Quantum One-class Classification With a Distance-based Classifier*

Nicolas M. de Oliveira[1], Teresa B. Ludermir[1], Wilson R. de Oliveira[2], and Adenilton J. da Silva[1]

[1] Centro de Informática, Universidade Federal de Pernambuco, Recife, Brazil
{nmo, tbl, ajsilva}@cin.ufpe.br
[2] Departamento de Estatística e Informática, Universidade Federal Rural de Pernambuco, Recife, Brazil
wilson.rosa@gmail.com

Abstract. Distance-based Quantum Classifier (DBQC) is a quantum machine learning model for pattern recognition. However, DBQC has a low accuracy on real noisy quantum processors. We present a modification of DBQC named Quantum One-class Classifier (QOCC) to improve accuracy on NISQ (Noisy Intermediate-Scale Quantum) computers. Experimental results were obtained by running the proposed classifier on a computer provided by IBM Quantum Experience and show that QOCC has improved accuracy over DBQC.

Keywords: Quantum Machine Learning · Quantum Computing · Pattern Classification

1 Introduction

In Quantum Computing (QC) [7] we find algorithms which can solve particular problems more efficiently than any known corresponding classical algorithms. Examples of this quantum speedup are Shor's algorithm and Grover's algorithm. Characteristics such as quantum parallelism and other phenomena only observed in quantum mechanics increased research interest in the application of QC to problems without efficient algorithmic solutions.

Several works propose Quantum Machine Learning (QML) [1] models and quantum supervised learning problems. For instance, quantum support vector machine [10]; learning algorithm for a perceptron through quantum amplitude amplification [4]; and a supervised pattern recognition algorithm that implements a linear regression model in quantum computers [13].

We investigate the Distance-based Quantum Classifier (DBQC) [11] and present a modification to this classifier named Quantum One-Class Classifier QOCC to allow its implementation on NISQ (Noisy Intermediate-Scale Quantum) computers [9]. QOCC is based on DBQC and *Probabilistic Quantum Mem-*

* Submitted for publication in November 2019. This work is supported by research grants from CNPq, CAPES and FACEPE (Brazilian research agencies).

ory (PQM) [14]. This work aims to improve the DBQC accuracy on NISQ computers removing the class register to avoid the impossibility of classifying samples of class 2 (class qubit $|1\rangle$) due to its rapid decoherence [11]. We perform experiments on IBM Quantum Experience [3] to validate our classifier, the first experiment in an error-free simulation environment, and then on the non-error-corrected quantum processor *ibmq_vigo*. The original DBQC tries to define a minimal quantum classifier. In this work we reduce the size of the classifier with improvements in the accuracy on quantum devices.

The remainder of this paper is divided into 6 sections. Section 2 summarizes the basic principles of quantum computing. Section 3 shows some examples of quantum machine learning. Section 4 describes the original distance-based quantum classifier that motivated our proposal. Section 5 presents the main results of this work: a description of our quantum one-class classifier. Section 6 gives details of the experiments performed as a proof of concept in prototypes of quantum devices. The proposed method has higher accuracy in NISQ devices. Finally, Section 7 is the conclusion.

2 Quantum Computing

In this section, a brief review of essential concepts for a basic understanding of quantum computing will be exposed. For a complete and more detailed approach we suggest the text contained in [7].

A quantum computer is a machine capable of performing computational calculations and operations based on properties and phenomena of quantum mechanics. Analogous to the classic bit, the unit of quantum information is the quantum bit, or *qubit*. To a qubit the logical values "0", "1", or any superposition of these can be assigned. This superposition consists of a linear combination of the states of the computational base described by a vector like in (1), where α and β are probabilistic amplitudes associated with the respective states and $|\alpha|^2 + |\beta|^2 = 1$.

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

As can also be seen in (1) one of the main characteristics of quantum computing compared to classical computing is the superposition of states. This superposition allows quantum computing to obtain a high degree of computational parallelism, described in general in (2), where we have n qubits and α_i probabilistic amplitudes associated with i states. Thus, n qubits can represent 2^n combinations of states.

$$|\psi\rangle := \sum_{i=0}^{2^n-1} \alpha_i|i\rangle \tag{2}$$

Operations under a quantum state are performed by unit operators. Given a $U : \mathcal{H} \rightarrow \mathcal{H}$ operator, with $\mathcal{H}$ denoting Hilbert's space, U is said to be unitary when its inverse is equal to its conjugate transpose. That is, $UU^\dagger = U^\dagger U = I$, where I designates the identity operator. Thus, an operator acting on n qubits needs to be in a complex Hilbert space of dimension 2^n. That is, a single application of U performs an exponential number of operations in the states of the base (3).

$$U|\psi\rangle = U\left(\sum_{i=0}^{2^n-1} \alpha_i |i\rangle\right) = \sum_{i=0}^{2^n-1} \alpha_i U|i\rangle \tag{3}$$

A constraint that is applied to the QC computing model refers to the *measurement* required to extract any information resulting from a qubit state. When performing this operation, the state collapses and only one of the states that were in superposition will be observed. Given the state in (2), after the measurement is performed, the probability of obtaining the output $|i\rangle$ is $p_i = |\alpha_i|^2$.

Quantum circuits are one of the ways available for performing quantum computing. It is the quantum circuits that determine which and in what order the operators are applied to one or more qubits. Examples of quantum operators/gates and their respective actions are: *Identity* (4), *Not* (5), *Hadamard* (6) and *Controlled-not (CNOT)* (2).

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \begin{matrix} I|0\rangle = |0\rangle \\ I|1\rangle = |1\rangle \end{matrix} \tag{4}$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \begin{matrix} X|0\rangle = |1\rangle \\ X|1\rangle = |0\rangle \end{matrix} \tag{5}$$

$$H = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}, \qquad \begin{matrix} H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{matrix} \tag{6}$$

$$\begin{gathered} CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\ CNOT|0\rangle|0\rangle = |0\rangle|0\rangle \; CNOT|1\rangle|0\rangle = |1\rangle|1\rangle \\ CNOT|0\rangle|1\rangle = |0\rangle|1\rangle \; CNOT|1\rangle|1\rangle = |1\rangle|0\rangle \end{gathered} \tag{7}$$

In addition to these, there is a generalization of the CNOT quantum gate, which can include more than one qubit having the control function $(0, ..., i)$ and more than one qubit as target $(0, ..., j)$, in addition to being able to apply an arbitrary U operator to the target qubits. The general controlled gate is represented in Figure 1.

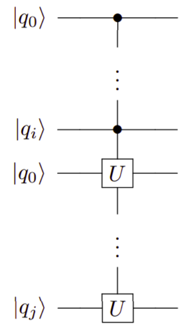

Fig. 1. Representation in the quantum circuit of a general controlled gate.

3 Quantum Machine Learning

Quantum machine learning is an area of quantum computing that combines artificial intelligence techniques with the power of quantum computing. In general, this integration aims to solve pattern recognition problems where the objective is to learn about new data from other data already analyzed. For a summary of some concepts related to QML we suggest the text contained in [1]. A more comprehensive text on QML, including information encoding and other techniques for performing machine learning on quantum computers can be found in [12].

As stated in Section 1, there are several approaches that employ the concept of quantum machine learning to perform classification tasks. In [10], the authors represented the Support Vector Machine (SVM) algorithm as an approximate least-squares problem to take advantage of the matrix inversion algorithm under the number of training vectors M. In addition, the presented algorithm uses fast quantum evaluation of inner products to obtain $O(\log NM)$ run time performance, where N is the dimension of the feature space.

The work presented in [4] develops two quantum algorithms for perceptron learning. The first one makes use of Grovers search to quadratically speed up online perceptron training. This algorithm works from a quantum analog of the traditional training of a perceptron to define a separating hyperplane with $O(\sqrt{N})$ steps, for N data points. The second one consists of a quantum version space training algorithm which improves its classical counterpart through amplitude amplification. Such an algorithm also made use of Grover's search and improves the classical mistake bound of $O\left(\frac{1}{\gamma^2}\right)$ to $O\left(\frac{1}{\sqrt{\gamma}}\right)$, where γ indicates the margin between the classes in the training data.

In [13], the authors use quantum computing to develop an algorithm to predict the class output to a new input from a dataset. This algorithm is based on a linear regression model with least squares optimisation and it runs in time logarithmic ($\mathcal{O}(\log N\kappa^2\epsilon^{-3})$, where N is the dimension of the input data, κ is its condition number and ϵ is the desired accuracy) regarding the size of the input space. To achieve this result, the authors used techniques of quantum Principal

Component Analyzes [5] and focused on guessing the output of a new input instead of facing the problem of reading out the optimal parameters of the fit.

In this paper we will focus on a machine learning quantum model that uses quantum interference to classify new input data [11]. Such interference occurs through the *Hadamard gate* shown in Section 2 which, even when applied to a single qubit, acts in all amplitudes present in a quantum state.

4 Distance-based Quantum Classifier

The Distance-based Quantum Classifier (DBQC) proposed by [11] aims to investigate how to perform a distance-based classification task from the construction of a quantum circuit. The strategy used in DQBC is to use *amplitude encoding* to encode the input features and to use quantum interference to evaluate the distance from a new input vector to the training set vectors. Such a construction aims to provide a classification model that can be implemented in existing quantum devices. To validate the DBQC, the authors performed supervised classification experiments using the Iris dataset [2] (available in Scikit-learn [8]). The quantum system that performs the classification is shown in (8). Where $|m\rangle$ is an index register flagging the mth training vector, $|\psi_{\mathbf{x}^m}\rangle$ is the mth training vector, $|\psi_{\tilde{\mathbf{x}}}\rangle$ is the new input, and $|y^m\rangle$ is a single qubit that stores the vector class.

$$|\mathcal{D}\rangle = \frac{1}{\sqrt{2M}} \sum_{m=1}^{M} |m\rangle \Big(|0\rangle |\psi_{\tilde{\mathbf{x}}}\rangle + |1\rangle |\psi_{\mathbf{x}^m}\rangle \Big) |y^m\rangle \tag{8}$$

The main point of DBQC is a conditional measurement, called *postselection*, that depends on the probability of measuring $|0\rangle$ in the *ancilla qubit*. A measurement is made in the *class qubit* only after the postselection succeeds. Experimental results showed 100% accuracy for classes 1 and 2 of the Iris dataset. However, it was impossible in [11] to classify new input vectors from class 2 on NISQ computers due to the rapid decoherence of the *class qubit* storing $|1\rangle$.

5 Quantum One-class Classifier

The postselection of the DBQC [11] succeeds with probability $p_{acc} = \frac{1}{4M} \sum_m |\tilde{\mathbf{x}} + \mathbf{x}^m|^2$. This probability depends on data distribution and can tend to zero. Figure 2 presents an artificial dataset where the postselection probability is approximately 0.02 for the pattern $\mathbf{x}_0$ and 0.98 for pattern $\mathbf{x}_1$. The postselection probability is a function of the Euclidean distance of the new pattern to the patterns in the dataset and returns 0 with higher probability if the new pattern is near to the patterns in the dataset. In this Section, we redefine the DBQC to use the outcome of the postselection as the output of the classifier. With this modification, we remove the *class qubit*, reduce the number of repetitions necessary to estimate the output of the classifier and reduce the number of operations necessary to perform the classification.

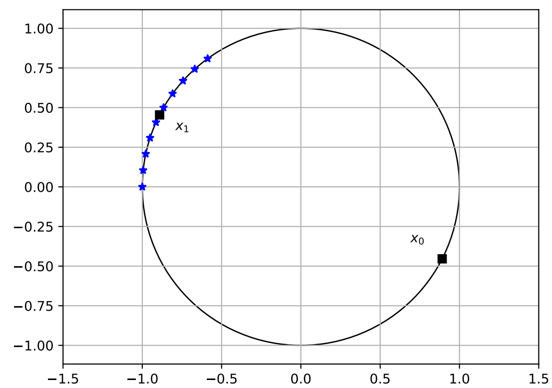

Fig. 2. Artificial example where the probability of postselection tends to zero for a given new input vector.

The QOCC is based on the DBQC and extends its applicability to allow the classification of a new input vector from a quantum one-class classifier indicating the probability that the vector will be associated with the set of loaded (training) vectors in the classifier. Such a modification allows us to work around the problem of rapid decoherence pointed out by [11] and correctly classify samples of class 2 (class qubit $|1\rangle$) on real NISQ computers.

In our approach we use the same data preprocessing strategy used in [11]. As with the original classifier, we use the quantum rotation gate R_y (9) to load training vectors and new input vectors. However, the classification of a new input vector works similar to a probabilistic quantum memory [14]. Thus, a new input vector is classified according to a *degree of membership* of this new vector against vectors already stored in the classifier. This degree of membership actually is the probability of measuring 0 on the output where previously was postselection.

$$R_y(\theta) = \begin{pmatrix} cos\left(\frac{\theta}{2}\right) & -sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{9}$$

Thus, the modification made here allows us to abstract the issue of data distribution, which could lead to a very low probability of postselection. Also, by loading training samples from a single class, our classifier proves to be more flexible by simplifying the classification/association of a new input vector in any class.

Figure 3 shows our classifier for a given class C. In step F the computed output of successive measurements represents the degree of membership of the new input vector to the C class. Therefore, a degree of membership greater than 0.5 means that the new input vector has been classified as class C. The procedure for performing QOCC is shown in Algorithm 1.

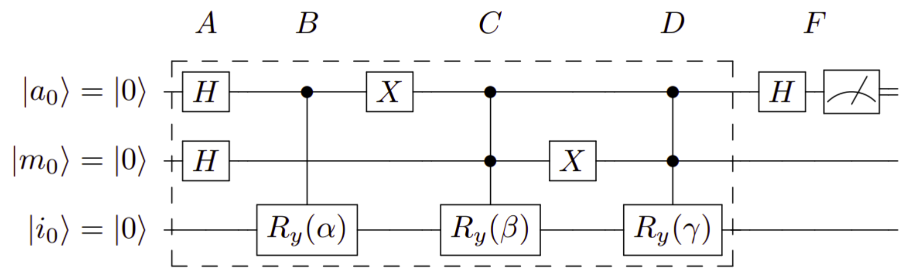

Fig. 3. Quantum circuit implementing quantum one-class classifier. The result of successive runs of this circuit represents a kind of degree of membership of the new input vector (step B) to the vectors already stored in the classifier (steps C and D). The quantum gates R_y are responsible for loading data from each vector through their associated angles α, β and γ. Enclosed the dashed lines is the subcircuit for state preparation. At the end of the computation, the Hadamard gate in the ancilla qubit interferes the copies of the new input vector with the loaded vectors and then the ancilla is measured.

Algorithm 1: Quantum One-class Classifier (QOCC)

Input: *test*, *training*
Initialize quantum registers $ancilla = |a\rangle$, $index = |m\rangle$, $data = |i\rangle$
Perform $H|a\rangle$ and $H|m\rangle$
Perform C-$R_y(test)|a\rangle|i\rangle$ to load the sample to be classified
Apply $X|ancilla\rangle$ to entangle test sample with the ground state of the ancilla
Perform CC-$R_y(training[0])|a\rangle|m\rangle|i\rangle$ to load the first training sample
Apply $X|m\rangle$ to entangle the first training sample with the ground state of the index and the excited state of the ancilla
Perform CC-$R_y(training[1])|a\rangle|m\rangle|i\rangle$ to load the second training sample
Apply $H|a\rangle$ to interferes the copies of the test sample with the training ones
Measure $|a\rangle$ to get the probability of the output being $|0\rangle$

In Figure 3 the rotation gates R_y in steps B, C and D load the classical input vectors in quantum amplitudes. However, this is not so straightforward and requires an auxiliary procedure to perform such an embedding, to be explained in the next section.

5.1 Amplitude Encoding

In order to perform the amplitude encoding necessary to encode input data in the amplitudes of the qubits, we follow the strategy present in [12]. Figure 4 illustrates how to start from the desired vector to the state $|0...0\rangle$.

In Figure 4 each β is a rotation angle to be performed by the respective R_y gate and is calculated according to (10), where $s = \{1, ..., n\}$, n is the number of qubits and $j = \left\{1, ..., 2^{i-1}\right\}$ (i is the index of the qubit in which the rotation gate is being applied).

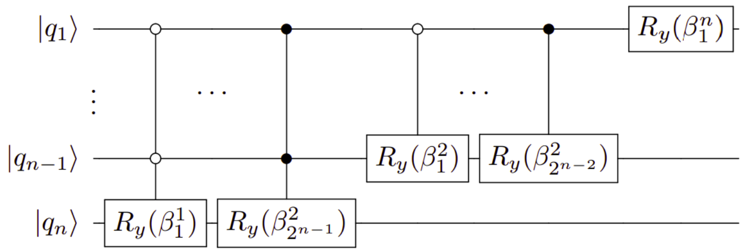

Fig. 4. Procedure to reach the vector $|0...0\rangle$ from any other vector.

$$\beta_j^s = 2 \text{ arcsin} \left(\frac{\sqrt{\sum_{l=1}^{2^{s-1}} |\alpha_{(2j-1)2^{s-1}+l}|^2}}{\sqrt{\sum_{l=1}^{2^s} |\alpha_{(j-1)2^s+l}|^2}} \right) \tag{10}$$

The main idea is to perform different rotations for each portion of the superposition state through multi-controlled rotation gates. Amplitude coding is performed by applying the inverse gates shown in Figure 4 in the inverse order. As an example of how amplitude encoding works, Figure 5 shows the procedure being performed to load the 4 feature input vector $|\psi\rangle = -0.286|00\rangle + 0.723|01\rangle - 0.464|10\rangle - 0.425|11\rangle$ (Iris sample 20) in a quantum circuit.

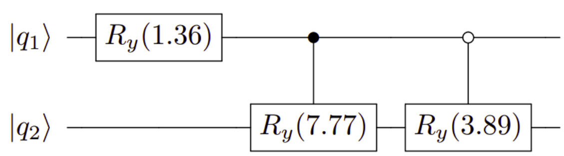

Fig. 5. Example of how to load a sample in the amplitudes of a quantum state.

6 Experiments and Results

Here we will show the results obtained with the execution of the classifier proposed in the previous section. Will be considered the simulation and experimental results. As in [11], we use the Iris dataset [2] in our experiments. Just like in DBQC, an adequate choice of training vectors strongly influences the probability of classification success. However, for all experiments, we choose to take 1 or 2 random samples, depending on the experiment.

Ideally, we should load a larger training set, but due to the limitations and errors of the available quantum hardware we decided to keep the original proposal and load only 2 training vector for the classifier. For the case where it was necessary to load one vector of each class, samples 20 (class 1) and 97 (class

2) were used. For situations where it was necessary to load two samples of each class we used the samples $\{21, 30\}$ (class 1) and $\{83, 89\}$ (class 2). The execution of the QOCC on IBM Quantum Experience [3] followed the default (1024 runs) for each experiment/sample.

As already mentioned in Section 5 the QOCC loads only the input vectors of a single class. Thus, in our results, where QOCC-X is shown, X represents the classes of the input vectors. Still, the accuracy takes into account both the correct classification of the samples of class X and the samples of the other class (which must be identified by the classifier with a low degree of membership).

Therefore, in order to validate our approach, we replicate the DBQC and compare with our QOCC (Table 1). We emphasize that the experiments were performed for classes and features 1 and 2 of the Iris dataset. All experiments were performed both on the simulator and on a real quantum computer provides by IBM Quantum Experience. The experiments for our classifier followed the Algorithm 2.

Algorithm 2: Experiments for Quantum One-class Classification

Load Iris dataset $\mathcal{D}$ (class = $\{0, 1\}$, features = $\{0, 1\}$)
Standardize and normalize $\mathcal{D}$
Pick 2 random samples $\mathbf{x}^0$ and $\mathbf{x}^1$ from dataset
Set training set $\mathcal{T} = (\mathbf{x}^0, y^0), (\mathbf{x}^1, y^1)$
Calculate training angles $\mathcal{T}_{angles}$ for R_y gates
for $x = 1$ *to size($\mathcal{D}$)* **do**
    if x *not in* $\mathcal{T}$ **then**
        Calculate test angle x_{angle} for R_y gate
        Perform QOCC(x_{angle}, $\mathcal{T}_{angles}$)
    end
end
Return the number of samples that were correctly associated with their classification memory

6.1 Discussion

The accuracy presented in [11] is only achieved if postselection succeeds.

The results of the previous section consider samples the went through the postselection and reached the classification step. We consider that a probability of postselection success greater than 0.5 means that postselection succeeds. In the experiments performed, postselection succeeds in 48% of the executions on the simulation and 71% of the execution on real quantum computer. Finally, the impossibility to classify input vectors of class 2 due to the rapid decoherence of the class qubit is solved with our QOC classifier, since it does not depend on a class qubit storing $|1\rangle$ to indicate input vectors of class 2.

Table 1. Accuracy of QOCC and DBQC for Classes and Features 1 & 2 of the Iris Dataset

Classifier	**Overall accuracy**	**C_1 accuracy**	**C_2 accuracy**
DBQC (simul.)	100%	100%	100%
DBQC (IBM Q)	53%	74%	26%
QOCC-1 (simul.)	100%	100%	100%
QOCC-1 (IBM Q)	92%	96%	88%
QOCC-2 (simul.)	98%	98%	98%
QOCC-2 (IBM Q)	89%	82%	96%

In order for our classifier to handle datasets containing more than 2 classes, it was observed that it is necessary to perform a more adequate pre-processing of the input vectors. This pre-processing is essential for our classifier to be able to efficiently distinguish among all available classes. Another possibility to overcome this obstacle is to parameterize the quantum circuit to evaluate its performance in the classification by inserting variable rotation gates, based on the theory of variational circuits presented in [6].

In the model presented by [11], it is not clear how to proceed in the presence of more features for the input vector. This is because the exposed DBQC was prepared to deal with a single qubit of data storing the amplitudes corresponding to the features of the samples, not being explicit as to generalize the operations involving the qubit that stores the class. For our classifier, it is possible to directly apply the procedure shown in [12] (see Section 5.1).

Finally, on datasets with only two classes the QOCC containing vectors from a single class works as a complete quantum binary classifier. Therefore, it is possible to classify the entire dataset only with respect to the samples of a single class stored in the classifier, so that the high (low) degree of membership of the test samples to those already stored gives us the classification between the classes 1 or 2.

7 Conclusion

In this work, we investigated a distance-based quantum classifier (DBQC), and we proposed a new one (QOCC) based on the idea of probabilistic quantum memory for a single class that performs better on NISQ computers. The QOCC has better accuracy on real quantum processors when compared with the DBQC.

Our QOCC shows an advantage regarding the DBQC with regard to its size, the accuracy of the class 1 classification and the absence of the post-selection stage. In addition, in datasets with two classes, the QOCC behaves like a complete quantum binary classifier, indicating in the ancilla qubit the degree of membership of the input vector to class 0 or 1, just needing to load data from one of the classes.

There are some possible improvements to be explored. Possible future works are to investigate how to use the QOCC to classify multiclass datasets and to explore how to insert parameters into the classifier to improve its accuracy. Also, it is possible to conduct a circuit optimization study for more efficient execution on real quantum computers.

8 Code Availability

The codes used during this study are available on the GitHub repository.

References

1. Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., Lloyd, S.: Quantum machine learning. Nature **549**(7671), 195–202 (2017)
2. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of eugenics **7**(2), 179–188 (1936)
3. IBM Quantum Experience (2020), `https://www.ibm.com/quantum-computing/technology/experience/`, Last accessed: 2020-05-13.
4. Kapoor, A., Wiebe, N., Svore, K.: Quantum perceptron models. In: Advances in Neural Information Processing Systems. pp. 3999–4007 (2016)
5. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum principal component analysis. Nature Physics **10**(9), 631–633 (2014)
6. McClean, J.R., Romero, J., Babbush, R., Aspuru-Guzik, A.: The theory of variational hybrid quantum-classical algorithms. New Journal of Physics **18**(2), 023023 (2016)
7. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press (2010)
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
9. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum **2**, 79 (Aug 2018). https://doi.org/10.22331/q-2018-08-06-79, `https://doi.org/10.22331/q-2018-08-06-79`
10. Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum support vector machine for big data classification. Physical review letters **113**(13), 130503 (2014)
11. Schuld, M., Fingerhuth, M., Petruccione, F.: Implementing a distance-based classifier with a quantum interference circuit. EPL **119**(6), 60002 (2017). https://doi.org/10.1209/0295-5075/119/60002, `https://doi.org/10.1209/0295-5075/119/60002`
12. Schuld, M., Petruccione, F.: Supervised learning with quantum computers, vol. 17. Springer (2018)
13. Schuld, M., Sinayskiy, I., Petruccione, F.: Prediction by linear regression on a quantum computer. Physical Review A **94**(2), 022342 (2016)
14. Trugenberger, C.A.: Probabilistic quantum memories. Physical Review Letters **87**(6), 067901 (2001)