algorithm flips all the values to generate the circuit and therefore the pattern learned is the opposite. The experiments consider the opposite pattern in these cases, in order to validate the biased approach proposal, however in real applications the Brute Force algorithm proposed in (TACCHINO et al., 2019) can be used.

3.4 EXPERIMENTS

We conducted experiments with a data set of 4×4 images with three different patterns (cross, X and square) as shown in Figure 10. There is also a variation in the test data set ranging from 1 to 3 noises in the image, to analyze the behavior of the quantum neurons when dealing with different patterns from those learned.

Figure 10 – Database of figures X (pattern 1), Cross (pattern 2) and Square (pattern 3), used in the experiments


1
1̂
2
2̂
3
3̂
Test patterns (with 1 noise)
Test patterns (with 2 noises)
Test patterns (with 3 noises)

At the top, the figures that are used during training in the absence of noise. Next to each pattern, there is its negative counterpart ($\hat{1}$, $\hat{2}$ and $\hat{3}$). Below, noises on three scales are added to the patterns that were used in the test stage to assess the quality of the training.
Source: The author (2021)

The quantum neurons used in the experiments are the Binary Weights Quantum Neuron (BWQN), Real Weights Quantum Neuron (RWQN), Binary Weights Classical Neuron (BWCN) and Real Weights Classical Neuron (RWCN). For each of these quantum neurons, we also test the biased approach, therefore duplicating the quantum states for U_i and U_w with the cost of adding only one qubit.

The quantum neuron is initialized with the input state $|\psi_i\rangle$ and random weights $|\psi_w\rangle$, then the model is executed to evaluate its probability output given by the number of shots with output 1 divided by the total shots, fixed at 8192 shots. The weights are updated during the training process in a quantum-classical approach, where the quantum simulator executes the quantum circuit of the quantum neuron and the Delta Rule algorithm update the weights based on the circuit output. We use Python (v.3.6) for coding and the Qiskit package for development and simulation of the quantum circuits (ALEKSANDROWICZ et al., 2019).

The first set of experiments test a target class for each of the three patterns (X-0, Square-1 and Cross-2), i.e. one neuron for each class, therefore each neuron learns a unique pattern. Table 1 1 shows the average error of each model, measured as the number of wrong predictions divided by the total number of predictions, alongside its standard deviation. Is possible to state that, when patterns have only one noise, all quantum neurons reached an error mean of 0 with a standard deviation of 0, explained by the specialist behavior of the quantum neurons when dealing with only one class of target and low differences form the pattern learned. Nevertheless, its also possible to state the behavior of the biased quantum neurons when dealing with 2 and 3 noises in the test image, which points to a significant error reduction compared to the quantum neurons without bias.

This indicates an error reduction for the Real Weights Quantum Neuron compared to the Binary Weights Quantum Neuron both for 2 and 3 noises. The mean error for 2 noises in the image achieved 0.1209 for RWQN and 0.1222 for BWQN, while for 3 noises the RWQN achieved 0.1308 and BWQN, 0.1407. The variance of the error for the Binary Weights Quantum Neuron is more than double of the Real Weights Quantum Neuron for this experiment, for 2 noises the quantum neurons exhibit a variance of 0.0004 (RWQN) and 0.0019 (BWQN), while for 3 noises the quantum neurons had a variance of 0.0003 (RWQN) and 0.0020 (BWQN). Therefore, expressing a better adjustment, making the proposed quantum neuron more stable, a feature that is highly relevant for machine learning and classification tasks.

The second set of experiments is executed with the target classes in a combined approach, creating a quantum neuron for pairs of classes, to analyze the behavior of the models to deal with multiple patterns. Here we define the 3 subsets of classification for this test: (1) X and Cross (class a) vs Square (class b); (2) X and Square (class a) vs Cross (class b); and (3) X (class a) vs Square and Cross (class b). The training process is designed with the Delta Rule algorithm with a learning parameter of 0.09 and 400 epochs.

In Table 2 it is possible to state that the Real Weights Quantum Neuron with Bias has the best performance for all experiments and noise configurations, with an error mean of 0 found in 7 of 8 experiments with bias, while also reducing the error variance when compared to the Binary Weights Quantum Neuron.

The benefits of the biased approach are more evident in these experiments and can be denoted by the subset of experiments (2) X and Square vs Cross (third column from left to right) and (3) X vs Square and Cross (right column), where for all noise configurations the biased approach had an effect of reducing the error of all quantum neurons. Nevertheless, for the experiment X and Cross (class a) vs Square (class b), the bias has reduced the error for the real weights quantum neurons but increased the error for the binary weights

Table 1 – Results of experiments with unique patterns

Model	Bias Approach	1 Noise (std)	2 Noises (std)	3 Noises (std)
RWQN	Biased	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
RWQN	Unbiased	0.00 (0.00)	0.12 (0.02)	0.13 (0.02)
BWQN	Biased	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
BWQN	Unbiased	0.00 (0.00)	0.12 (0.04)	0.14 (0.04)
RWCN	Biased	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
RWCN	Unbiased	0.00 (0.00)	0.11 (0.00)	0.11 (0.00)
BWCN	Biased	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
BWCN	Unbiased	0.00 (0.00)	0.07 (0.00)	0.07 (0.00)

Each quantum neuron learns a unique pattern, with noise levels ranging between 1, 2 and 3 (pixels changed from the original pattern). The values in the table corresponds to the average error in the training set, measured as the number of wrong predictions divided by the total number of predictions, with its standard deviation between parenthesis.
Source: The author (2021)

quantum neurons. This indicates a faster learning convergence for the proposed quantum neuron, due to the property of storing real information in the quantum states and therefore having more discriminant power than the binary quantum neuron while reducing the learning error caused by the symmetry problem using the biased approach.

In Figure 11, it is possible to visualize the error reduction of the real weights quantum neuron and binary weights quantum neuron, by epoch (0-20) and for the four configurations of data set: (1) X and Cross vs Square; (2) X and Square vs Cross; (3) X vs Square and Cross; and (4) X vs Cross vs Square. For each configuration we tested the biased and unbiased approaches. In the quantum neurons without bias, although the RWQN has a greater error reduction on the first epochs, the BWQN decreases more on the long run for 3 of the 4 data sets tested. Nevertheless, when applying the biased approach, the RWQN has the best performance in the 3 data sets with combined classes, and reach the same error of the BWQN in the individual class neurons (X vs Square vs Cross).

Table 2 – Results of experiments with multiple patterns (classes)

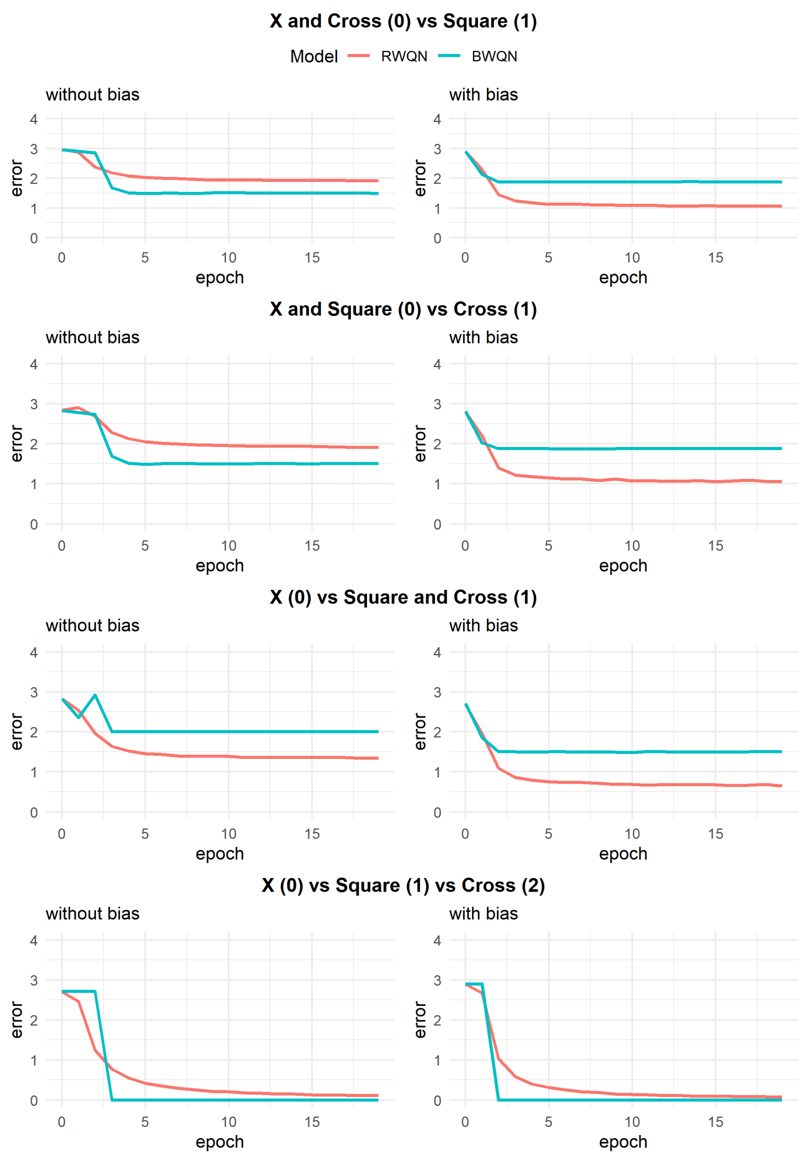
Model	Bias Approach	Subset Data	1 Noise (std)	2 Noises (std)	3 Noises (std)
RWQN	Biased	Subset 1	**0.00 (0.00)**	**0.00 (0.00)**	**0.05 (0.02)**
RWQN	Unbiased	Subset 1	0.33 (0.00)	0.19 (0.03)	0.26 (0.00)
BWQN	Biased	Subset 1	0.18 (0.03)	0.19 (0.03)	0.22 (0.03)
BWQN	Unbiased	Subset 1	0.15 (0.04)	0.14 (0.04)	0.11 (0.02)
RWCN	Biased	Subset 1	0.00 (0.00)	0.00 (0.00)	0.04 (0.00)
RWCN	Unbiased	Subset 1	0.33 (0.00)	0.30 (0.00)	0.26 (0.00)
BWCN	Biased	Subset 1	0.11 (0.00)	0.11 (0.00)	0.18 (0.00)
BWCN	Unbiased	Subset 1	0.07 (0.00)	0.07 (0.00)	0.07 (0.00)
RWQN	Biased	Subset 2	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**
RWQN	Unbiased	Subset 2	0.33 (0.00)	0.22 (0.01)	0.22 (0.01)
BWQN	Biased	Subset 2	0.15 (0.04)	0.16 (0.03)	0.10 (0.00)
BWQN	Unbiased	Subset 2	0.19 (0.04)	0.19 (0.04)	0.22 (0.03)
RWCN	Biased	Subset 2	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
RWCN	Unbiased	Subset 2	0.33 (0.00)	0.22 (0.01)	0.22 (0.02)
BWCN	Biased	Subset 2	0.07 (0.00)	0.07 (0.00)	0.11 (0.00)
BWCN	Unbiased	Subset 2	0.11 (0.00)	0.11 (0.00)	0.18 (0.00)
RWQN	Biased	Subset 3	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**
RWQN	Unbiased	Subset 3	0.04 (0.00)	0.20 (0.02)	0.15 (0.01)
BWQN	Biased	Subset 3	0.01 (0.02)	0.07 (0.02)	0.07 (0.02)
BWQN	Unbiased	Subset 3	0.31 (0.02)	0.25 (0.05)	0.28 (0.05)
RWCN	Biased	Subset 3	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
RWCN	Unbiased	Subset 3	0.04 (0.00)	0.18 (0.00)	0.11 (0.00)
BWCN	Biased	Subset 3	0.04 (0.00)	0.11 (0.00)	0.11 (0.00)
BWCN	Unbiased	Subset 3	0.33 (0.00)	0.30 (0.00)	0.33 (0.00)

The quantum neurons learn a combination of two patterns defined as subsets (1) X and Cross (class a) vs Square (class b); (2) X and Square (class a) vs Cross (class b); and (3) X (class a) vs Square and Cross (class b), with noise levels ranging between 1, 2 and 3 (pixels changed from the original pattern). The values in the table corresponds to the average error in the training set, measured as the number of wrong predictions divided by the total number of predictions, alongside its standard deviation between parenthesis.

Source: The author (2021)

3.5 RUNNING IN A REAL QUANTUM PROCESSOR

Figure 11 – Error by epoch during training of the RWQN and the BWQN



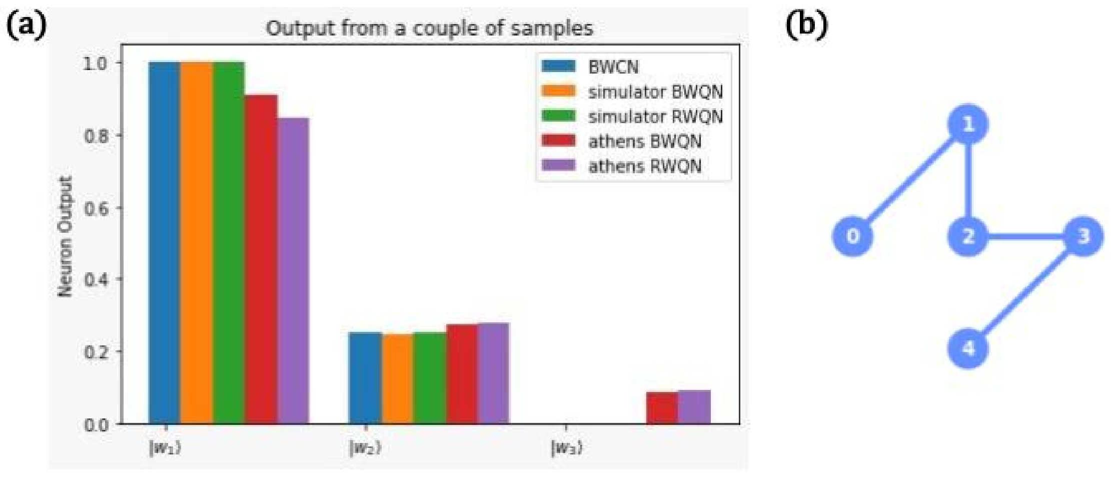
Error during training of the RWQN and the BWQN by epoch in the delta rule weight adjust, measured as the number of wrong predictions divided by the total number of predictions. We analyzed 4 different configurations of binary classifications: (A) X and Cross (class 0) vs Square (class 1); (B) X and Square (class 0) vs Cross (class 1); (C) X (class 0) vs Square and Cross (class 1); and (D) X (class 0) vs Cross vs Square (class 1). For each configuration we tested the biased (right) and unbiased approaches (left). The results show the generalization capacity of the proposed quantum neuron combined with biased approach, when dealing with different patterns.
Source: The author (2021)

In order to evaluate the proposed quantum neuron on real quantum devices, we developed experiments with a noisy intermediate-scale quantum (NISQ) computer, testing the models: binary weights classical neuron (BWCN), binary weights quantum neuron (BWQN), and real weights quantum neuron (RWQN). Three input and weight vectors that generate three different outputs for the neuron were simulated: (1) $v_i = [1, 1, 1, 1]$ and $w_i = [1, 1, 1, 1]$; (2) $v_i = [1, 1, 1, 1]$ and $w_i = [1, 1, 1, -1]$; and (3) $v_i = [1, 1, 1, 1]$, $w_i = [1, 1, -1, -1]$. The circuit to be executed in a determined architecture is generated by a compilation routine performed by the Qiskit tool of IBM and can be replicated from these inputs and weights (ALEKSANDROWICZ et al., 2019).

The quantum neurons are executed in the IBM Athens NISQ computer, available for experimentation through Qiskit, running 8192 shots to get the results for each quantum neuron. The quantum neurons are simulated using the quantum circuit simulator backend QasmSimulator, to compare the expected output of the neurons with the NISQ computer output.

The results of the experiments are shown in Figure 12, where is possible to notice that the outputs of the quantum neurons, running in the quantum device, are similar to the same quantum neurons running on the simulator. The difference between the expected output in the simulator and in the quantum device is not substantial and the neuron output levels for $N = 2$ can be distinguishable even in the presence of noise from the real computer. This indicates the potential for near-future applications as the quantum devices reduce noise and expands the number of qubits available for experimentation.

Figure 12 – Results of proof of concept of the proposed quantum neuron in a real quantum computer



a Neuron outputs (vertical axis) for 3 possible values of the input and weight vectors (horizontal axis, weights $|w_1\rangle$, $|w_2\rangle$ and $|w_3\rangle$) for the models: binary weights classical neuron (BWCN), binary weights quantum neuron (BWQN), and real weights quantum neuron (RWQN). The last two executed both in the classical simulator and in the IBM Athens quantum device. The results demonstrate the effectiveness of the quantum neurons in a real quantum processor, evidenced by the proximity of the simulation output and the quantum device output, therefore indicating the potential for near-future applications. **b** Scheme of IBM Q-5 “Athens” backend quantum processor

Source: The author (2021)

4 QUANTUM MACHINE LEARNING TO SUPPORT MEDICAL DECISION

This chapter presents relevant preliminary results from the application of quantum machine learning in the healthcare industry. We test quantum neurons that can carry an exponential amount of information to a linear number of quantum information units (qubits) using the quantum property of superposition. We compared their performance on the problems of (1) simulating the XOR operator, (2) solving a generic nonlinear problem and (3) classify patients with diabetes. The results of the experiments shows that a single quantum neuron is capable of reach 100% accuracy rate in the XOR problem and 100% accuracy rate in a nonlinear dataset, demonstrating the plausibility of the quantum neurons in modelling non linearly separable problems. The quantum neurons achieved 76% accuracy rate and 88% AUC-ROC in the problem of diagnosing diabetes. These findings indicates that a single quantum neuron has good generalization capacity, demonstrating potential for use in a near-future application in healthcare.

4.1 MACHINE LEARNING AND BIG DATA IN HEALTHCARE

Health can be defined not just as the absence of diseases but also in a broader approach, where the the individual state of well-being is relevant, relating to the physical, mental and social health (UN, 1946; BURTON-JEANGROS et al., 2015). This implies that the individual health is a continuous state in time, with variances and events that has a sequential effect. Therefore, identifying the patterns that leads to a health state, a care company can prevent the demand and propose an intervention to the individuals before they reach a critical point. This improves the quality of their services, promote cost-efficient management while reducing risks for the health of the patients.

With the growing development of information systems for the management of healthcare institutions, the storage of care data such as medical diagnosis, personal characteristics, events and procedures during hospitalization and infrastructure of the health institutions reached a point of pettabytes or more of storage in some locations such as the Kaiser Permanente network in California, United States (HARTZBAND, 2019). All this information can be used to promote a better health for patients, lower costs for private and public healthcare companies, improve the patient experiences during hospitalization and enhance the workflow for healthcare practitioners (FLöTHER MURPHY; SOW, 2020).

Machine learning has been used in healthcare as a tool to support the decision-making process, with companies developing strong predictors from lots of weak attributes. Although machine learning is widely exploited in image diagnosis such as x-ray for identifying lung and breast cancer, the data mining of healthcare processes is still not fully exploited. This is due to the complexity of these problems when compared to image recognition, for example the prediction of patient hospitalization can explore many different measures of data, e.g. patient previous health conditions, food consumption habits, physical movement habits, family relatives comorbidities, environmental and healthcare conditions in the neighborhood, image diagnosis, genomics, wearable devices and more (SOLENOV; BRIELER; SCHERRER, 2018). All this information is relevant data that can be used to train machine learning learning algorithms. These applications require large databases for continuous storage, and the data preprocessing techniques can easily expand exponentially the attributes for the algorithm training when generating longitudinal information to be encoded in different time stamps and levels of aggregation. The need of computing power and model degrees of freedom to learn the patterns from these sources of big data is also growing and quantum computing can be used as a tool to address this need.

Some articles have applied quantum computing to health problems, although it is recognized that the area is still in development, requiring the test and simplification of quantum circuits so that they are actually capable of running on real processors (SHAIKH; ALI, 2016; SOLENOV; BRIELER; SCHERRER, 2018). Wiebe, Kapoor and Svore (WIEBE; KAPOOR; SVORE, 2015) have proposed Quantum Nearest-Neighbor Algorithms to classify breast cancer, heart disease, and diabetes in patients, showing that their algorithms are competitive with classical algorithms while also leading to polynomial reductions in query complexity relative to the corresponding classical algorithms. Gupta *et al* (GUPTA et al., 2021) made a comparative study of the performance of quantum machine learning (QML) algorithms with deep learning (DL) for diabetes prediction, using the PIMA Indian Diabetes dataset, showing that the DL models has a high diabetes prediction accuracy when compared with the developed QML and existing state-of-the-art models. Nevertheless, they highlight that the performance of the QML model has been found satisfactory and comparable with existing literature.

In 2021 IBM has announced its first private commercial sell of a quantum computer through a partnership with the Cleveland Clinic, situated in Ohio (US), to develop quantum computing applications for the healthcare center (HEALTHCARE..., 2021). The aim of this cooperation is to address the need for innovative discoveries, harnessing the power of quantum computers to promote advances in research areas that require analysis of large amounts of data such as genomics, cell transcriptomics, population health and drug discovery (HEALTHCARE..., 2021), indicating the relevance of the field to healthcare applications. Nevertheless, although there is quantum machine learning applications being developed, there is still need to investigate the fitness of these algorithms to healthcare problems (SOLENOV; BRIELER; SCHERRER, 2018).

4.2 CONTINUOUSLY VALUED QUANTUM NEURON

Parametric models with real valued parameters have a greater performance than its counterparts with binary values. This is due to the gain in representing information with real values, therefore having a larger space for memory association. To develop a quantum neuron capable of store real weights and preserve the gain with the superposition property, we need to encode the information in the probability amplitudes of the quantum system.

Möttönen *et al.* (MöTTöNEN et al., 2005) proposed an algorithm to store real values in the amplitudes of quantum states. However, this operator is not equal to its adjoint due to the rotation operations that it produces, therefore it can only be used to generate the operator U_{gi} or the operator U_{gw}, while the HSGS algorithm is used to generate the other one. Mangini *et al.* (MANGINI et al., 2020) exploit the Sign-Flip Algorithm proposed in (TACCHINO et al., 2019) to implement a phase encoding operation on the quantum circuit and encode continuously valued information in the amplitudes of the quantum states through the usage of rotation in the qubits. This procedure makes possible to encode both input and weight vectors with continuous values, producing a quantum neuron that is suitable for classification problems with real valued data.

The input for the phase encoding is the classical information to be encoded, given by $\Theta = (\theta_0, ..., \theta_{N-1})$ with $\theta_i \in [0, \pi]$, whereas weights are given by $\Phi = (\phi_0, ..., \phi_{N-1})$ with $\phi_i \in [0, \pi]$. The corresponding vectors for inputs and weights are defined as $\vec{i}$ and $\vec{w}$:

$$\vec{i} = \begin{pmatrix} e^{i\theta_0} \\ e^{i\theta_2} \\ \vdots \\ e^{i\theta_{N-1}} \end{pmatrix}, \vec{w} = \begin{pmatrix} e^{i\phi_0} \\ e^{i\phi_2} \\ \vdots \\ e^{i\phi_{N-1}} \end{pmatrix} \tag{4.1}$$

The information is then carried to the amplitudes of the quantum states $|\psi_i\rangle$ and $|\psi_w\rangle$:

$$|\psi_i\rangle = \frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} i_k |k\rangle \quad , \quad |\psi_w\rangle = \frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} w_k |k\rangle \tag{4.2}$$

Where $n = \log_2 N$ qubits and the states $|k\rangle$ denote the computational basis states of n qubits ordered by increasing binary representation, $[|00...0\rangle, |00...1\rangle,$
$..., |11...1\rangle]$. The similarity between the input and weight vectors is given by the inner product in Equation 4.3, which is equivalent to the scalar product between the input vector $\vec{i}$ and the conjugated of the weight vector $\vec{w}$, $\vec{w}*$.

$$\begin{aligned} \langle\psi_w| |\psi_i\rangle &= \frac{1}{2^n} \sum_{k,j=0}^{2^n-1} i_k w_j^* \langle j| |k\rangle = \frac{1}{2^n} \vec{i} \cdot \vec{w}^* \\ &= \frac{1}{2^n} (e^{i(\theta_0 - \phi_0)} + \cdots + e^{i(\theta_{2^n-1} - \phi_{2^n-1})}) \end{aligned} \tag{4.3}$$

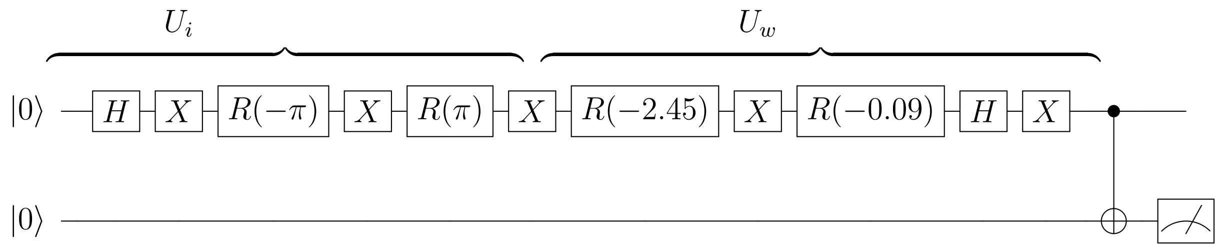
Considering that probabilities in quantum systems are given by the square modulus of the wave-function amplitudes, the Equation 4.4 represents a nonlinear activation function of the phase encoding quantum neuron.

$$|\langle\psi_w|\,|\psi_i\rangle|^2 = \frac{1}{2^n} + \frac{1}{2^{2n-1}} \sum_{i<j}^{2^n-1} \cos\left((\theta_j - \phi_j) - (\theta_i - \phi_i)\right) \tag{4.4}$$

The quantum circuit for implementing the phase encoding quantum neuron is similar to the HSGS quantum neuron circuit defined in Figure 3. The difference for the phase encoding quantum neuron is at U_i and U_w, where the quantum operators used to encode the information in the probability amplitudes are a combination of X gates and multi-controlled phase shift gate $C^{n-1}R(\theta)$ rotation gates.

The Phase Encoding Algorithm transforms the probability amplitudes of the quantum states such that they represent continuously valued information in the range $[0, \pi]$. For each of the quantum state in computation basis, the first step is to identify the positions of the state where the qubit is in $|0\rangle$ and apply the $\mathbf{X}$ gate to that position to flip it to the state $|1\rangle$. Then, if the size of the vector is greater than 2 a multi-controlled rotation $C^{n-1}R(\theta)$ is applied, where θ is the continuous value in the range $[0, \pi]$, otherwise only the rotation $R(\theta)$ is applied. The last step is to reverse the inversion of the qubits in the first step, applying the $\mathbf{X}$ gate in the positions where the qubit is in $|0\rangle$. Figure 13 shows an example of the phase encoding quantum neuron implemented using the phase encoding algorithm to input data and weights in the quantum circuit.

Figure 13 – Example of quantum circuit for the continuously valued quantum neuron



Inputs $v_i = [-1, 1]$, encoded in the circuit by the operator U_i, and weights $w_i = [0.7828, 0.0287]$ encoded by U_w.
Source: The author (2021)

4.2.1 Preprocessing Strategies for The Continuously Valued Quantum Neuron

The application of preprocessing techniques is inspired by the notion of hypercomplex numbers, where we calculate the hyperspherical representations of the input vector to represent them as high-dimensional vectors. Some articles propose the use of hypercomplex numbers in pattern recognition problems (VALLE; LOBO, 2021). Inspired by the operations of these numbers, we define the calculation of the angle λ and radius σ preprocessing strategies of the paired numbers of an input vector v_k of size n, $k = 0, ..., n-1$, generating v'_k.

Therefore, for each subsequently pair of numbers in input v_k, the *angle preprocessing strategy* applies the function $\arctan(\frac{v_k}{v_{k+1}})$, the *radius preprocessing strategy* executes the function $\sqrt{v_k^2 + v_{k+1}^2}$, and the combination of *angle and radius preprocessing strategy* executes the angle preprocessing strategy resulting v'_k and then append the radius preprocessing strategy to the output vector v'_k. For all strategies, we also couple to v'_k in the final execution the angles resulting from the functions $\sqrt{\sum_{j=0}^{n-1} v_k^2}$ and $\arcsin((n-1)/\sqrt{\sum_{j=0}^{n-1} v_k^2})$.

4.3 EXPERIMENTS AND RESULTS

We compare the behaviour of the Continuously Valued Quantum Neuron (CVQN) and the usage of the preprocessing strategies of angle, radius and the combination of both, with the behaviour of the Real Weights Quantum Neuron (RWQN) and the Binary Quantum Neuron (BQN) proposed by Tacchino *et al.* (TACCHINO et al., 2019), which are describe in chapter 3. In the first set of experiments we test two datasets to analyse the nonlinearity of the quantum neurons, the xor problem dataset and the nonlinear real valued dataset. In the second set of experiments we test the application of these algorithms in a healthcare problem, the classification of patients with diabetes. Each quantum neuron is trained with different parameters of learning rate (0.1, 0.02), neuron threshold (0.1, 0.3, 0.5, 0.7 and 0.9), and using bias or not using bias. The dataset is split in 80% for training and 20% for test. The results shows the average accuracy with the standard deviation, and also presents the best values found for the other metrics.

The three proposed strategies of preprocessing the input vector were also tested in the CVQN during training to identify some gain by applying the radius transformation, angle transformation or a combination of both. For each quantum neuron is performed a search grid for the delta rule learning rate, the neuron threshold and the usage of the biased approach or not. Python (v.3.6) is used for coding and the Qiskit package for development and simulation of the quantum circuits (ALEKSANDROWICZ et al., 2019).

To analyse the performance of the models, we measure the metrics of: *accuracy*, defined as the number of correct prediction divided by the total number of predictions; *precision*, as the number of true positives divided by the sum of true positives and false positives, which measure the proportion of positive identifications that were corrected predicted; *recall*, as the number of true positives divided by the sum of true positives and false negatives, measuring the proportion of actual positives that were correctly identified; and the *f1-score*, which is two times the product of precision and recall divided by the sum of precision and recall, as a measure of balance between precision and recall; *AUC-ROC*, defined as the area under the roc curve; and the *KS* and *KS p-value*, the Kolmogorov-Smirnov test of hypothesis to observe if there is a statistical significant difference between

Table 3 – Inputs and outputs of the XOR problem.

Input 1	Input 2	Output
0	0	0
0	1	1
1	1	0
1	0	1

Source: The author (2021)

the classes for the predicted outputs. Additionally, we also measure the error by epoch of the quantum neurons as the number of incorrect predictions divided by the total number of predictions.

For the datasets with real valued input data, we make a binarization of the data for inputing data in the BQN and the RWQN, such that number $x \in \mathbb{R}$ is approximated to its binary representation x' to 4 decimal places:

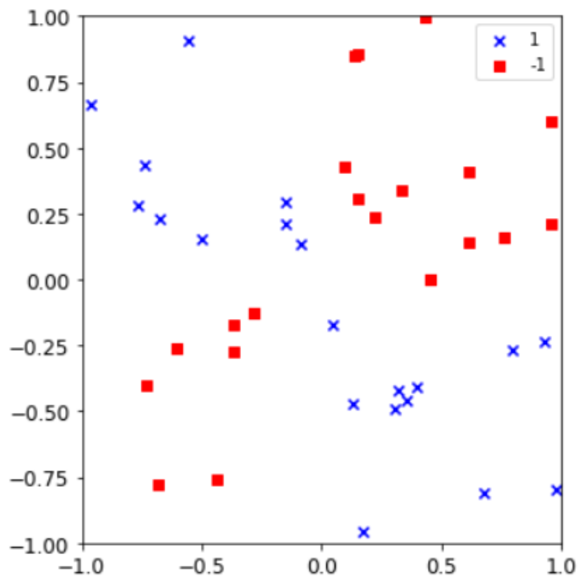
$$x' = \sum_{i=0}^{4} b_{-i} \cdot \frac{1}{2^i} \tag{4.5}$$

The first experiment has the purpose of assess the nonlinearity of the CVQN and RWQN and compare it with the BQN. The dataset used for this experiment consists of 4 patterns and two classes, representing the exclusive-OR (XOR) problem (True XOR True = False (class 0), True XOR False = True (class 1), False XOR True = True (class 1), and False XOR False = False (class 0) (see Table 3). Because the inputs are not linearly separable, it takes more than just a processing unit (neuron) to perform the correct classification, so a neural network is needed (MINSKY; PAPERT, 1969). We therefore test the ability of each quantum neuron to tackle this problem.

The second experiment is executed with a artificial nonlinear dataset with 4 real inputs values, ranging from -1.0 to 1.0, to evaluate the performance of the quantum neurons on real valued data. The data distribution is presented in figure 14, where is possible to observe that a single line could not separate the classes and therefore the dataset is suitable for the experiment. The dataset consists of 20 examples, 9 of class 0 (blue dots) and 11 of class 1 (red dots).

To analyse the applicability of the quantum neurons for healthcare problems, the third experiment is conducted with the Pima Diabetes Dataset, which consists of information from the National Institute of Diabetes and Digestive and Kidney Diseases (SMITH et al., 1988). The objective of this problem is to diagnostically classify whether a patient has diabetes or not, based on diagnostic measurements such as glucose level, blood pressure, age, number of pregnancies and other as presented in Table 4 . The target class measures whether the patient has diabetes (268 individuals) or not (500 individuals). A constraint particular to this dataset is that all patients are females of at least 21 years old. For

Figure 14 – Distribution of the real valued nonlinear dataset



The real valued nonlinear dataset used for training and testing the quantum neurons, with patterns for class 0 in blue color and patterns for class 1 in red color. It is possible to observe that a single line could not separate the classes.
Source: The author (2021)

Table 4 – Description of the Pima Diabetes dataset variables

Variable	Minimum Value	Maximum Value	Type
Pregnancies	0	17	Integer
Glucose	0	199	Integer
Blood Pressure	0	122	Integer
Skin Thickness	0	99	Integer
Insulin	0	846	Integer
Body Mass Index	0	67.1	Real
Diabetes Pedigree	0.08	2.42	Real
Age	0	81	Integer
Outcome	0	1	Integer

Values in the table represent minimum value, maximum value and type of each feature in the dataset.
Source: The author (2021)

each feature we applied a min-max normalization to put the data in the range between zero and one. We also tested the performance of the Classical Perceptron (CP) and the Multilayer Perceptron (MLP) for comparison with the quantum neurons. For MLP we performed a grid search on the activation functions (tanh and relu), alpha, learning rate, solver (stochastic gradient descent and adam) and hidden layer sizes ((50, 50, 50), (50, 100, 50) and (100,)), whereas for the CP we tested the regularization terms L1, L2 and elasticnet.

4.3.1 Nonlinear behaviour experiment

The experiments with the XOR problem in table 5 shows that all three quantum neurons were able to achieve an accuracy of 100% and KS value of 1.0, statistically significant

Table 5 – Results of experiments with the XOR problem for each quantum neuron and preprocessing strategy

Model	Preprocess. Strategy	Best Accuracy	Average Accuracy (std)	Min Accuracy	Best Precision	Best Recall	Best F1-Score	Best AUC-ROC	Best KS	Best KS p-value
BQN	-	1.00	0.82 (0.23)	0.25	1.00	1.00	1.00	1.00	1.00	0.00
RWQN	-	1.00	0.85 (0.23)	0.50	1.00	1.00	1.00	1.00	1.00	0.00
CVQN	Original	0.50	0.50 (0.00)	0.50	0.50	1.00	0.67	0.80	0.70	0.01
CVQN	Angle	1.00	0.86 (0.12)	0.75	1.00	1.00	1.00	1.00	1.00	0.00
CVQN	Radius	0.70	0.51 (0.05)	0.50	0.67	1.00	0.73	0.80	0.60	0.05
CVQN	Angle and Radius	1.00	0.43 (0.40)	0.00	1.00	1.00	1.00	1.00	1.00	0.00

The values in the table corresponds to the evaluation metrics of best accuracy, average accuracy with standard deviation, minimum accuracy, best precision, best recall, best f1-score, best AUC-ROC score and best Kolmogorov-Smirnov test for classes and predicted probabilities with its p-value.
Source: The author (2021)

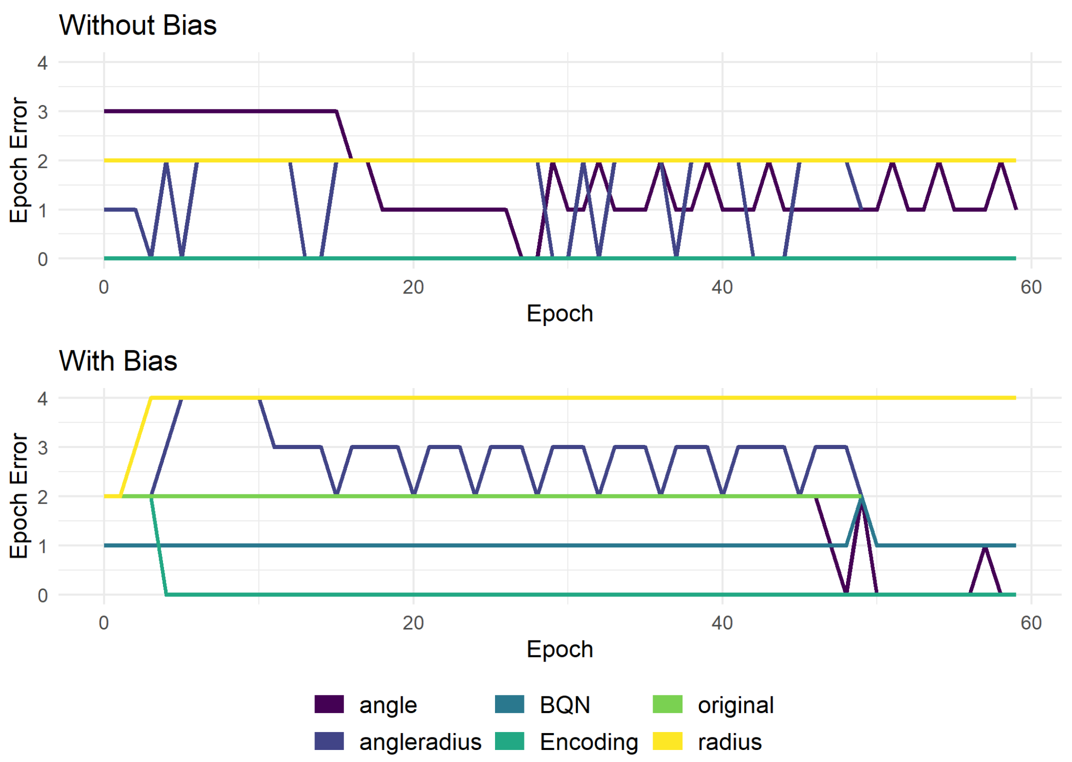
at p<0.00. For the CVQN, the best results were found when the angle preprocessing strategy (100% best accuracy and KS 1.0, significant at p<0.00) and the combinations of the angle and radius preprocessing (100% best accuracy and KS 1.0, significant at p<0.00) were employed. They also show good precision, recall and F1-score, reaching 100%, with highlight to the angle and the combination of angle and radius preprocessing strategies for the CVQN. The performance of the BQN can be explained by the binary nature of the input values and therefore an experiment with real valued input data is also presented forward in the paper.

Analysing the error by epoch in figure 15, in the test without bias, all quantum neurons were able to reach zero error, with attention to the BQN and the RWQN that stayed at zero error since the first epoch and the preprocessing strategies of angle and the combination of angle and radius that showed a variance during training but could also reach zero error. For the tests with bias, the only the CVQN (with angle preprocessing strategy) and the RWQN were able to reach zero error.

In figure 16 is possible to observe the outputs from a trained CVQN, which combines the angle and radius transformation for the input vector and has a learning rate of 0.1, a neuron threshold of 0.7 and without bias. The patterns from class 0 are under the threshold value whereas the patterns from class 1 are above the threshold value. This quantum neuron achieved zero error and the figure illustrates its functioning when dealing with nonlinear patterns.

The results of the experiments with the artificial nonlinear dataset (see Table 6) shows that the CVQN has the top performance in best accuracy (100% and KS 1.0, significant at p<0.00), average accuracy (77%), precision (100%), recall (100%) and f1-score (100%), when compared to the BQN and RWQN. The original CVQN without a preprocessing strategy and the combination of angle and radius strategies shows the best performance among the processing strategies for the CVQN model. This indicates a better adjustment of the CVQN model for real valued input data, due to the continuous valued information that this model can carry in inputs and weights . Although the BQN and the RWQN

Figure 15 – Error by epoch for the best weights found for each quantum neuron in the XOR problem experiment



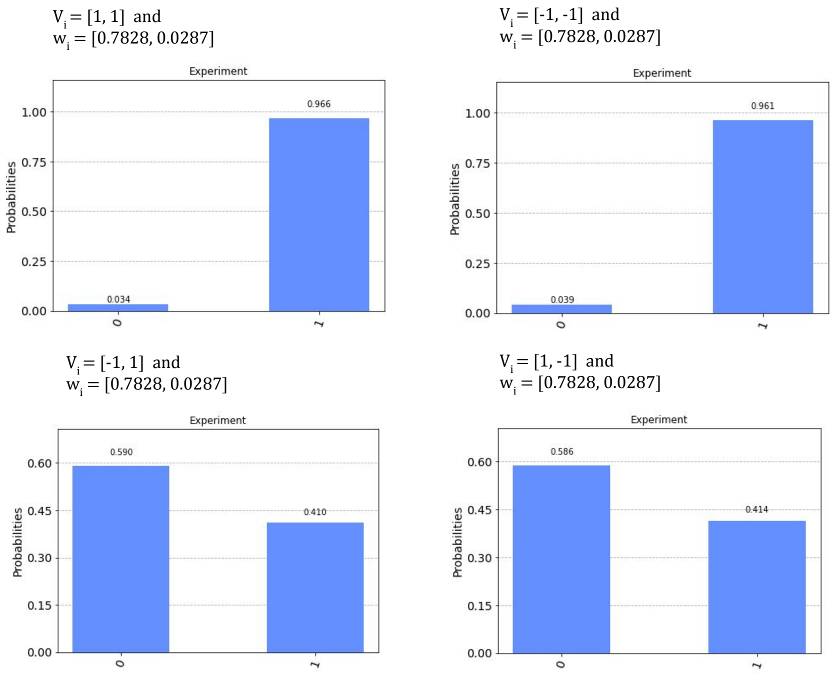
In the graph above, without bias, the original CVQN without a preprocessing strategy stays at 2 error, along with the CVQN with the radius preprocessing strategy and the RWQN. In the graph bellow, with bias, the CVQN with the angle preprocessing strategy initiates at error 2 and then descend to zero error at epoch 48, whereas the RWQN starts at error 2 and drops to zero error at epoch 4.
Source: The author (2021)

have shown a lower performance, it is relevant that they were able to reach 70% and 71% maximum accuracy and 66% and 57% maximum f1-score, respectively.

4.3.2 Diabetes Classification

Table 7 shows the results of the experiments, where the CVQN achieved the top performance in best accuracy (76%) among the quantum neurons and the best AUC-ROC (88%) among all models, including the classical MLP. Extremely low or high values of precision and recall and are explained by the tendency of classifying one prevalent class. Therefore we focus on analysing the best accuracy rate and AUC-ROC. All quantum neurons achieved better accuracy than the CP (63%), although the MLP performed better than the quantum neurons in best accuracy (80%), the CVQN combined with the radius preprocessing reached the best AUC-ROC (88%). This indicate the advantage of the quantum neurons over the classical perceptron and incite the further development of Quantum Neural Networks.

Figure 16 – Example of circuit outputs of the CVQN



Outputs of the CVQN trained in the XOR problem, combining the angle and radius transformation for the input vector, with a learning rate of 0.1, a neuron threshold of 0.7 and without bias. Input vectors represent the patterns for class 0 (bottom) and patterns for class 1 (up).
Source: The author (2021)

Table 6 – Results of experiments with the nonlinear dataset problem for each quantum neuron and preprocessing strategy

Model	Preprocess. Strategy	Best Accuracy	Average Accuracy (std)	Min Accuracy	Best Precision	Best Recall	Best F1-Score	Best AUC-ROC	Best KS	Best KS p-value
BQN	-	0.70	0.55 (0.07)	0.43	0.65	1.00	0.66	0.72	0.52	0.01
RWQN	-	0.71	0.58 (0.06)	0.43	1.00	0.67	0.57	0.66	0.37	0.16
CVQN	Original	1.00	0.77 (0.16)	0.51	1.00	1.00	1.00	1.00	1.00	0.00
CVQN	Angle	0.86	0.66 (0.14)	0.29	1.00	1.00	0.86	1.00	1.00	0.00
CVQN	Radius	0.86	0.61 (0.12)	0.43	1.00	1.00	0.80	0.88	0.67	0.00
CVQN	Angle and Radius	1.00	0.69 (0.18)	0.43	1.00	1.00	1.00	1.00	1.00	0.00

The values in the table corresponds to the evaluation metrics of best accuracy, average accuracy with standard deviation, minimum accuracy, best precision, best recall, best f1-score, best AUC-ROC score and best Kolmogorov-Smirnov test for classes and predicted probabilities with its p-value.
Source: The author (2021)

Table 7 – Results of experiments with the Pima Diabetes dataset for each quantum neuron and preprocessing strategy

Model	Preprocess. Strategy	Best Accuracy	Average Accuracy (std)	Min Accuracy	Best Precision	Best Recall	Best F1-Score	Best AUC-ROC	Best KS	Best KS p-value
BQN	-	0.68	0.52 (0.15)	0.32	0.40	0.46	0.43	0.57	0.29	0.00
RWQN	-	0.69	0.66 (0.05)	0.30	0.32	0.96	**0.48**	0.33	0.41	0.00
CVQN	Original	0.72	0.68 (0.02)	0.54	0.40	0.46	0.43	0.64	0.57	0.00
CVQN	Angle	0.68	0.66 (0.03)	0.53	0.70	0.27	0.39	0.70	0.40	0.00
CVQN	Radius	**0.76**	0.65 (0.09)	0.33	0.87	0.36	0.45	**0.88**	**0.64**	0.00
CVQN	Angle and Radius	0.68	0.66 (0.05)	0.50	0.21	0.08	0.11	**0.84**	0.56	0.00
CP	Original	**0.80**	0.63 (0.00)	0.63	0.66	0.03	0.06	0.51	0.20	0.09
MLP	Original	0.80	0.70 (0.04)	0.65	0.77	0.66	**0.71**	**0.80**	**0.65**	0.00

The values in the table corresponds to the evaluation metrics of best accuracy, average accuracy with standard deviation, minimum accuracy, best precision, best recall, best f1-score, best AUC-ROC score and best Kolmogorov-Smirnov test for classes and predicted probabilities with its p-value.
Source: The author (2021)

5 CONCLUSIONS AND FUTURE WORK

In this work we proposed a new model of quantum neuron with real-valued weights (RWQN), combining the hypergraph states generation subroutine proposed by Tacchino *et al.* (TACCHINO et al., 2019) to generate the operator U_{gw} to represent the binary inputs and the uniformly controlled rotations proposed by Möttönen *et al.* (MöTTöNEN et al., 2005) which generates the operator U_{gi} that represents the real weights. These models are able to carry an exponential amount of information to a linear number of quantum information units (qubits) using the quantum property of superposition, and the results shows that when combining the quantum neuron with real-valued weights and the biased approach, we find the best performance in the classification problems. The variance error reduction of the proposed models, compared to the quantum neuron with binary weights, is another benefit of this model, indicating more stability in classification tasks. When executing the quantum neuron models in the NISQ computer, we found that the expected output given by the simulation is close to the real output executed in the quantum device, even in the presence of noise, indicating that the proposed model has the potential to be executed in real-world applications.

We also compared the performance of the proposed Real Weights Quantum Neuron (RWQN) with the Continuously Valued Quantum Neuron (CVQN) proposed by Mangini *et al* (MANGINI et al., 2020) and the Binary Quantum Neuron proposed by Tacchino *et al.* (TACCHINO et al., 2019), while also testing the performance of proposed preprocessing strategies based on the notion of hypercomplex numbers. We executed experiments with these algorithms on the problems of simulating the XOR operator, solving a generic nonlinear problem and a test on a healthcare problem, the prediction of patients with diabetes. The results of the experiments showed that a single quantum neuron, the proposed RWQN and the CVQN, are capable of reaching 100% accuracy rate in the XOR problem and the CVQN reaches 100% accuracy rate in the nonlinear dataset, demonstrating the plausibility of the quantum neuron in modelling non-linearly separable problems, while also exhibiting a 76% accuracy rate and 88% AUC-ROC in the problem of diagnosing diabetes. Hence, there is evidence that the quantum neurons are a viable modelling approach for healthcare problems of classification. The preprocessing strategies has also shown evidence that it can be useful as a method for boosting up the performance of the CVQN in some cases, presented by the performance of radius and the combination of radius and angle preprocessing strategies in the Pima Diabetes dataset.

We show that the healthcare industry can benefit from investments in this area, while experimenting with more complex models as the quantum devices advances. Limitations of this work include evaluating the performance of the models on quantum devices, given the limited capacity of current quantum systems available. Furthermore, the depth of the circuit is a challenge that can be explored using other methods of amplitude encoding. Future work can explore the behavior of quantum neural networks by connecting these quantum neuron models and also developing encoding algorithms that can be applied in for generating U_i to encode the input data and U_w to encode the weights.

5.1 SUMMARY OF RESULTS

- Discussion of the restriction imposed to the Binary Quantum Neuron of (TACCHINO et al., 2019);
- Analysis of the symmetry problem in the quantum neurons and proposition of a biased approach to address it;
- Proposition of the Real Weights Quantum Neuron (RWQN) based on uniformly controlled rotations;
- Proposition of preprocessing strategies based on the notion of hypercomplex numbers;
- Analysis of the suitability of quantum neurons for healthcare problems, in specific the prediction of diabetes, where the experiments shows satisfactory results for the quantum neurons that can carry real or continuous values in the quantum states;

5.2 FUTURE WORK

Future work can explore the behaviour of quantum neural networks by connecting these quantum neurons. This could serve as a basis for real value input neurons with binary weights on a network, therefore allowing tests with more complex data sets.

An analysis of the computational cost of the algorithms used to encode the information on quantum states amplitudes would be of great benefit, along the analysis of circuit dept and time of execution. The circuit architecture can also be optimised in future efforts exploring available approaches that optimise specific circuit architectures (ZHANG et al., 2018; ZHANG et al., 2019).

The quantum neurons can be used for classifying other problems like image classification for identifying breast cancer, forecasting of time series data such as stock price and others. Finally, the preprocessing strategies based on hypercomplex numbers could be further explored for other quantum neural models.

REFERENCES

ALEKSANDROWICZ, G.; ALEXANDER, T.; BARKOUTSOS, P.; BELLO, L.; BEN-HAIM, Y.; BUCHER, D.; CABRERA-HERNÁNDEZ, F.; CARBALLO-FRANQUIS, J.; CHEN, A.; CHEN, C. et al. Qiskit: An open-source framework for quantum computing. *Accessed on: Mar*, v. 16, 2019.

ALTAISKY, M. Quantum neural network. *arXiv preprint quant-ph/0107012*, 2001.

BIAMONTE, J.; WITTEK, P.; PANCOTTI, N.; REBENTROST, P.; WIEBE, N.; LLOYD, S. Quantum machine learning. *Nature*, v. 549, p. 195–202, 2017.

BOYER, M.; BRASSARD, G.; HøYER, P.; TAPP, A. Tight bounds on quantum searching. *Fortschritte der Physik*, v. 46, n. 4-5, p. 493–505, 1998. Available at: <https://onlinelibrary.wiley.com/doi/abs/10.1002/>.

BRIEGEL, H. J.; BROWNE, D. E.; DüR, W.; RAUSSENDORF, R.; NEST, M. Van den. Measurement-based quantum computation. *Nature Physics*, Springer Science and Business Media LLC, v. 5, n. 1, p. 19–26, Jan 2009. ISSN 1745-2481. Available at: <http://dx.doi.org/10.1038/nphys1157>.

BROWN, T. B. et al. *Language Models are Few-Shot Learners*. 2020.

BULLOCK, S. S.; MARKOV, I. L. Smaller circuits for arbitrary n-qubit diagonal computations. *arXiv preprint quant-ph/0303039*, 2003.

BURTON-JEANGROS, C.; CULLATI, S.; SACKER, A.; BLANE, D. A life course perspective on health trajectories and transitions. Springer Nature, 2015.

CAO, Y.; GUERRESCHI, G. G.; ASPURU-GUZIK, A. *Quantum Neuron: an elementary building block for machine learning on quantum computers*. 2017.

FARHI, E.; GOLDSTONE, J.; GUTMANN, S.; SIPSER, M. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

FLöTHER MURPHY, M.; SOW. *Exploring Quantum Computing Use Cases for Healthcare*. [S.l.], 2020.

GROVER, L. K. A fast quantum mechanical algorithm for database search. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. [S.l.: s.n.], 1996. p. 212–219.

GUPTA, H.; VARSHNEY, H.; SHARMA, T. K.; PACHAURI, N.; VERMA, O. P. Comparative performance analysis of quantum machine learning with deep learning for diabetes prediction. *Complex & Intelligent Systems*, Springer, p. 1–15, 2021.

HAN, K.-H.; KIM, J.-H. Genetic quantum algorithm and its application to combinatorial optimization problem. In: IEEE. *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512)*. [S.l.], 2000. v. 2, p. 1354–1360.

HARTZBAND, D. *Information Technology and Data in Healthcare: Using and Understanding Data*. [S.l.]: CRC Press, 2019.

HEALTHCARE IT News, Cleveland Clinic, IBM launch 10-year quantum computing partnership. 2021. <https://www.healthcareitnews.com/news/cleveland-clinic-ibm-launch-10-year-quantum-computing-partnership>. Accessed: 2021-04-05.

JONES, J. A.; MOSCA, M.; HANSEN, R. H. Implementation of a quantum search algorithm on a quantum computer. *Nature*, Nature Publishing Group, v. 393, n. 6683, p. 344–346, 1998.

JOZSA, R.; LINDEN, N. On the role of entanglement in quantum-computational speed-up. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, v. 459, n. 2036, p. 2011–2032, 2003.

KRISTENSEN, L. B.; DEGROOTE, M.; WITTEK, P.; ASPURU-GUZIK, A.; ZINNER, N. T. *An Artificial Spiking Quantum Neuron.* 2020.

LAMATA, L. Quantum machine learning and quantum biomimetics: A perspective. *Machine Learning: Science and Technology*, IOP Publishing, v. 1, n. 3, p. 033002, 2020.

MANGINI, S.; TACCHINO, F.; GERACE, D.; MACCHIAVELLO, C.; BAJONI, D. Quantum computing model of an artificial neuron with continuously valued input data. *Machine Learning: Science and Technology*, IOP Publishing, v. 1, n. 4, p. 045008, Oct 2020. ISSN 2632-2153. Available at: <http://dx.doi.org/10.1088/2632-2153/abaf98>.

MANGINI, S.; TACCHINO, F.; GERACE, D.; BAJONI, D.; MACCHIAVELLO, C. Quantum computing models for artificial neural networks. *EPL (Europhysics Letters)*, IOP Publishing, v. 134, n. 1, p. 10002, 2021.

MINSKY, M.; PAPERT, S. An introduction to computational geometry. *Cambridge tiass., HIT*, 1969.

MONTEIRO, C. A.; FILHO, G. I.; COSTA, M. H. J.; NETO, F. M. de P.; OLIVEIRA, W. R. de. Quantum neuron with real weights. *Neural Networks*, 2021. ISSN 0893-6080. Available at: <https://www.sciencedirect.com/science/article/pii/S0893608021003051>.

MöTTöNEN, M.; VARTIAINEN, J.; BERGHOLM, V.; SALOMAA, M. Transformation of quantum states using uniformly controlled rotations. *Quantum Information & Computation*, v. 5, p. 467–473, 09 2005.

NETO, F. M. de P.; FILHO, I. G.; MONTEIRO, C. A. Approaches to avoid overfitting in a quantum perceptron. In: IEEE. *2020 International Joint Conference on Neural Networks (IJCNN).* [S.l.], 2020. p. 1–8.

NETO, F. M. de P.; LUDERMIR, T. B.; OLIVEIRA, W. R. de; SILVA, A. J. da. Implementing any nonlinear quantum neuron. *IEEE Transactions on Neural Networks and Learning Systems*, p. 1–6, 2019. ISSN 2162-2388.

NIELSEN, M. A.; CHUANG, I. L. *Quantum Computation and Quantum Information.* [S.l.]: Cambridge University Press, 2000.

OLIVEIRA, W. de. Quantum ram based neural netoworks. In: *European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning.* [S.l.: s.n.], 2009. p. 331–336.

ORUS, R.; MUGEL, S.; LIZASO, E. Quantum computing for finance: Overview and prospects. *Reviews in Physics*, v. 4, p. 100028, 02 2019.

PANELLA, M.; MARTINELLI, G. Neural networks with quantum architecture and quantum learning. *International Journal of Circuit Theory and Applications*, John Wiley & Sons, Ltd., v. 39, n. 1, p. 61–77, 2011. ISSN 1097-007X.

PRESKILL, J. Quantum computing in the nisq era and beyond. *Quantum*, Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften, v. 2, p. 79, 2018.

RAGHUPATHI, W.; RAGHUPATHI, V. Big data analytics in healthcare: promise and potential. *Health information science and systems*, Springer, v. 2, n. 1, p. 1–10, 2014.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, p. 65–386, 1958.

SCHULD, M.; SINAYSKIY, I.; PETRUCCIONE, F. The quest for a quantum neural network. *Quantum Information Processing*, v. 13, 08 2014.

SCHULD, M.; SINAYSKIY, I.; PETRUCCIONE, F. Simulating a perceptron on a quantum computer. *Physics Letters A*, Elsevier BV, v. 379, n. 7, p. 660–663, Mar 2015. ISSN 0375-9601. Available at: <http://dx.doi.org/10.1016/j.physleta.2014.11.061>.

SHAIKH, T. A.; ALI, R. Quantum computing in big data analytics: A survey. In: *2016 IEEE International Conference on Computer and Information Technology (CIT)*. [S.l.: s.n.], 2016. p. 112–115.

SHENVI, N.; KEMPE, J.; WHALEY, K. B. Quantum random-walk search algorithm. *Physical Review A*, APS, v. 67, n. 5, p. 052307, 2003.

SIEGELMANN, H. T. Computation beyond the turing limit. *Science*, American Association for the Advancement of Science, v. 268, n. 5210, p. 545–548, 1995.

SIEGELMANN, H. T.; SONTAG, E. D. Turing computability with neural nets. *Applied Mathematics Letters*, Elsevier, v. 4, n. 6, p. 77–80, 1991.

SILVA, A. J. da; OLIVEIRA, W. R. de; LUDERMIR, T. B. Comments on "quantum m-p neural network". *International Journal of Theoretical Physics*, v. 54, n. 6, p. 1878–1881, 2015.

SMITH, J. W.; EVERHART, J. E.; DICKSON, W.; KNOWLER, W. C.; JOHANNES, R. S. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In: AMERICAN MEDICAL INFORMATICS ASSOCIATION. *Proceedings of the annual symposium on computer application in medical care*. [S.l.], 1988. p. 261.

SOLENOV, D.; BRIELER, J.; SCHERRER, J. F. The potential of quantum computing and machine learning to advance clinical research and change the practice of medicine. *Missouri medicine*, Missouri State Medical Association, v. 115, n. 5, p. 463, 2018.

TACCHINO, F.; BARKOUTSOS, P.; MACCHIAVELLO, C.; TAVERNELLI, I.; GERACE, D.; BAJONI, D. Quantum implementation of an artificial feed-forward neural network. *Quantum Science and Technology*, IOP Publishing, v. 5, n. 4, p. 044010, 2020.

TACCHINO, F.; MACCHIAVELLO, C.; GERACE, D.; BAJONI, D. An artificial neuron implemented on an actual quantum processor. *npj Quantum Information*, v. 5, p. 1–8, 2019.

TORRONTEGUI, E.; GARCíA-RIPOLL, J. J. Unitary quantum perceptron as efficient universal approximator. *EPL (Europhysics Letters)*, IOP Publishing, v. 125, n. 3, p. 30004, Mar 2019. ISSN 1286-4854. Available at: <http://dx.doi.org/10.1209/0295-5075/125/30004>.

UN, G. A. Constitution of the world health organization. *American Journal of Public Health and the Nations Health*, v. 36, n. 11, p. 1315–1323, 1946. PMID: 18016450. Available at: <https://doi.org/10.2105/AJPH.36.11.1315>.

VALLE, M. E.; LOBO, R. A. Hypercomplex-valued recurrent correlation neural networks. *Neurocomputing*, Elsevier BV, v. 432, p. 111–123, Apr 2021. ISSN 0925-2312. Available at: <http://dx.doi.org/10.1016/j.neucom.2020.12.034>.

VERSTRAETE, F.; WOLF, M. M.; CIRAC, J. I. *Quantum computation, quantum state engineering, and quantum phase transitions driven by dissipation.* 2009.

WIDROW, B.; HOFF, M. E. *Adaptive switching circuits.* [S.l.], 1960.

WIEBE, N.; KAPOOR, A.; SVORE, K. M. Quantum nearest-neighbor algorithms for machine learning. *Quantum Information and Computation*, Rinton Press, v. 15, n. 3-4, p. 318–358, March 2015. Available at: <https://www.microsoft.com/en-us/research/publication/quantum-nearest-neighbor-algorithms-for-machine-learning/>.

WIEBE, N.; KAPOOR, A.; SVORE, K. M. Quantum perceptron models. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems.* Red Hook, NY, USA: Curran Associates Inc., 2016. (NIPS'16), p. 4006–4014. ISBN 9781510838819.

YAN, S.; QI, H.; CUI, W. Nonlinear quantum neuron: A fundamental building block for quantum neural networks. *Physical Review A*, APS, v. 102, n. 5, p. 052421, 2020.

YANOFSKY, N. S.; MANNUCCI, M. A. *Quantum Computing for Computer Scientists.* 1. ed. USA: Cambridge University Press, 2008. ISBN 0521879965.

ZHANG, X.; XIANG, H.; XIANG, T.; FU, L.; SANG, J. *An efficient quantum circuits optimizing scheme compared with QISKit.* 2018.

ZHANG, Y.; DENG, H.; LI, Q.; SONG, H.; NIE, L. Optimizing quantum programs against decoherence: Delaying qubits into quantum superposition. In: *2019 International Symposium on Theoretical Aspects of Software Engineering (TASE).* [S.l.: s.n.], 2019. p. 184–191.

ZHOU, R.; DING, Q. Quantum m-p neural network. *International Journal of Theoretical Physics*, Springer US, v. 46, n. 12, p. 3209–3215, 2007. ISSN 0020-7748.