

Quantum support vector machine for multi classification

Li Xu, Xiao-yu Zhang, Ming Li[✉] and Shu-qian Shen

College of Science, China University of Petroleum, Qingdao 266580, China

E-mail: liming@upc.edu.cn

Received 21 March 2024, revised 9 May 2024

Accepted for publication 9 May 2024

Published 12 June 2024



Abstract

Classical machine learning algorithms seem to be totally incapable of processing tremendous data, while quantum machine learning algorithms could deal with big data unhurriedly and provide exponential acceleration over classical counterparts. In this paper, we propose two quantum support vector machine algorithms for multi classification. One is the quantum version of the directed acyclic graph support vector machine. The other one is to use the Grover search algorithm before measurement, which amplifies the amplitude of the phase storing of the classification result. For k classification, the former provides quadratic reduction in computational complexity when classifying. The latter accelerates the training speed significantly and more importantly, the classification result can be read out with a probability of at least 50% using only one measurement. We conduct numerical simulations on two algorithms, and their classification success rates are 96% and 88.7%, respectively.

Keywords: quantum support vector machine, quantum feature mapping, Grover search algorithm

1. Introduction

Over the past two decades, quantum computing has received extensive research and developed rapidly, and has been given high expectations in certain computational tasks [1]. Quantum computing has been proven to provide significant acceleration in solving some important computational problems, such as large number factorization via the Shor algorithm [2], unstructured search using the Grover search algorithm [3], and quantum algorithms for solving linear equations [4]. These are three prominent examples among many problems that have been discovered.

At the same time, machine learning has become a powerful tool in modern computation. There are two types of machine learning, supervised learning and unsupervised learning, the difference is whether the label is stamped in advance. Support vector machine (SVM) for binary classification is a frequently-used supervised learning algorithm [5, 6]. Its variants, such as least-squares SVM [7] and twin SVM [8], have been widely studied and applied. However, like the vast majority of machine learning algorithms, SVM is weak in dealing with big data. Quantum least-squares support vector machine (QSVM) emerges in the era of big data by the reason of exponential acceleration over classical algorithms due to quantum parallelism [9], and the experiment also

proves its feasibility [10]. Compared with SVM, QSVM owns logarithmic level complexity due to the outstanding Harrow–Hassidim–Lloyd (HHL) algorithm and quantum superposition. That is the fundamental reason that has lead QSVM to be a hot research topic over the past few years [11–16]. However, the quantum acceleration meets obstacles when taking the time of loading classical data into the amplitude of quantum state into consideration.

Schuld *et al* and Havlíček *et al* have proposed an artful path that could circumvent the preparation of coherent superpositions by introducing quantum feature mapping (QFM) [17, 18]. The core idea of QFM is mapping the data to an exponential Hilbert feature space, and then finding a fitting curve or separating hyperplane for fitting or classification. Generally speaking, the higher the dimension to which data is mapped, the better the expressive power obtained. Utilizing QFM skillfully, one cannot only circumvent the invocation of quantum random access memory (QRAM) [19, 20], but achieve better machine learning performance through the help of the exponential Hilbert space. The QFM based binary classification SVM is extensively exploited, but there still lacks an extension of multi classification.

In this paper, we propose two QSVM algorithms for multi classification with lower complexity in contrast with canonic multi classification QSVMs. One of them plays a

trick in the classification path, while the other one significantly accelerates the training stage by means of fully utilizing quantum parallelism, and reduces the reading cost with the application of the simplified quantum search algorithm. In numerical simulations, we not only verify the feasibility of the algorithms, but also explore the impact of QFMs on classification. The rest of the paper is organized as follows. In section 2, we briefly review QSVM, and two common used QSVM algorithms for multi classification. Quantum feature mapping is addressed in section 3. The two QSVM algorithms we propose are introduced in section 4. The corresponding numerical simulations are presented in section 5. Finally, section 6 gives conclusions and prospects.

2. Background and related work

This section provides a comprehensive overview of the original QSVM algorithm, as well as two mainstream multi classification QSVMs.

2.1. Quantum least-squares support vector machine

Given a training set $T = \{(\vec{x}_i, y_i) : \vec{x}_i \in \mathbb{R}^N, y_i = \pm 1\} (i = 1, 2, \dots, M)$, suppose that there are oracles we can invoke at any time and feedback a quantum state that encodes the features of $\vec{x}_i$ into amplitudes. Similar to SVM, QSVM needs to find a separating hyperplane to partition the training data. However, the introduction of the least-squares method transforms the inequality constraint of the soft margin SVM into an equality constraint

$$\begin{aligned} \min_{w, b, e} \quad & \frac{1}{2} \|w\|^2 + \frac{\gamma}{2} \sum_{i=1}^M e_i^2, \\ \text{s.t.} \quad & y_i = w^T x_i + b + e_i, \end{aligned} \quad (1)$$

where e_i represents the degree to which the sample does not satisfy the constraint. The user-specified γ determines the relative weight of the training error and the SVM objective.

We further derive the dual program of equation (1) based on Lagrangian duality

$$L = \frac{1}{2} \|w\|^2 + \frac{\gamma}{2} \sum_{i=1}^M e_i^2 - \sum_{i=1}^M \alpha_i [w^T x_i - b - y_i + e_i]. \quad (2)$$

The conditions for optimality

$$\begin{aligned} \frac{\partial L}{\partial w} = 0 & \rightarrow w = \sum_{i=1}^M \alpha_i y_i x_i, \\ \frac{\partial L}{\partial e_i} = 0 & \rightarrow \alpha_i = \gamma e_i, \\ \frac{\partial L}{\partial \alpha_i} = 0 & \rightarrow y_i = w^T x_i + b + e_i. \end{aligned}$$

Substituting the optimal conditions into equation (2) yields the linear equation

$$F \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & K + \gamma^{-1} I \end{pmatrix} \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{y} \end{pmatrix}. \quad (3)$$

Here, K is symmetric kernel matrix whose entry $K_{ij} = x_i^T x_j$, $\vec{y} = (y_1, y_2, \dots, y_M)^T$, and $\vec{1} = (1, 1, \dots, 1)^T$. Once the above

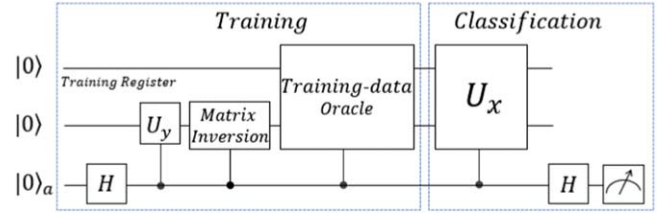


Figure 1. Brief circuit diagram of QSVM. U_y stores the labels of training samples. The Matrix Inversion is employed to acquire the hyperplane parameters. The Training-data Oracle prepares $|u\rangle$. The information of query state is stored in U_x .

equation is solved by a linear solver, $|u\rangle$ can be constructed through the Training-data Oracle for classification based on the obtained hyperparameters b and $\vec{\alpha}$

$$|u\rangle = \frac{1}{\sqrt{N_u}} (b|0\rangle|0\rangle + \sum_{i=1}^M \alpha_i |\vec{x}_i\rangle|i\rangle |\vec{x}_i\rangle),$$

with the normalization factor $N_u = b^2 + \sum_{k=1}^M \alpha_k^2 |\vec{x}_k|^2$.

When classifying, one should prepare the query state $|v\rangle$ as follows

$$|v\rangle = \frac{1}{\sqrt{N_x}} (|0\rangle|0\rangle + \sum_{i=1}^M |\vec{x}\rangle|i\rangle |\vec{x}\rangle),$$

where $N_x = M|\vec{x}|^2 + 1$. Take notice that the state $|v\rangle$ does not appear directly in the circuit, but is stored in the unitary matrix U_x , and $U_x|v\rangle = |00\rangle$. Before taking measurements, the expression of the final state should be $|\Phi\rangle = \frac{1}{2}(|00\rangle|0\rangle_a + |00\rangle|1\rangle_a + |\phi\rangle|1\rangle_a - |\phi\rangle|1\rangle_a)$, with $|\phi\rangle = U_x|u\rangle$, and $| \cdot \rangle_a$ is the ancillary qubit. The classification result will be revealed by the sign of $\langle v|u\rangle$, i.e. $\langle 00|\phi\rangle$. One just measures the auxiliary qubit and subtracts the probability P_1 of obtaining 1 from the probability P_0 of obtaining 0, then gets the explicit classification formula

$$y(\vec{x}) = \text{sgn}(\langle 00|\phi\rangle) = \text{sgn}(P_0 - P_1).$$

If the result is greater than 0, x belongs to +1 class. Otherwise, x belongs to -1 class. The circuit diagram is shown in figure 1.

2.2. Quantum multi classification support vector machine

One-versus-one QSVM [13]—In the training section, we need to train any two of these k classes. There are $k(k-1)/2$ equations in the form of equation (3) to be solved. To perform classification, the query state of the trial datum $\vec{x}$ will vote for a type when it runs a classifier. After running all classifiers, the class with the largest number of votes is the classification result.

One-versus-rest QSVM [14]—In the training section, we construct k classifiers, each of which separates one class from the others. When classifying, each classifier runs the query state once, and the classification results are read out with probabilities. Then, QRAM is used to find the maximum probability, which is the classification result.

In summary, one-versus-one QSVM algorithms need to construct $k(k-1)/2$ classifiers and classify $k(k-1)/2$ times. Generally speaking, one-versus-one QSVM owns a high

accuracy, but it also incurs a higher cost in training as a punishment. In contrast to one-versus-one QSVM, one-against-all QSVM algorithms just demand k classifiers and classify k times. However, the problem of imbalanced classes will lead to an unpleasant accuracy.

3. Quantum feature mapping

Loading classical data into quantum state via QRAM will take $O(N)$ operations, where N represents the data dimension. This scaling can dominate the complexity of quantum algorithms, thereby weakening potential quantum accelerations. This is not advisable unless original data must be used in quantum machine learning.

The essence of machine learning is to learn and analyze the structure of existing datasets, and then utilize the experience of learning previous features to process new data. Directly and inefficiently preparing quantum states such as amplitude encoding is undesirable, we have a better choice—quantum feature mapping.

Similarly, by learning and analyzing data structures, we write classical data x_i into parameterized quantum circuits U_{x_i} to obtain exponential dimensional quantum state $|\phi(x_i)\rangle = U_{x_i}|0\dots 0\rangle$, which better represents the characteristics of the data. Hence, we can obtain better quantum machine learning models using the quantum feature states. For kernel function based machine learning algorithms such as SVM, QFM can be independently used for kernel function, and the remaining problems in the shape of equation (3) can be solved by classical, quantum, or hybrid means.

4. Quantum support vector machine for multi classification

In this section, we present our QSVMs for multi classification. In essence, the one-versus-one directed acyclic graph quantum support vector machine (DAG-QSVM) is a quantum version of the classical directed acyclic graph support vector machine, and it is a special one-versus-one QSVM algorithm. The iterative quantum support vector machine for multi classification (IQSVM) belongs to the one-versus-rest QSVM algorithm, which fully utilizes quantum parallelism and quantum search algorithm to achieve acceleration and reduce reading cost.

4.1. One-versus-one directed acyclic graph quantum support vector machine

In the training of DAG-QSVM, we need to construct classifiers layer by layer. Assume that the number of training samples $\{(\vec{x}_{ij}, y_{ij}): \vec{x}_{ij} \in \mathbb{R}^N, y_{ij} = j, j = 1, 2, \dots, k\}$ in different classes are $M/2$. As shown in figure 2, we start from the first layer by regarding class 1 as a positive class, and class k as a negative class. After the execution of QSVM in section 2.1, it is the second layer's turn. In order to facilitate the follow-up work, we choose the small class as the positive

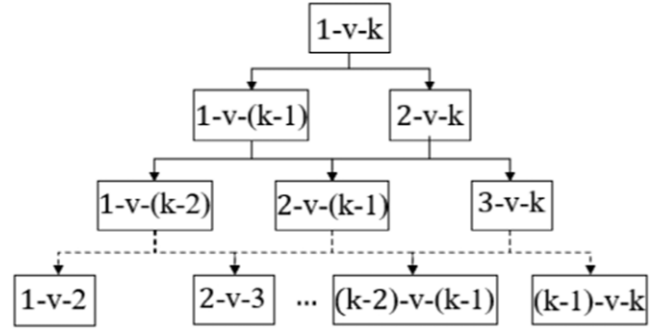


Figure 2. Sketch of the directed acyclic graph support vector machine. The number indicates the class.

class and the big class as the negative class. Applying the quantum least-squares support vector machine algorithm to each classifier layer by layer until all the $\frac{k(k-1)}{2}$ classifiers are complete in training.

In the classification stage, it is worth noting that we have to classify layer by layer. For example, if the classification result is 1 in the first layer, then in the second layer we choose the classifier 1-v-(k-1). Although we are not sure whether the final result is 1, we could confirm that the final result is not k absolutely. This approach is repeated until the classifier in the last layer gives the classification result for the trail state.

The advantage of DQG-QSVM is that the analysis of complexity can be established without difficulty. The executions of classifiers during classification have been reduced from $k(k-1)/2$ to $k-1$, resulting in a quadratic acceleration.

4.2. Iterative quantum support vector machine for multi classification

This IQSVM for multi classification algorithm is limited to one-versus-rest QSVM. Before solving equations about hyperplane parameters, we have to preprocess the matrix so that it can be solved only once. Set class m as $+1$, the rest $k-1$ as -1 and specify γ as an invariant constant, that is, maintain the matrix F of equation (3) invariant for all classifiers. Note that $K_{ij} = k(x_i, x_j)$, which means that the kernel function remains unchanged as long as we opt for the same QFM for each classifier. In summary, we obtain the following matrix equations

$$F \begin{pmatrix} b_m \\ \vec{\alpha}_m \end{pmatrix} = \begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & K + \gamma^{-1}I \end{pmatrix} \begin{pmatrix} b_m \\ \vec{\alpha}_m \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{y}_m \end{pmatrix}, \quad m = 1, 2, \dots, k. \quad (4)$$

In our previous work, we have modified the HHL algorithm to enable it to solve multiple linear equations using a single quantum circuit [21]. The form of the above k equations should be rewritten as follows

$$(I_k \otimes F) \vec{b}_\alpha = Y,$$

where the right end of equation (4) is arranged in a column, forming Y . In addition to the modified HHL algorithm, we can also apply variational quantum algorithms to this linear equation. Since $I_k \otimes F$ has the same number of decomposition

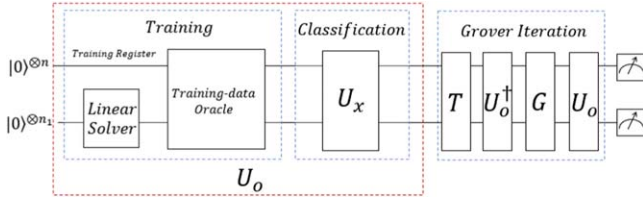


Figure 3. Brief circuit diagram of IQSVM. The Grover Iteration section is a repeatable part and $n_1 = \lceil \log km \rceil$.

terms as F , the system with a larger dimension will not add too much extra work to us. The detailed processes are shown in figure 3, we start with $|0\rangle^{\otimes n}|0\rangle^{\otimes n_1}$. After applying the linear system solver to the equation (4), the state becomes

$$|0\rangle|b_K; \vec{\alpha}_K\rangle,$$

with $|b_K; \vec{\alpha}_K\rangle = \frac{(b_1; \vec{\alpha}_1; \dots; b_k; \vec{\alpha}_k)}{\|(b_1; \vec{\alpha}_1; \dots; b_k; \vec{\alpha}_k)\|}$. Then, the feature states are compiled into the final state of the training stage through the training register, one gets

$$|u_1; u_2; \dots; u_k\rangle, \quad (5)$$

with each $|u_i\rangle = b_i|0\rangle|0\rangle + \sum_{j=1}^M \alpha_i^j |j\rangle |\psi(x_j)\rangle$.

In the classification section, we construct a unitary matrix U_x according to trial datum x ,

$$|v\rangle = \frac{1}{\sqrt{N_x}}|0\rangle|0\rangle + \sum_{i=1}^M |i\rangle |\psi(x)\rangle, \quad N_x = M + 1,$$

$$\langle v|U_{x_0}|0 \dots 0\rangle = 1, \quad U_x = I_k \otimes U_{x_0}.$$

The k classification results are stored in k amplitudes at specific positions in the final state $|\xi\rangle = U_x|u_1; u_2; \dots; u_k\rangle$. If there is no error present, the category of x will be given by the classifier that classifies it as positive. It is not advisable to directly measure $|\xi\rangle$ because the measurement result is a square of amplitude and cannot determine positive or negative. Due to the special properties of this state, we can perform a specific operation on it so that the classification results can be measured directly.

For the sake of simplicity, suppose the classification results are stored in the first k amplitudes of $|\xi\rangle$, this can be achieved by a swap operator. We called the aforementioned specific operation as the mean inversion (MI) operator

$$\text{MI} = \begin{pmatrix} \frac{2}{N} - 1 & \frac{2}{N} & \dots & \frac{2}{N} \\ \frac{2}{N} & \frac{2}{N} - 1 & \dots & \frac{2}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{2}{N} & \frac{2}{N} & \dots & \frac{2}{N} - 1 \end{pmatrix}.$$

Applying an MI operator to the first k amplitudes of $|\xi\rangle$ can amplify the absolute value of the positive amplitude to the maximum. Now, we can obtain the final classification result through measurement, as it has the maximum absolute value, i.e. the maximum probability.

The useful information is stored in the first k amplitudes of the final state, but it only occupies a tiny portion. This makes it difficult for the quantum state to collapse to the target phases during measuring, thus requires a large number

of measurements, with the vast majority still collapsing to useless phases. What is more, if the shots for first k amplitudes are not enough, the impact of random errors is greatly amplified.

The phases of the target amplitudes are known in advance, and we can use a simplified version (no oracle workspace to mark target phase) of the quantum search algorithm to amplify their amplitudes. The workflow of a single quantum search algorithm is as follows: (i) implement a phase inversion operator T to invert the amplitudes for first k phases; (ii) apply $U_o^\dagger$ to $T|\xi\rangle$; (iii) reverse the state by $|0 \dots 0\rangle$; (iv) apply U_o to step (iii) yielding the state after amplitude amplification. It should be noted that $U_o|0 \dots 0\rangle = |\xi\rangle$. Repeating the above operations for $R \leq \frac{\pi\sqrt{MN}}{4}$ times, the amplitudes of classification results can be amplified maximally [1]. $N = 2^n$ and M is the number of samples. The final state will collapse to the target phases with a high probability, thereby, one gets the classification result accurately. The algorithm flowchart is shown in **Algorithm 1**.

Algorithm 1. Iterative quantum support vector machine

Input. Dataset and encoding scheme.

1. Build a circuit for calculating the inner product for kernel matrix F .
2. Apply a linear solver to $(I_k \otimes F)\vec{b}_\alpha = Y$.
3. Construct the classification circuit according to $\vec{b}_\alpha$ and training samples, that is, equation (5).
4. Prepare the query state based on the trial datum and then compile the query state into U_x .
5. Implement a simplified quantum search algorithm to the final state to amplify the amplitudes that store the results.
6. Repeat step 5 for R times.
7. Execute an MI operator and then measure the final state.

5. Numerical simulations

We select the Iris dataset for numerical simulations, which has four features, 150 samples, and is divided into three classes equally. The execution of the HHL algorithm requires a very deep circuit and controlled unitary gates, it is not advisable for near-term quantum devices. Before the emergence of fault-tolerant quantum computers, the theoretical effects of the HHL algorithm far exceeds its practical value. Therefore, we choose the variational quantum algorithm as an alternative for solving equations, as its structure is simple and easy to implement. The following simulations are conducted in qiskit package, and the Constrained Optimization BY Linear Approximations optimizer is chosen.

5.1. DAG-QSVM

In the construction of QSVM, four samples from each class are chosen as training samples. To explore the impact of QFM, we adopted three encoding methods, namely amplitude encoding, Pauli y rotation operator, and the QFM $U_{\Phi(x)} = U_{\phi(x)} H^{\otimes n} U_{\phi(x)} H^{\otimes n}$ defined in [18] to generate state

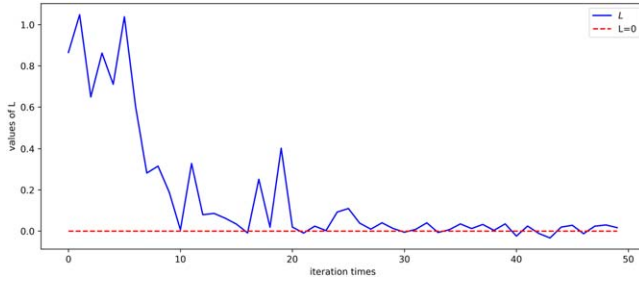


Figure 4. The relationship between the number of optimization iterations and the loss function value of 1-v-3 classifier.

and

$$U_{\phi}(x) = \bigotimes_{i=1}^n R_z(x^{(i)}).$$

A first-order kernel function is used, i.e. $K_{ij} = \langle \phi(x_i) | \phi(x_j) \rangle$. After obtaining the kernel matrix, we need to construct classifiers by solving a system of equations by means of variational quantum algorithm. The DQG-QSVM of our simulation is divided into two layers, with the first layer 1-v-3 classifier, and the second layer 1-v-2 and 2-v-3 classifiers. Figure 4 shows the loss function when training the 1-v-3 classifier using the $U_{\phi(x)}$ encoding method, the values below 0 are caused by measurement errors.

In the classification stage, a query state goes through the 1-v-3 classifier at first. The subsequent use of which classifier is determined by the classification results of 1-v-3 classifier, that is, result 1 corresponds to 1-v-2 classifier, and result 3 corresponds to the 2-v-3 classifier. The final result is given by the last two classifiers, and the classification results are shown in Table 1. All classifiers can accurately classify Class 1, and the errors occur from Class 2 and 3 as they are not linear separable. The poor performance of amplitude encoding reflects the shortage of linear classifier. In contrast, the both QFMs show satisfactory classification accuracies (over 95%) using only 12 samples for training, accounting for less than 10%.

5.2. IQSVM

The focus of this simulation is on amplitude amplification, so we choose only three (one for each class) samples for a proof-of-principle. We also chose a simpler QFM, the Pauli y rotation operator, which simply compiles the features into rotation angles to prepare the feature state. The kernel matrix obtained after retaining two decimal places is

$$F = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & -0.04 & -0.22 \\ 1 & -0.04 & 1 & 0.64 \\ 1 & -0.22 & 0.64 & 1 \end{pmatrix}.$$

The solution gives $|b_K; \alpha_K\rangle = (-0.01; 0.17; -0.12; -0.04; -0.12; -0.12; 0.56; -0.44; -0.04; -0.04; -0.44; 0.48)$, then we construct equation (5) for classification based on it and the feature states.

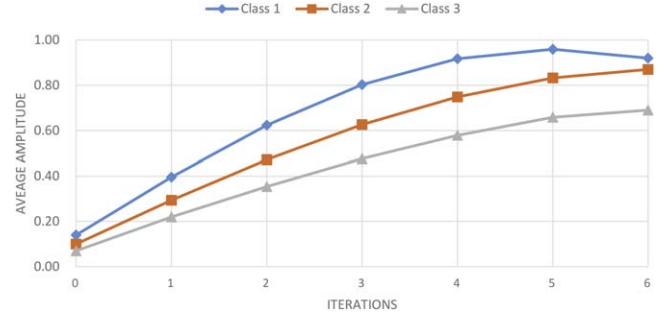


Figure 5. Diagram of the relationship between average amplitude and iterations.

Table 1. The classification results of DAG-QSVM for three different kernels. The number in () represents the number of misclassifications in the class.

	1	2	3	Error	Accuracy
Amplitude	50	57(23)	43(16)	39	74%
R_y	50	54(5)	46(1)	6	96%
$U_{\phi(x)}$	50	51(4)	49(3)	7	95.3%

After calculation, the maximum number of iterations is $R \leq 2\pi$. The average amplitudes are shown in figure 5, and the average amplitudes for Class i ($i = 1, 2, 3$) only represents 50 samples of its class. A total of 17 errors occurred, with a classification success rate of 88.7%. This is acceptable as we only use three samples for training. The final average amplitudes of the three classes of samples are 0.92, 0.87, and 0.69, respectively. Additionally, the extreme value 0.96 of Class 1 appears in the 5th iteration.

For phases with amplitudes exceeding 0.9, we have a probability of at least 81% for gaining the classification result with only one measurement. This is a considerable success rate, however, Class 2 and 3 are not able to reach this threshold, especially with an average value of only 0.69 for Class 3. But we find that if we exclude erroneous results (four in Class 2 and 15 in Class 3), the average amplitude of these two classes can be increased to 0.90 and 0.82, respectively. Although it does not reach 0.9, the average amplitude of Class 3 is also within an acceptable range.

6. Conclusion and discussion

In this study, we propose two QSVM algorithms for multi classification and conduct corresponding numerical simulations. The DAG-QSVM is a kind of one-versus-one QSVM which plays a trick on the classifying path, enabling it to complete classification with considerable accuracy and efficiency. Conventional one-versus-one QSVM usually implements all the classifiers, while our algorithm just demands $k - 1$ times classification. Therefore, it is possible to achieve quadratic acceleration.

Compared with general one-versus-rest QSVM, the one-versus-rest iterative quantum support vector machine

accelerates significantly. Firstly, it utilizes quantum parallelism to achieve a one-time acceleration in respect of solving linear equations, whether using HHL algorithm or variational quantum algorithm. Secondly, and more importantly, the use of a simplified quantum search algorithm greatly reduces the burden of measurements. We verify through numerical simulation that the significant increase in average amplitude allows us to accurately read classification results with only a few measurements.

We also compare the impact on SVM of different encoding methods. The numerical simulation results demonstrate the advantage of QFM over amplitude encoding in nonlinear classification. This confirms the powerful expressiveness of QFM in terms of data features. We conclude that QFM will bring substantial improvements to kernel-based machine learning algorithms, and we will further study it in the future.

Acknowledgments

This work is supported by the Shandong Provincial Natural Science Foundation for Quantum Science (No. ZR2021LLZ002) and the Fundamental Research Funds for the Central Universities (No. 22CX03005A).

ORCID iDs

Ming Li  <https://orcid.org/0000-0003-0835-9635>

References

- [1] Nielsen M A and Chuang I 2002 *Quantum computation and quantum information* (Cambridge University Press)
- [2] Shor P 1997 Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer *SIAM J. Comput.* **26** 1484
- [3] Grover L K 1997 Quantum mechanics helps in searching for a needle in a haystack *Phys. Rev. Lett.* **79** 325
- [4] Harrow A W, Hassidim A and Lloyd S 2009 Quantum algorithm for linear systems of equations *Phys. Rev. Lett.* **103** 150502
- [5] Zhou Z-H 2016 *Machine Learning* (Tsinghua University Press)
- [6] Vapnik V 1998 *Statistical learning theory* (Wiley) Vol. 3, 401–92 Chapter 10–11
- [7] Suykens J A K and Vandewalle J 1999 Least Squares Support Vector Machine Classifiers *Neural Process. Lett.* **9** 293
- [8] Khemchandani J R and Chandra S 2007 Twin Support Vector Machines for Pattern Classification *IEEE Trans. Pattern Anal. Mach. Intell.* **29** 905–10
- [9] Rebentrost P, Mohseni M and Lloyd S 2014 Quantum Support Vector Machine for Big Data Classification *Phys. Rev. Lett.* **113** 130503
- [10] Li Z-K et al 2015 Experimental realization of a quantum support vector machine *Phys. Rev. Lett.* **114** 110504
- [11] Hsu C-W and Lin C-J 2002 A comparison of methods for multiclass support vector machines *IEEE Trans. Neural Networks* **13** 415–25
- [12] Wang D-Y et al 2020 insulation defect diagnostic method for oip bushing based on multiclass LS-SVM and Cuckoo Search *IEEE Trans. Instrum. Meas.* **69** 163–72
- [13] Bishwas A K, Mani A and Palade V 2018 An all-pair quantum svm approach for big data multiclass classification *Quantum Inf. Process.* **17** 282
- [14] Bishwas A K, Mani A and Palade V 2016 Big data classification with quantum multiclass svm and quantum one-against-all approach *2nd International Conference on Contemporary Computing and Informatics (IC3I) (Greater Noida, India)* (IEEE) **875–80**
- [15] Uke D, Soni K K and Rasool A 2020 Quantum based support vector machine identical to classical model *11th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (Kharagpur, India)* (IEEE) **1–6**
- [16] Ye Z-K et al 2020 Quantum speedup of twin support vector machines *Sci. China Inf. Sci.* **63** 189501
- [17] Schuld M and Killoran N 2019 Quantum machine learning in feature Hilbert spaces *Phys. Rev. Lett.* **122** 040504
- [18] Havlíček V et al 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209–12
- [19] Giovannetti V, Lloyd S and Maccone L 2008 Quantum random access memory *Phys. Rev. Lett.* **100** 160501
- [20] Giovannetti V, Lloyd S and Maccone L 2008 Architectures for a quantum random access memory *Phys. Rev. A* **78** 052310
- [21] Xu L et al 2022 Quantum Algorithm for Matrix Equation of the Form $AX = B$ *Laser Phys. Lett.* **19** 055202