CEUR-WS.org/Vol-3077/paper06.pdf

Classification problem solving using quantum machine learning mechanisms

Alina O. Savchuk[1], Nonna N. Shapovalova[1]

[1]Kryvyi Rih National University, 11 Vitalii Matusevych Str., Kryvyi Rih, 50027, Ukraine

Abstract

Due to the similarity of the computational and probabilistic nature of quantum computing and machine learning, the idea arose to optimize the learning process using quantum methods. There are both fundamentally new algorithms, such as HHL, and quantum-improved ones: QPCA, QSVM. In this article, we will look at the QSVM algorithm step by step, starting from the basics described in section 2 and gradually delving into the composition of the algorithm. Thus, after the basics, we will consider the quantum phase estimation, which is part of the HHL algorithm, and then the QSVM algorithm, the component of which is the HHL. We also will consider the QPCA algorithm, which can be applied before the QSVM algorithm to reduce the dimension of the data sample. In this way, we explore the fundamental difference between classical algorithms and their quantum counterparts. We also implement the QSVM method in practice and compare the obtained practical results with theory. As a result, we obtained high quality of accuracy (100 %) compared to the classical SVM (83 %) on a data sample of dimension 72. However, we found out that the learning time on a quantum device is far from ideal (it can reach 5 min for a sample of this size). The study aims to theoretically argue or disprove the hypothesis about the efficiency of quantum computing for machine learning algorithms. The object of research is the programming of quantum computers. The research subject is the study of quantum computing mechanisms for the implementation of machine learning problems. The research result is a software module that allows evaluating the efficiency of the classification task on a quantum computer. It also can be used to compare the results obtained from classical and quantum devices. Research methods: theoretical analysis of the foundations of quantum computing: principles of superposition and entanglement, linear algebra, probability theory over complex numbers; building a model of one qubit and multi-qubit system; research of quantum machine learning algorithms' work principles and their complexity; empirical comparison of quantum machine learning methods with their classical counterparts.

Keywords

quantum machine learning, classification problem, HHL algorithm, quantum principal component analysis, quantum support vector machine

1. Introduction

While building a quantum computer is still a daunting engineering challenge, it already has a long list of areas to be effective [1, 2]:

- data modeling (weather forecast, chemical modeling);

CS&SE@SW 2021: 4th Workshop for Young Scientists in Computer Science & Software Engineering, December 18, 2021, Kryvyi Rih, Ukraine

stereoalina@gmail.com (A. O. Savchuk); shapovalovann09@gmail.com (N. N. Shapovalova)
http://mpz.knu.edu.ua/pro-kafedru/vikladachi/shapovalova-n-n (N. N. Shapovalova)
0000-0003-4335-3714 (A. O. Savchuk); 0000-0001-9146-1205 (N. N. Shapovalova)

© 2022 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

- creation of drugs (search for new formulas for drugs by studying the spatial structure of protein compounds, diagnostics, and testing of various types of diseases);
- logistics (combinatorial traveling salesman problem);
- search optimization (Google uses quantum technologies to refine search results, taking into account more important ones);
- data security (quantum protocols for encryption and data transmission);
- military engineering (improving the accuracy of radar weapons);
- machine learning (more accurate solutions and faster learning algorithms).

Therefore, the field of quantum computing have met with increasing interest. Governments of countries such as China, the USA, Germany, Switzerland, Russia, and others are investing in quantum computing technologies development to take a leading position in the quantum race. Private companies, including Google, IBM, Microsoft, Intel, Amazon, are not lagging behind.

The number of qubits plays an influential role in quantum superiority achievement, but no less important, one might even say, the decisive factor is the "quantum volume" – a metric that determines the power of a quantum computer in conjunction with the level of computational errors. Today, Honeywell is the leader in these indicators, setting a record of 64 units of quantum volume [3].

However, quantum computers may be useless without appropriate scientific knowledge about their proper use. In this regard, the IBM company has a policy of encouraging interest in research in this area, providing hardware (IBM Q) and software (Qiskit) tools on a costless basis [4]. These tools allow anyone to conduct experiments on real-life quantum installations of the order of 1 to 5 qubits, with a quantum volume from 1 to 32 units.

There are the following technologies for constructing qubits, implemented in practice: quantum dots (Intel), superconducting elements (D-Wave, Google, IBM), vacuum traps (IonQ, AQT, Honeywell), vacancies in the diamond lattice (Quantum Diamond Technologies) [5]. The most popular approach is to build qubits on superconducting elements.

The biggest challenge for building any quantum computer is the problem of overcoming decoherence. Decoherence is the process of destroying the coherence of a coupled qubits system. This phenomenon is created by thermodynamic processes when interacting with the environment. Decoherence disrupts the communication of the system of qubits and destroys their quantum properties, which negates the advantage of a quantum computer over a classical one. This problem is usually solved by combining more physical qubits into a single logical one. This technique is called quantum error correction. Quantum error correction is well developed in theory but not achievable in practice because the noise level of physical qubits is still too high.

There are such fundamental problems associated with the possibility of applying quantum machine learning algorithms [6, 7, 8]:

- input problem;
- output problem;
- problem of calculation;
- the problem of comparative analysis.

Quantum computers are known to provide exponential acceleration in some cases. But data input and output operations are carried out by transferring information from the classical world to the quantum one and vice versa. These operations can be quite time-consuming and, most importantly, can completely negate the benefits of exponential problem solving. The data entry problem can be solved by creating a quantum equivalent memory (QRAM). This problem is another significant engineering problem.

The calculation problem is related to input and output problems. It is impossible to know in advance how many gates a quantum computer will need when working with classical devices. Although quantum computers can solve big tasks better than classical computers, it is still unknown how big a problem must be to gain a quantum advantage.

The problem with the comparative analysis is that it is sometimes difficult to argue that the quantum algorithm is better than all the known classical ones. This claim requires extensive comparative testing with modern heuristic methods. Establishing lower bounds for quantum algorithms would partially solve this problem.

Thus, it is necessary to develop software for quantum machine learning considering all the described technical and algorithmic problems.

2. Quantum computing basics

A quantum computer operates with minimal information units - qubits. A qubit can be represented as a unit vector of a 2-dimensional complex space:

$$|\phi\rangle \in H, \parallel \phi \parallel = 1, dim\ H = 2. \tag{1}$$

Here $|\phi\rangle$ is a column vector, H is a Hilbert space. A qubit can be 0 and 1, just like a bit:

$$|q_0\rangle = |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |q_0\rangle = |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2}$$

And also other values that represent the probability of occurrence of zero or one. Such probabilities are called probability amplitudes. In example (3) the qubit $|q_0\rangle$ can take the value of either 0 or 1 (the probability of meeting this or that event is $\frac{1}{2}$). In this case, they say that the qubit is in superposition.

$$|q_0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{bmatrix} \tag{3}$$

The sum of the probabilities should always be equal to one. For complex numbers, this means that the vector must be normalized. That is, for the vector:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \alpha, \beta \in \mathbb{C} \tag{4}$$

the following condition has to be met:

$$\sqrt{|\alpha|^2 + |\beta|^2} = 1 \tag{5}$$

To measure the state of a system with one qubit of information, one must first choose an orthonormal basis:

$$|x\rangle, |y\rangle : \langle x|y\rangle = 0, \|x\| = \|y\| = 1, \tag{6}$$

and then take measurements on this basis. Thus, the state $|x\rangle$ encountering probability when measuring the state $|\psi\rangle$ equal to the square of the modulus of the scalar product of the row vector $\langle x|$ and the column vector $|\psi\rangle$:

$$p(|x\rangle) = |\langle x|\psi\rangle|^2 \tag{7}$$

After measurement, the system from the state $|\psi\rangle$ can pass into the state $|x\rangle$ or the state $|y\rangle$ and does not change it after repeated measurement. Therefore, we can say that the measurement process converts quantum information into classical information. That is why measurements are most often at the very end of the algorithm.

Usually, we make measurements in the computational basis $|0\rangle, |1\rangle$. However, we can make measurements in other bases. For example, measuring the qubit from (3) in the Hadamard basis $|+\rangle, |-\rangle$, we obtain $|\langle +|q_0\rangle|^2 = 0$ and $|\langle -|q_0\rangle|^2 = 1$.

The state of a system consisting of more than one qubit can be represented as the tensor product of qubits. For example, the system state consisting of qubits $|a\rangle$ and $|b\rangle$ is described in (8). Thus, a system of n qubits is a vector with dimension 2^n. In this case, the number m, encoded by a vector in its pure form, reflects the state of the system, where the string $m + 1$ takes the value 1.

$$|ab\rangle = |a\rangle \otimes |b\rangle = \begin{bmatrix} a_0 \times \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \\ a_1 \times \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{bmatrix} \tag{8}$$

If two qubits are in a superposition state, the system will transition to one of four states during measurement:

$$(\alpha|0\rangle + \beta|1\rangle)(\gamma|0\rangle + \delta|1\rangle) = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle \tag{9}$$

In this case, the sum of the squares of all possible states is equal to 1:

$$|\alpha\gamma|^2 + |\alpha\delta|^2 + |\beta\gamma|^2 + |\beta\delta|^2 = 1 \tag{10}$$

However, not all multi-qubit systems can be decomposed into the tensor product of several separate states. For example, for the Bell state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, the measurement of the first qubit will result in the instantaneous measurement of the second: it will no longer be in a superposition state. If the measurement of one qubit leads to the measurement of the second, the state of such qubits is called entangled.

Based on formula (9), the system qubits independence can be verified by the equality of the product of the external and internal coefficients of the system. Thus, if equality is not met (as shown in (11)), then the system is confusing:

$$\alpha\gamma \times \beta\delta \neq \alpha\delta \times \beta\gamma \tag{11}$$

All operators performed on the state vector must be unitary. That is, for the operator U, its adjoint operator must be equal to the inverse:

$$UU^{*} = U^{*}U = I \tag{12}$$

Another property of unitary operators is that when they are applied, the scalar product of vectors (the angle between them) has preserved:

$$\forall \phi, \psi \in H |\langle U|\phi\rangle| \langle U|\psi\rangle| = |\langle \phi \psi \rangle| \tag{13}$$

In addition to this, the length of the vector has also preserved:

$$\forall \phi \parallel U|\phi\rangle \parallel = \parallel |\phi\rangle \parallel \tag{14}$$

We can also write the wave function from (5) in the following form:

$$|\psi\rangle = e^{i\gamma}(\cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle) \tag{15}$$

where $e^{i\gamma}$ is global phase, $e^{i\phi}$ is local phase. The global phase does not affect the calculation results because it is not observable.

Each operator has a finite set of eigenvalues and eigenphases. Eigenstate is a state that is immune to some operations. After applying the operator for such a state, the probability amplitude and the local phase will not change, but the global phase will change.Eigenphase is the global phase acquired by the eigenvalue. The global phase carries information about the applied operator. We can obtain this information by using a phase estimation.

3. Quantum phase estimation

Quantum phase estimation (QPE) is one of the commonly used building blocks of quantum machine learning algorithms. This algorithm helps to extract unobservable information from the global phase θ of the quantum operator U:

$$U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle \tag{16}$$

where $|\psi\rangle -$ is an eigenvector and $|e^{2\pi i\theta} -$ is an eigenvalue.

As we can see from (16), each eigenphase θ_j is associated with eigenstate u_j. Worth noticing the eigenphase θ_j is a fraction of 360° in the output quantum register. Looking ahead, the value of the output register R with size m will take the value:

$$R = \forall\theta_j 360 \times 2^m \tag{17}$$

The QPE algorithm writes the eigenphase of the U operator in the Hadamard basis. Then, using inverse QFT, we convert the value from the Hadamard basis to the computational basis. In other words, using the QPE algorithm, information is converted from the global phase to its own one. Then the information is decoded to produce the result with a READ operation. Algorithm description:

- system initialization where $|\psi\rangle$ has stored in one register and the value $2^n\theta$ in another register:
$$|\psi_0\rangle = |0\rangle^{\otimes n}|\psi\rangle; \tag{18}$$
- obtaining a superposition by applying the n-bit operation $H^{\otimes n}$ on the second register:
$$|\psi_1\rangle = \frac{1}{2^{\frac{n}{2}}}(|0\rangle + |1\rangle)^{\otimes n}|\psi\rangle; \tag{19}$$
- creates a supervised CU statement that applies only if the control bit is $|1\rangle$. Based on (15):
$$U^{2^j}|\psi\rangle = U^{2^{j-1}}|\psi\rangle = U^{2^{j-1}}e^{2\pi i\theta}|\psi\rangle = ... = e^{2\pi i 2^j\theta}. \tag{20}$$
Now it is necessary to apply controlled operations $CU^{2^{j-1}}$ in the range $0 \leq j \leq n-1$:
$$\begin{gathered}|\psi_2\rangle = \frac{1}{2^{\frac{n}{2}}}(|0\rangle + e^{2\pi i\theta 2^{n-1}} + |1\rangle) \otimes ... \otimes (|0\rangle + e^{2\pi i\theta 2^1} + |1\rangle)\otimes \\ \otimes(|0\rangle + e^{2\pi i\theta 2^0} + |1\rangle) \otimes |\psi\rangle = \frac{1}{2^{\frac{n}{2}}}\sum_{k=0}^{2^n-1} e^{2\pi i\theta k}|k\rangle) \otimes |\psi\rangle\end{gathered} \tag{21}$$
where k is an integer representation of n-bit numbers.
- perform inverse Fourier transform:
$$|\psi_3\rangle = \frac{1}{2^{\frac{n}{2}}}\sum_{k=0}^{2^n-1} e^{2\pi i\theta k}|k\rangle) \otimes |\psi\rangle \overset{QFT_n^{-1}}{\longrightarrow} \frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{-\frac{2\pi i k}{2n}(x-2^n\theta)}|x\rangle \otimes |\psi\rangle \tag{22}$$
- measure the system and get the answer in the auxiliary register:
$$|\psi_4\rangle = |2^n\theta\rangle \otimes |\psi\rangle \tag{23}$$

Thus, you can get reliable information about the global phase if $2^n\theta$ is an integer. Otherwise, reliable information will be received with a probability of about 40%. We can improve this probability by increasing the number of qubits. If you need to get your eigenphase by p bits of accuracy and the probability does not exceed e, you can determine the number of qubits m in the following way:
$$m = p + \lceil \log 2 + \frac{1}{\varepsilon} \rceil \tag{24}$$

The considered algorithm is irreplaceable for some QML algorithms. For example, the HHL algorithm uses QPE to obtain critical information about the matrix to be inverted.

4. Quantum machine learning algorithms

Systems of linear equations (SLE) in machine learning are the foundation of many algorithms. In general, a system of n equations is written as follows:

$$A|x\rangle = |b\rangle \tag{25}$$

To get a matrix form solution, it is necessary to find the inverse matrix A^{-1}:

$$|x\rangle = A^{-1}|b\rangle = \sum_{j=0}^{N-1} \lambda_j^{-1} b_j |u_j\rangle \tag{26}$$

where $|u_j\rangle$ is the j^{th} eigenvector of A and λ_j id eigenvalue.

Among the traditional algorithms for finding the solution vector x, the nonlinear conjugate gradient method can be distinguished as the most optimal algorithm. Matrix A. in this case, must be valid and meet the condition:

$$A = A^T > 0 \tag{27}$$

The HHL algorithm is used to solve SLN on a quantum computer. Since only unitary operators can be used in quantum algorithms, the matrix A must be Hermitian.

The HHL algorithm (figure 1) is effective in such cases:

- if one uses it as a structural element of a quantum machine learning algorithm;
- when it's necessary to check the equality of vectors $|x\rangle$ and $|y\rangle$;
- when one needs to find some derivative characteristic (sum, average, frequency component) of the vector $|x\rangle$.

Figure 1 shows an example of a quantum circuit that implements the HHL algorithm. Here are three quantum registers, initialized to $|0\rangle$. The n_l register stores the binary representation of the eigenvalues of A. The rest of the registers are used as service registers for storing intermediate calculations.

A step-by-step description of the algorithm [9]:

- we put $N = 2^{n_b}$
- loading the vector $|b\rangle$ through a transformation of the form $|0\rangle_{n_b} \mapsto |b\rangle_{n_b}$
- application of Quantum Phase Evaluation (QPE):

$$U = e^{iAt} = \sum_{j=0}^{N-1} e^{i\lambda_j t |u_j\rangle\langle u_i|} \tag{28}$$

The quantum state of the register, expressed in its own basis A:

$$\sum_{j=0}^{N-1} b_j |\lambda_j\rangle_{n_l} |u_j\rangle_{n_b} \tag{29}$$

where $|\lambda_j\rangle_{n_l}$ – is n_l-bit binary representation of λ_j.

- adding an auxiliary qubit and applying rotation under the condition $|lambda_j\rangle$

$$\sum_{j=0}^{N-1} b_j|\lambda_j\rangle_{n_l}|u_j\rangle_{n_b}\left(\sqrt{1-\frac{C^2}{\lambda_j^2}}|0\rangle+\frac{C}{\lambda_j}|1\rangle\right) \tag{30}$$

  where C – is a normalization constant and, as expressed in the current form above, should be less than the smallest eigenvalue λ_{min} in magnitude $|C| < \lambda_{min}$;
- applying the QPE † operation. In this case, ignoring QPE errors will be displayed in the following form:

$$\sum_{j=0}^{N-1} b_j|0\rangle_{n_l}|u_j\rangle_{n_b}\left(\sqrt{1-\frac{C^2}{\lambda_j^2}}|0\rangle+\frac{C}{\lambda_j}|1\rangle\right) \tag{31}$$

- measurement of the auxiliary qubit on a computational basis. If the result is 1, then the register is following after measurement:

$$\left(\sqrt{\frac{1}{\sum_{j=0}^{N-1}|b_j|^2/|\lambda_j|^2}}\right)\sum_{j=0}^{N-1}\frac{b_j}{\lambda_j}|0\rangle_{n_l}|u_j\rangle_{n_b} \tag{32}$$

- applying the observable M to calculate $F(x) = \langle x|M|x\rangle$.

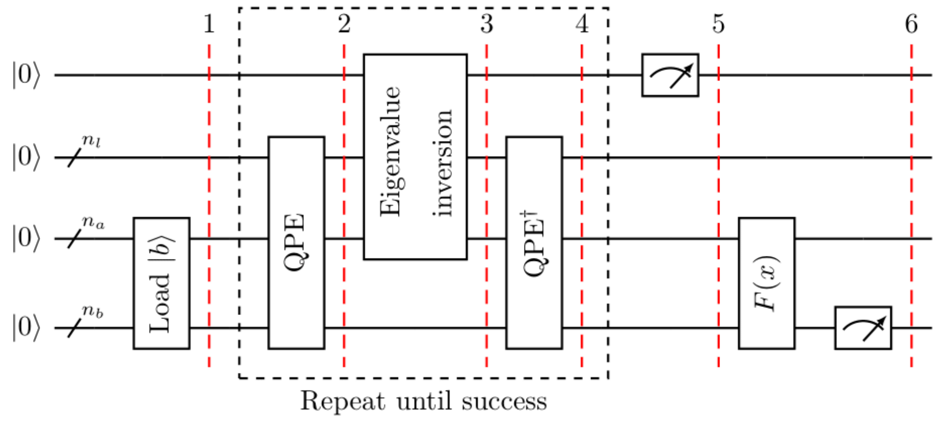

Figure 1: HHL algorithm schematic.

The following parameters affect the performance of the SLN solution: n is the size of the SLN, k is the condition number of the matrix A s is the sparsity of the matrix A, ε, is the accuracy of the solution.

The computational complexity for the conjugate vector method is $O(nsk \log 1/\varepsilon)$. The HHL algorithm has a computational complexity of $O(k^2 s^2 \varepsilon^{-1} \log n)$, which provides exponential acceleration depending on the size of the SLN n. However, one should remember that this algorithm does not provide an exact solution. By analogy with numerical methods, HHL gives an approximate solution with an accuracy ε.

Principal component analysis (PCA) is used to filter out correlated features and dimensionality reduction. We can achieve this by projecting the data point onto a new, lower-dimensional basis. The vectors covering this basis are called principal components and are the eigenvectors of the covariance matrix. The most costly action of PCA algorithm is considered to be the proper decomposition of the covariance matrix σ:

$$\sigma = \frac{1}{n-1} X^T X \tag{33}$$

where X is an $m \times n$, m is data points, n is features. As with the HHL algorithm, the QPCA algorithm should help us speed up the costly process of decomposing the covariance matrix.

The algorithm consists of two steps:

1) the QPCA algorithm requires reusable SWAP operations on $\rho \otimes \sigma$ for representation of the covariance matrix. The density operator is:

$$\rho = \sum_j \rho_j |\psi_j\rangle\langle\psi_j|, \sum_j \rho_j = 1 \tag{34}$$

where ρ_j is the probability of encountering a mutually orthogonal state ψ_j. This view requires reusable SWAP operations;

2) phase estimation to determine the eigenvalues of σ. The representation of σ in the density operator ρ form makes it possible to obtain the phase estimates of the eigenvector at the output and the associated eigenvalue (variance value) in the output register. The principal component with an eigenvalue and a vector is determined randomly but depends on the magnitude of the variance. Most likely, it will be the component with the greatest variance. Using ρ as the initial state, we get the state with the highest rank value:

$$\sum_j r_j |\chi_j\rangle\langle\chi_j| \otimes |\tilde{r}_j\rangle\langle\tilde{r}_j| \tag{35}$$

where χ_j – eigenvector of ρ operator, $\tilde{r}_j$ – estimate of the corresponding eigenvalue r_j. This state will be decomposed into an eigenvector/eigenvalue pair. Then we can get the mathematical expectation of an eigenvector with an eigenvalue r_j using the measurement operation $\langle\chi_j|M|\chi_j\rangle\rangle$.

The traditional PCA algorithm has a computational complexity of $O(d)$, where d is the number of features. The quantum analogue of this method is performed in time $O(R \log d)$, where R is the smallest rank of the approximation of the covariance matrix. In this case, the exponential acceleration by the quantum algorithm is achieved under the condition $R < d$ [10]. Support vector machine allows you to find a hyperplane maximizing the distance between two hyperplanes:

$$\sum_i a_j y_i - \frac{1}{2}\sum_i \sum_j a_i \overrightarrow{x}_i \times a_j \overrightarrow{x}_j \tag{36}$$

where x_i is the i^{th} – point in the feature space, y_j is the tagged class and a_j – parameter of the model. Thus, to find the optimal hyperplane, it is necessary to find $\overrightarrow{a} = [a_1, ..., a_m]$. Unlabeled data can be classified by the formula:

$$sign(\sum_i a_i \overrightarrow{x}_i \times \overrightarrow{x} - b) \tag{37}$$

The QSVM method allows you to speed up the calculation of the dot product $\overrightarrow{x}_i \times x$. This product is also known as the kernel matrix K.

Let us rewrite the problem of finding the vector $\overrightarrow{a}$ in the LS-SVM form of a system of linear equations to apply the HHL algorithm:

$$\begin{bmatrix} b \\ \overrightarrow{a} \end{bmatrix} = F^{-1} \begin{bmatrix} 0 \\ \overrightarrow{y} \end{bmatrix} \tag{38}$$

Matrix F consists of scalar products of the kernel matrix K of the training sample:

$$\begin{bmatrix} 0 & i^T \\ 1 & K + \frac{1}{\gamma} I \end{bmatrix} \tag{39}$$

where i is the unit vector, I is the unit matrix, γ is the hyperparameter.

We use HHL to find F^{-1}. At the same time, we pass the amplitude-encoded $|\overrightarrow{y}\rangle = [0, \overrightarrow{y}]$ classes' vector to the input register of the phase estimation eigenstate. Moreover, we regard $|\overrightarrow{y}\rangle$ as a superposition of eigenstates of F. As a consequence, $|F^{-1}\overrightarrow{y}\rangle$ exactly coincides with the desired solution $|b, \overrightarrow{a}\rangle$.

It remains us only to classify the data by obtaining a sign. If we store the superposition of the training data amplitudes a_i in one QRAM quantum register and the new data point $\overrightarrow{x}$ in the other, we will be able to apply the SWAP test procedure. This procedure is constructed so that the probability p becomes $p < 0.5$ for the +1 sign and $p \geq 0.5$ for the -1 sign. By repeating the SWAP test, it is possible to estimate the value of the probability p to the desired accuracy.

In the best case, the implementation of the SVM algorithm has a complexity of $O(poly(m, n))$, where m is the amount of data used for training, and n is the number of features. The quantum analog of the SVM algorithm allows you to obtain a logarithmic acceleration of $O(\log mn)$.

5. Research results

It was decided to use the QSVM algorithm using the Qiskit public library and the IBM Quantum Experience cloud service to solve the assigned task. For comparison, the original classic SVM algorithm and the Google Colaboratory cloud development environment were used. It was significant to check the practical applicability of quantum machine learning algorithms. As well as the existing algorithms' ability to correctly perform the task of not only binary but also multiple classifications have been checked. The dataset "ElectricalFaultDetection.csv" was selected to detect electrical faults at various locations on the power line. This dataset has six features and six classes. Since the maximum number of qubits in the free-for-all IBM machines is five, only the first five features participated in the training. The number of data points in this dataset is 7861. As it turned out, using all the data for training and testing would take at least a couple of days on a quantum device (without waiting in the queue). For this reason, we use a sample of shuffled data with a dimension of 72 for the quantum algorithm training. However, we used all data to train the classical SVM. Each sample has been divided into 75 % of the training data and 25 % of the test data. With the optimal selection of parameters, the classical SVM algorithm has been trained in 2.64 s. with an accuracy of 73.25 %. Let us consider in more detail the results obtained by execution on different quantum devices.

First, let's look at why training on large data sets on today's quantum devices is not desirable. Figure 2 shows how long it takes a model to train in a quantum simulator. As we can see, the time increases with the growth of the size of the training and test samples with a polymodal rate. It makes sense since we are only simulating quantum behavior.

If you perform the same actions on the real quantum device, the time will grow less and less with the growth of the sample according to the logarithmic law (figure 3). This result is in line with theory, but if one pays attention, the execution time for a sample size of 72 reaches almost 5 minutes. The classical method only needs about 3 seconds. Of course, as the size of the qubit increases, it will be possible to train the data in parallel, which will give a greater speedup. However, one cannot ignore the fact that the need to run the same circuit many times, external noise, and the time spent on error correction influence the total program execution time.

In terms of accuracy, it ranges from 61 % to 94 % on a training set of up to three data points. For the training set with a dimension of four and higher, we got an accuracy of 100 %. The accuracy of the classical SVM with the same training and test sample size was 83 %. Consider now the time it took for the program to execute on various quantum devices: qasm_simulator 474 s, lima 481 s, quito 352 s and belem 347 s. As we see, the runtime on real quantum devices ultimately does not differ much from the runtime on the simulator.

From the results obtained earlier, we can conclude that quantum devices are not efficient for small datasets. However, QPU needs more qubits for large datasets and, of course, more quantum volume. When QPUs achieve this condition, quantum computers can become an indispensable assistant in machine learning.

6. Conclusion

In the problem studying process using a quantum computer to solve the classifying machine learning problem from the theoretical side, the high advantage of quantum computing algorithms over classical ones was substantiated. It was written a module that allows you to perform the classification task on a classical and quantum device to check the current state of quantum computing. This module enables performing the classification task on a classical and quantum device. It has been proven that a quantum computing device can solve both binary and multiclass classification tasks. Empirical research has shown that the current state of quantum computing is far from quantum superiority in accelerating classical teaching methods. However, this does

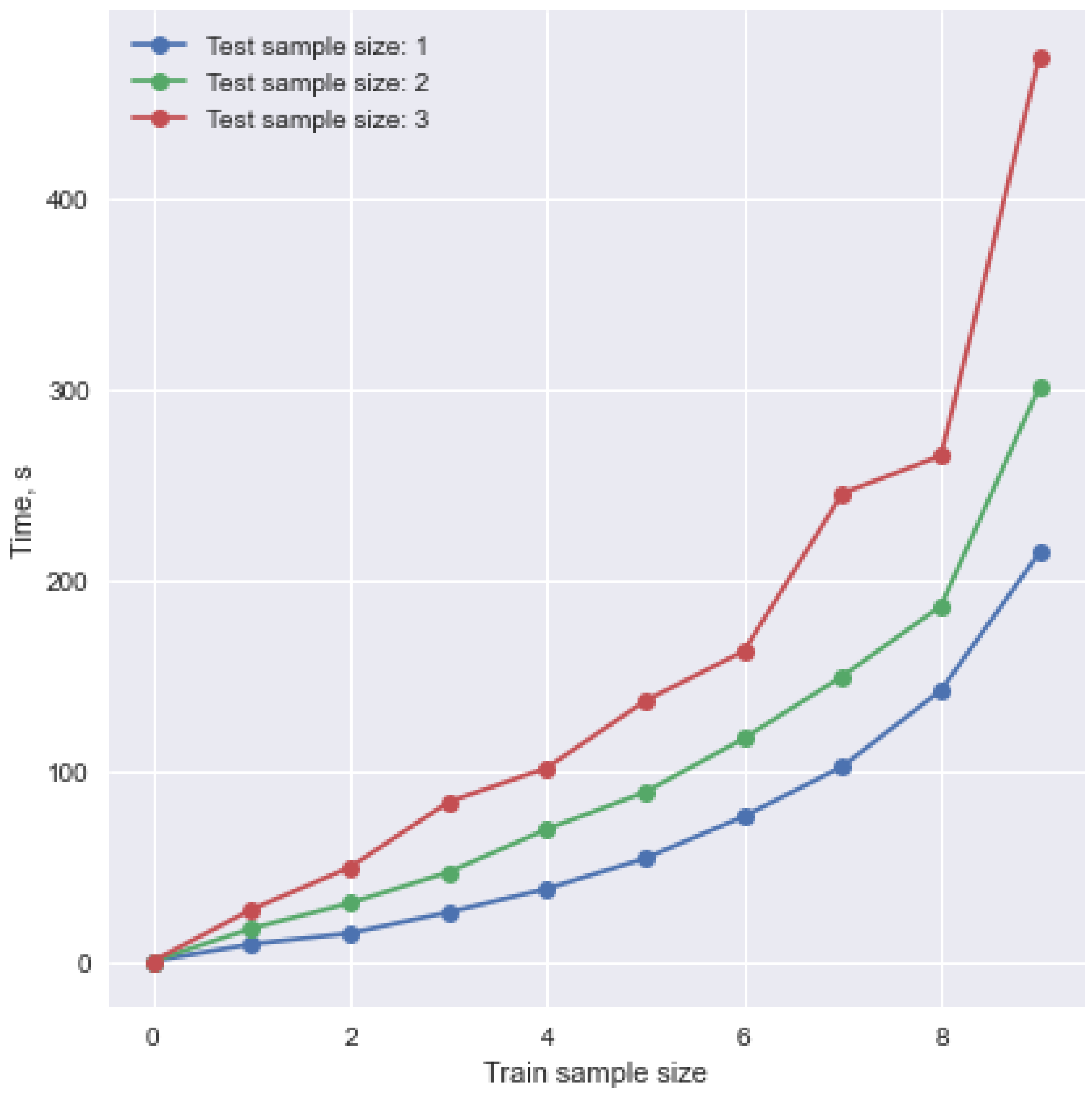

Figure 2: Training of the model on the IBM quantum simulator.

not negate the importance of research in this area. When the stable quantum computer is invented with sufficient computational resources to perform real-world tasks, the need for theoretically based quantum algorithms will increase. Therefore, an important task is to build fundamentally new algorithms for quantum machine learning. In the future, it is planned to study the possibility of using quantum machine learning to increase the forecasting accuracy of short-term time series.

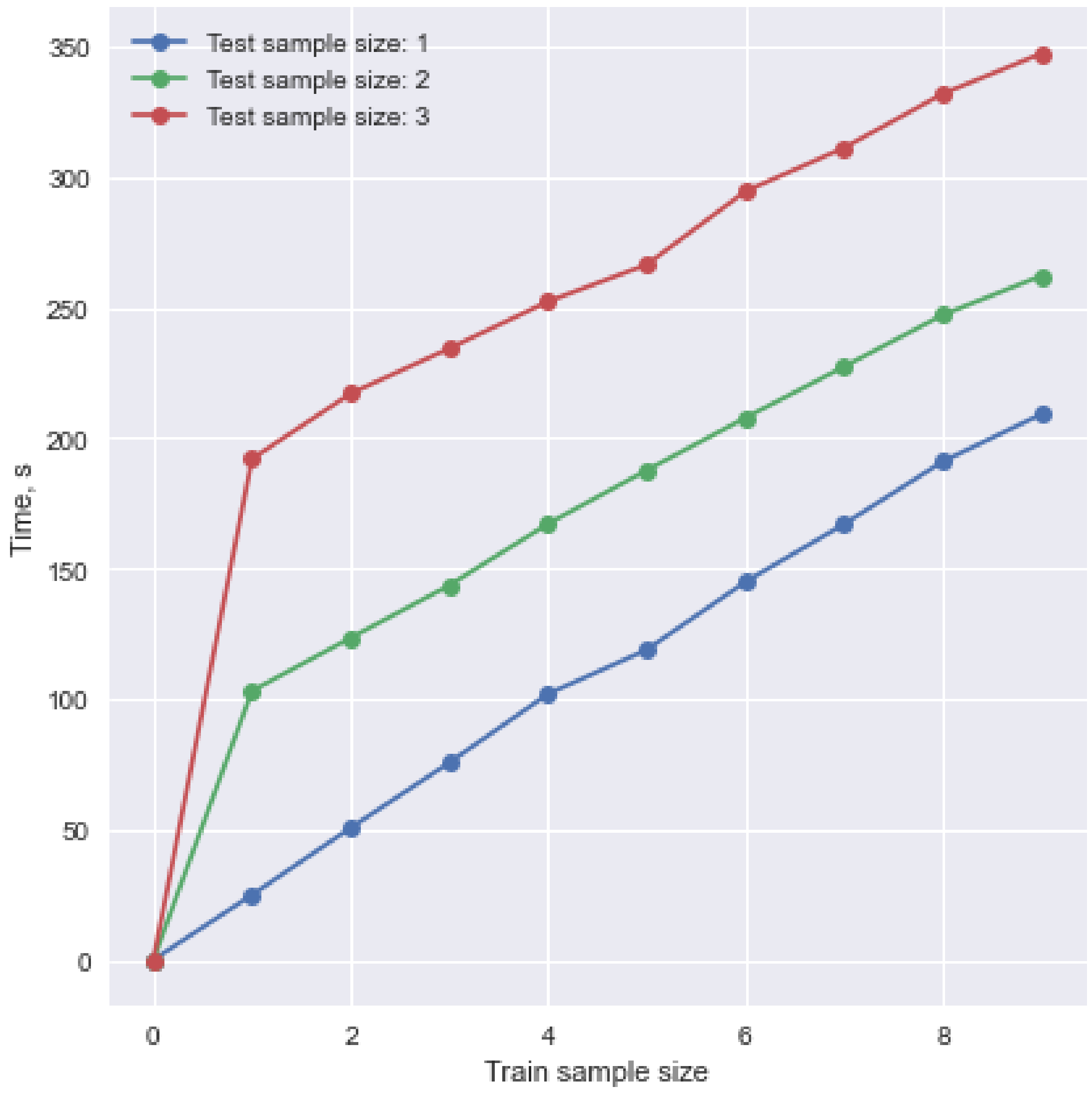

Figure 3: Training of the model on the IBM quantum belem device.

References

[1] L. V. Lehka, S. V. Shokaliuk, Quantum programming is a promising direction of IT development, CEUR Workshop Proceedings 2292 (2018) 76–82.

[2] S. O. Semerikov, A. M. Striuk, T. A. Vakaliuk, A. V. Morozov, Quantum information technology on the Edge, CEUR Workshop Proceedings 2850 (2021) 1–15. URL: http://ceur-ws.org/Vol-2850/paper0.pdf.

[3] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, B. Neyenhuis, Demonstration

of the trapped-ion quantum CCD computer architecture, Nature 592 (2021) 209–213. doi:10.1038/s41586-021-03318-4.

[4] L. V. Lehka, A. O. Bielinskyi, S. V. Shokaliuk, V. N. Soloviev, P. V. Merzlykin, Y. Y. Bohunenko, Prospects of quantum informatics and the study of its basics in the school course, in: S. Semerikov, V. Osadchyi, O. Kuzminska (Eds.), Proceedings of the Symposium on Advances in Educational Technology, AET 2020, University of Educational Management, SciTePress, Kyiv, 2022.

[5] R. V. Dushkin, Review of modern state of quantum technologies, Computer Research and Modeling 10 (2018) 165–179. doi:10.20537/2076-7633-2018-10-2-165-179.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, Quantum machine learning, Nature 549 (2017) 195–202. doi:10.1038/nature23474.

[7] P. V. Zahorodko, S. O. Semerikov, V. N. Soloviev, A. M. Striuk, M. I. Striuk, H. M. Shalatska, Comparisons of performance between quantum-enhanced and classical machine learning algorithms on the IBM Quantum Experience, Journal of Physics: Conference Series 1840 (2021) 012021. doi:10.1088/1742-6596/1840/1/012021.

[8] P. V. Zahorodko, Y. O. Modlo, O. O. Kalinichenko, T. V. Selivanova, S. O. Semerikov, Quantum enhanced machine learning: An overview, CEUR Workshop Proceedings 2832 (2020) 94–103. URL: http://ceur-ws.org/Vol-2832/paper13.pdf.

[9] B. Duan, J. Yuan, C.-H. Yu, J. Huang, C.-Y. Hsieh, A survey on hhl algorithm: From theory to application in quantum machine learning, Physics Letters A 384 (2020) 126595. doi:10.1016/j.physleta.2020.126595.

[10] S. Lloyd, M. Mohseni, P. Rebentrost, Quantum principal component analysis, Nature Physics 10 (2014) 631–633. doi:10.1038/nphys3029.