# Quantum KNN Classification With $K$ Value Selection and Neighbor Selection

Jiaye Li, Jian Zhang, Jilian Zhang, and Shichao Zhang, *Senior Member, IEEE*

***Abstract*—The K-nearest neighbors (KNNs) algorithm is one of Top-10 data mining algorithms and is widely used in various fields of artificial intelligence. This leads to that quantum KNN (QKNN) algorithms have developed and achieved certain speed improvements, denoted as Q-KNN. However, these Q-KNN methods must face two key problems as follows. The first one is that they are mainly focused on neighbor selection without paying attention to the influence of $K$ value on the algorithm. The second is that only the neighbor selection process is quantized, and the selection of $K$ value is not quantized. To solve these problems, this article designs a novel quantum circuit for KNN classification, so as to simultaneously quantumize the neighbor selection and $K$ value selection process. Specifically, the least squares loss and sparse regularization term are first used to construct the objective function of the proposed quantum KNN, so that it can simultaneously obtain the optimal $K$ value and $K$ nearest neighbors of the testing data. And then, a new quantum circuit is proposed to quantumize the process through quantum phase estimation, controlled rotation, and inverse phase estimation techniques. Finally, experiments are conducted with qiskit and MATLAB to output the quantum and classical results of the algorithm, verifying that the proposed algorithm can output the optimal $K$ value and $K$ nearest neighbors for each testing data.**

***Index Terms*—Controlled rotation, $K$ value, phase estimation, quantum circuit, quantum KNN (QKNN).**

## I. Introduction

K-NEAREST neighbor (KNN) classification is one of the simplest methods in data mining classification technology. It has a wide range of applications in fields, such as character recognition, text classification, and image recognition. It is a lazy learning algorithm, which means that there is no training process, and given a testing data, it only needs to predict the state of the testing data based on the states of its $K$ nearest neighbors in the training data. The main challenges for K-nearest neighbor (KNN) include: 1) $K$ value selection and 2) neighbor selection, where neighbor selection includes distance measurement and neighbor search.

Since its introduction by Cover and Hart [1], many researchers have proposed improved models for KNN. In recent years, with the development of quantum technology, researchers have proposed several quantum KNN (QKNN) algorithms. For example, Wiebe et al. [2] used quantum technology to improve KNN by calculating inner products and Euclidean distances. Ruan et al. [3] designed a quantum circuit for computing the Hamming distance between testing data and training data for KNN. Feng et al. [4] designed a new distance metric function (i.e., Polar distance), and quantumized it for KNN classification. Although these QKNN algorithms have improved KNN in terms of distance measurement, they all ignore the search for near neighbors (i.e., finding the $K$ smallest distances between the testing data and all training data). Subsequently, Basheer et al. [5] used the quantum $K$ maxima algorithm to perform neighbor search for KNN, thus addressing the challenge of neighbor search. It can be seen that the above QKNN algorithms only optimize one challenge in distance measurement or neighbor search.

Besides the above-mentioned QKNN algorithms, some researchers have proposed new ones. For example, Dang et al. [6] used controlled swap gates and quantum minimum search algorithm to obtain neighbors. Zhang et al. [7] proposed a quantum weighted KNN algorithm. Tian and Baskiyar used Ruan and Baskiyar's [8] quantum Hamming distance calculation method and Grover search algorithm to construct QKNN. Gao et al. [9] used quantum Mahalanobis distance and quantum minimum search algorithm to construct QKNN. Li et al. [10] proposed a quantum $K$-fold cross-validation method to improve KNN. These QKNN algorithms improve KNN by addressing both distance measurement and neighbor search challenges, which is a significant improvement.

In summary, although current QKNN algorithms have made a series of breakthroughs, they have not addressed the challenge of $K$ value selection. Different $K$ values will produce different prediction results. If a smaller $K$ value is selected, it is equivalent to using a smaller neighborhood to predict the testing data. If noise appears in the neighborhood, it will lead to incorrect predictions. If a larger $K$ value is selected, although it can avoid the influence of noise, it will cause a large number of dissimilar samples to enter the neighborhood, leading to incorrect predictions. Fig. 1 shows the position distribution of a testing data and training data. From Fig. 1,

Manuscript received 20 April 2023; revised 27 August 2023 and 29 November 2023; accepted 15 December 2023. Date of publication 20 December 2023; date of current version 23 April 2024. This work was supported in part by the Natural Science Foundation of China under Grant 61836016, Grant 62272485, and Grant 62076255; in part by the Project of Guangxi Science and Technology under Grant GuiKeAB23026040; in part by the National Natural Science Foundation of China under Grant 62020106013 and Grant 61972177; and in part by the Guangdong Key Laboratory of Data Security and Privacy Preserving under Grant 2017B030301004. This article was recommended by Associate Editor R. Wille. *(Corresponding authors: Jian Zhang; Jilian Zhang.)*

Jiaye Li and Jian Zhang are with the School of Computer Science and Engineering, Central South University, Changsha 410083, China (e-mail: lijiaye@csu.edu.cn; jianzhang@csu.edu.cn).

Jilian Zhang is with the School of Cyberspace Security, Jinan University, Guangzhou 510632, China (e-mail: zhangjilian@yeah.net).

Shichao Zhang is with the Guangxi Key Laboratory of Multisource Information Mining and Security, Guangxi Normal University, Guilin 541004, China (e-mail: zhangsc@csu.edu.cn).

Digital Object Identifier 10.1109/TCAD.2023.3345251

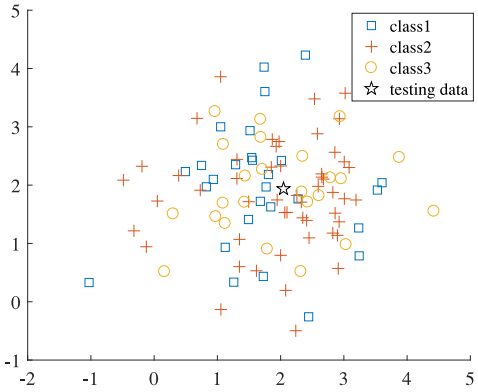



Fig. 1. Position distribution of a testing data and training data.

we can see that for the testing data, if different $K$ values are chosen, the predicted labels will be different, possibly any one of class1, class2, and class3. This indicates that the choice of $K$ value is crucial.

This article proposes a novel QKNN algorithm that can simultaneously handle the challenges of $K$ value selection and neighbor selection (including distance measurement and neighbor search). It can obtain the optimal $K$ value and corresponding $K$ nearest neighbors for each testing data through matrix computation. Specifically, the initial state $(|0\rangle|Yb_1\rangle|0\rangle|Yb_2\rangle, \ldots, |0\rangle|Yb_m\rangle)(|0\rangle^{\otimes n_0})(|0\rangle^m)$ of the entire quantum system is first prepared. And then, phase estimation and controlled rotation techniques are used to obtain the required quantum parameter values of the model (including phase of the eigenvalues and angle of the controlled rotation). Finally, combined with inverse phase estimation, a quantum circuit is constructed, which outputs the model $W$ (recording the optimal $K$ value and $K$ nearest neighbors for each testing data). In addition, this article validates the efficiency of the proposed algorithm on datasets using qiskit and MATLAB.

The main contributions of this article are as follows.

1) This article is the first to utilize quantum technology to optimize both $K$ value selection and neighbor selection in KNN simultaneously.
2) This article designs a new quantum circuit to quantization KNN. This circuit leverages the advantages of quantum computing to rapidly output the quantum state of the optimal $K$ value and the indices of $K$ nearest neighbor samples for each testing data. This further reduces the time complexity of the KNN algorithm.

A series of experiments were conducted to validate the efficiency of the proposed algorithm using IBM's qiskit and MATLAB. The experiments demonstrate that the proposed algorithm can output the quantum state of the optimal $K$ value and the indices of $K$ nearest neighbor samples for each testing data. In addition, compared with advanced comparative algorithms, the proposed algorithm also achieves better classification performance on public dataset.

The remainder of this article is organized as follows. Section II briefly reviews previous related work. Section III describes our proposed method in detail, quantum circuit, and time complexity analysis. Section IV shows the results of algorithm on dataset. Section V summarizes the full paper.

## II. Related Work

In this section, we introduce some work of classical KNN and QKNN.

### A. Classic KNN

KNN, as one of the top ten data mining algorithms, is widely used in classification tasks. It is a well-known representative of "lazy learning" and does not require any training process. Given a testing data, KNN can predict its label by calculating its $K$ nearest neighbors based on the measurement function. KNN has been further studied and explored by researchers due to its simple concept, ease of implementation, and practicality. These studies mainly focus on the improvement of $K$ value selection and neighbor selection.

*K Value Selection:* In KNN, the selection of $K$ value is crucial as different $K$ values can lead to different prediction results. If a smaller value of $K$ is chosen, the algorithm may struggle to generalize effectively, potentially leading to overfitting. Conversely, if a larger value of $K$ is selected, the model might become excessively generalized, resulting in less accurate predictions of data class labels. Therefore, selecting an appropriate $K$ value is the goal of researchers. Li et al. [11] set different $K$ values for different categories of data. Nair and Kashyap [12] choose the best $K$ value based on accuracy. Saputra et al. [13] determined the optimal $K$ value based on the local structure of the data and accuracy. Laksono et al. [14] optimized the $K$ value in KNN using frequency distribution clustering. Although these methods have improved the KNN algorithm, their $K$ values are fixed (i.e., the optimal $K$ value for different testing data or different categories of data is the same). Therefore, to further improve the performance of KNN, some researchers have learned different optimal $K$ values for different testing data or different categories of data. For example, Li used a tree structure to obtain an adaptive $K$ value [15]. Zhang et al. [16] learnt the optimal $K$ value of the testing data using sparse techniques. Gong and Liu dynamically obtained the optimal $K$ value using similarity calculation [17]. Ling et al. [18] proposed a new decision strategy based on probability and iterative $K$ values to improve the KNN algorithm. Gallego et al. [19] proposed an adaptive $K$ value strategy to improve KNN search performance. García-Pedrajas et al. [20] set the optimal $K$ value based on the local and global impact of different $K$ values.

In conclusion, the enhancement of KNN from the perspective of selecting the $K$ value has undergone a process of transitioning from a fixed optimal $K$ value to adapting different optimal $K$ values for distinct data points. However, in the process of researchers using models or algorithms to learn the optimal $K$ value, KNN no longer has the lazy learning characteristic, i.e., KNN has a training process (to obtain the optimal $K$ value during training). Moreover, the process of obtaining the optimal $K$ value is often time consuming. Therefore, in recent years, Zhang et al. [21] proposed the ktree and k*tree algorithms, whose core is to use the leaf nodes in the tree structure to save the optimal $K$ value of the training data and the data subset of the

potential $K$ nearest neighbors of the testing data. In this way, the optimal $K$ value and class label of each testing data can be obtained quickly (because it narrows down the search space of $K$ nearest neighbors). In addition, Zhang and Li also proposed the one-step KNN algorithm [22], which combines $K$ value selection and neighbor search into one-step matrix computation, thereby reducing the running cost of the algorithm.

*Neighbor Selection:* In KNN, when the $K$ value is provided as input, it only specifies the number of neighbors without determining which training data points are neighbors of the test data. Therefore, it is usually necessary to use a distance metric function to calculate the $K$ nearest neighbors that are similar to the testing data [23]. Traditional distance metric functions include Euclidean distance, Manhattan distance, Chebyshev distance, Minkowski distance, cosine distance, Hamming distance, etc. They can all be used to calculate the distance between the testing data and the training data, thus obtaining the $K$ nearest neighbors of the testing data. However, they all have some shortcomings. For example, Euclidean distance and Manhattan distance become ineffective as the data dimension increases. The scope of the Chebyshev distance is relatively small, and it only applies to specific scenarios. The Minkowski distance is a set of the first three distances, so its shortcomings include their set and the additional parameter $p$. The Cosine distance only considers the directional characteristics of the data and does not consider their magnitude. The Hamming distance cannot be applied to situations where the dimensions of two data sets are unequal.

Due to the limitations of traditional distance metrics in KNN, some researchers have proposed new distance functions to improve KNN. Rastin et al. [24] proposed a generalized prototype-weighted distance function to obtain neighbors, which makes the algorithm perform stably in class-imbalanced scenarios. Gou et al. [25] proposed a distance-weighted function to obtain neighbors. Liao et al. [26] used convolutional neural network to learn appropriate distance metrics and prototype reduction to find the nearest neighbors of testing data. Zhang et al. [27] proposed a mixed distance metric to improve the separability of samples. Syaliman et al. [28] proposed a new distance function using local means and distance weights to improve the majority rule. Liu improved the general distance metric using vertical shallow cuttings to improve KNN [29]. Lubis et al. [30] compared and optimized different distance functions in different situations, making KNN more effective. Gou et al. [31] designed a generalized average distance based on the local mean vector of each category to obtain neighbors.

For the neighbor selection, current work mainly focus on learning and constructing distance metrics. They mainly address issues, such as class imbalance, local structure of data, and accuracy. However, there are still some challenges to be addressed, such as the "time" dimension of data, where neighbors often change over time. For a testing data, the $K$ nearest neighbors at one moment may not be so at the next moment. To address this challenge, a series of distance metrics considering the time dimension need to be proposed.

### B. Quantum KNN

In recent years, with the development of quantum computing technology, quantum machine learning can further optimize traditional machine learning by leveraging the high parallelism of quantum computing [32] [33]. Many machine learning algorithms have been improved by combining with quantum computing technology, such as quantum principal components analysis, quantum support vector machine, QKNN, quantum feature selection, and quantum search algorithms [34]. Compared with traditional machine learning algorithms, they have achieved exponential or quadratic speedup [35]. It is worth mentioning the Grover search algorithm, which has achieved a qualitative leap compared to traditional search algorithms. This undoubtedly brings hope to many challenging problems that require searching in very large spaces and encourages researchers.

For the KNN algorithm, researchers have proposed some work on QKNN by combining quantum computing technology. For example, Wiebe et al. [2] proposed a quantum nearest-neighbor algorithm that uses a quantum technology to calculate the distance between testing data and training data. Compared with classical algorithms, in the worst case, it achieves polynomial acceleration. In some specific cases, it even achieves exponential acceleration. This method fully utilizes the advantages of quantum computing, but only improves the process of finding neighbors and does not consider the impact of $K$ values on the algorithm. Similarly, Ruan et al. [3] proposed a quantum circuit for Hamming distance to improve KNN. This method uses a quantum circuit to calculate the Hamming distance between testing data and training data, and sets a threshold to obtain $K$ nearest neighbors of the testing data. Subsequently, Dang et al. [6] proposed a QKNN algorithm and applied it to image classification. Specifically, it first uses classical methods to extract the feature vector of the image, which is nonquantum. Then, the feature vector obtained in the previous step is transformed into a quantum state and inputted into a quantum circuit. Finally, the quantum minimum search algorithm is used to obtain $K$ nearest neighbors of the testing data. This method can be seen as a combination of classical algorithms and quantum technology, and only quantizes a part of the algorithm.

Subsequently, Basheer et al. [5] proposed a QKNN algorithm, which encodes fidelity information between the testing state and all sequence states into the amplitudes of a quantum state, thereby performing measurement to obtain $K$ nearest neighbors. In recent years, several new QKNN algorithms have been proposed. For example, Zhang et al. [7] proposed an improved QKNN algorithm, which uses efficient encoding methods and amplitude estimation techniques to construct a weighted QKNN circuit, thereby improving similarity calculation. Tian and Baskiyar also proposed a QKNN method and applied it to fake news detection [8]. Gao et al. [9] proposed a QKNN algorithm, which uses a quantum subalgorithm to search for the minimum value in an unordered dataset, thereby obtaining $K$ nearest neighbors of the testing data. Li et al. [36] used quantum computing to obtain the Hamming distance between testing data and training data for KNN classification.

Li et al. [10] proposed a quantum k-fold cross-validation method to improve KNN. Specifically, it first calculates the Hamming distance between the testing data and training data. And then, it obtains the threshold $t$ of the nearest neighbor distance through quantum technology, thereby getting the $K$ nearest neighbors of the testing data. Finally, it uses the majority rule to classify the testing data. Feng et al. [4] proposed a QKNN algorithm based on the polar distance. The core of this method is to propose a new similarity measure function (i.e., polar distance). The $K$ nearest neighbors of the testing data are further found and classified them based on this distance measure.

In summary, we can see that in QKNN, the core breakthrough lies in obtaining the neighbor selection (i.e., distance measurement or neighbor search). Although they have achieved acceleration compared to classical KNN, it is mainly only a partial quantumization of the process, and they have not considered the influence of the $K$ value on the algorithm. In addition, they have not studied the selection of $K$ value using a quantum technology.

## III. Our Approach

### A. Notation

In this article, we use $X \in \mathbb{R}^{n\times d}$ to represent the training data and $Y \in \mathbb{R}^{m\times d}$ to represent the testing data, where $n$ and $m$, respectively, denote the number of training data and testing data, and $d$ represents the number of features. $W \in \mathbb{R}^{n\times m}$ represents the relationship matrix between the testing data and the training data, which records the optimal $K$ value and $K$ nearest training data for all testing data. $Y_j \in \mathbb{R}^{1\times d}$ represents the $j$th testing data. In addition, $\|W\|_1 = \sum_i \sum_j |w_{ij}|$ represents the $l_1-$norm of $W$, and $\|W\|_{2,p} = [\sum_{i=1}^{n} (\sum_{j=1}^{m} |W_{ij}|^2)^{p/2}]^{1/p}$ represents the $l_{2,p}-$norm of $W$, where $0 < p < 2$. $|0\rangle$ and $|1\rangle$, respectively, denote the quantum bit in the zero state and 1 state. $H$ represents the Hadamard gate. $FT$ and $FT^{-1}$, respectively, represent the quantum Fourier transform and its inverse transform.

### B. KNN With K Value Selection and Neighbor Selection

The conceptual similarity between the proposed objective function and the ideas presented in [21], [37], and [38] is that they are all based on the least squares loss function. The key distinction lies in the core of the proposed KNN objective function, which uniquely acquires both the $K$ value and $K$ nearest neighbors of the testing data through matrix calculations, a feature not shared by other methods. We first utilize the training data to reconstruct the testing data, in order to establish the relationship between the testing data and training data. This is shown in the following equation:

$$\min_{W} \sum_{j=1}^{m} \left\| X^T W_j - Y_j^T \right\|_2^2 = \min_{W} \left\| X^T W - Y^T \right\|_F^2 \tag{1}$$

where $X \in \mathbb{R}^{n\times d}$ represents the training data matrix, $Y \in \mathbb{R}^{m\times d}$ represents the testing data matrix, where $n$ and $m$, respectively, denote the number of training data and testing data, and $d$ represents the number of data features. $W \in \mathbb{R}^{n\times m}$ represents the reconstruction coefficient matrix between the testing data and training data. Equation (1) can be utilized to sequentially reconstruct each testing data by the training data, thereby obtaining the reconstruction coefficient matrix. However, two issues still remain.

1) During the reconstruction process, there may be fitting errors that can affect the accuracy of the obtained reconstruction matrix.
2) The reconstruction matrix is nonsparse, which prevents us from obtaining the $K$ value and corresponding $K$ nearest neighbors for each testing data.

Therefore, we further introduce a bias term to obtain the following equation:

$$\min_{W} \left\| X^T W + eb - Y^T \right\|_F^2 \tag{2}$$

where $e \in \mathbb{R}^{d\times 1}$ is a vector with all elements equal to 1 and $b \in \mathbb{R}^{1\times m}$ is the bias term which can be used to reduce errors during the reconstruction of training data for testing data. Clearly, (2) addresses only the issue of existing fitting errors, without resolving the challenge of obtaining $K$ values and $K$ nearest neighbors. Therefore, we further introduce the $\|W\|_1$ and $\|W\|_{2,p}$ regularization terms to achieve a dual sparsity effect on the matrix $W$. This can lead to a significant number of zero elements in the reconstruction coefficient matrix, thereby revealing the $K$ values and corresponding $K$ nearest neighbors for testing data. The objective function of our proposed KNN is shown as follows:

$$\min_{b,W} \left\| X^T W + eb - Y^T \right\|_F^2 + \lambda_1 \|W\|_1 + \lambda_2 \|W\|_{2,p}. \tag{3}$$

For (3), we use the alternating iteration method to optimize it [39]. When $W$ is fixed, we can obtain the following equation:

$$\min_{b} \left\| X^T W + eb - Y^T \right\|_F^2. \tag{4}$$

For (4), we can further rewrite it in the form of a trace, i.e., $\min_b tr((W^T X + b^T e^T - Y)(X^T W + eb - Y^T))$, and take the derivative of $b$ to obtain the following equation:

$$\begin{aligned} &\frac{\partial \left\| X^T W + eb - Y^T \right\|_F^2}{\partial b} = 0 \\ &\Rightarrow 2e^T X^T W + 2e^T eb - 2e^T Y^T = 0. \end{aligned} \tag{5}$$

According to (5), the solution of $b$ can be further obtained as follows:

$$b = \frac{1}{d} e^T Y^T - \frac{1}{d} e^T X^T W. \tag{6}$$

Similarly, when $b$ is fixed, (3) can be further written as follows:

$$\begin{aligned} \min_{W} &tr\big(\big(W^T X + b^T e^T - Y\big)\big(X^T W + eb - Y^T\big)\big) \\ &+ \lambda_1 \|W\|_1 + \lambda_2 tr\big(W^T Q W\big). \end{aligned} \tag{7}$$

By making $f(W) = tr((W^T X + b^T e^T - Y)(X^T W + eb - Y^T)) + \lambda_1 \|W\|_1 + \lambda_2 tr(W^T Q W)$ and taking the derivative of each column $W_j$ of $W$, the following equation can be obtained:

$$\begin{aligned} &\frac{\partial f(W)}{W_j} = 0 \\ &\Rightarrow 2XX^T W_j + 2Xeb_j - 2X\big(Y^T\big)_j \\ &\quad + 2\lambda_1 D_j W_j + 2\lambda_2 Q W_j = 0. \end{aligned} \tag{8}$$

Furthermore, according to (8), we can obtain the solution for each column $W_j$ of $W$ as follows:

$$W_j = \left(XX^T + \lambda_1 D_j + \lambda_2 Q\right)^{-1}\left(X(Y^T)_j - Xeb_j\right) \tag{9}$$

where $D_j$ and $Q$ are both diagonal matrices, and their diagonal elements are

$$D_j^{ii} = \frac{1}{2\left|W_{ij}\right| + \varsigma}, \quad i = (1, 2, \ldots, n) \tag{10}$$

$$Q^{ii} = \frac{1}{\frac{2}{p}\left\|W^i\right\|_2^{2-p} + \varsigma}, \quad i = (1, 2, \ldots, n) \tag{11}$$

where $\varsigma$ is a small enough constant. It should be noted that the above solving process is an alternating iteration process. i.e., When solving the $b$ of the $(t+1)$th iteration, the $W$ value of the $t$-th iteration will be used. When solving $W$ for the $(t+1)$th iteration, the value of $b$ in the $t$-th iteration needs to be used. The final solution for the model $W$ is obtained only when the algorithm converges. The variable $W$ records the optimal $K$ value for all testing data and the indices of its $K$ nearest neighbor samples.

To facilitate the reader's understanding, we introduce an example. Suppose the final value of the variable $W$ that we obtain is

$$W = \begin{bmatrix} 0.2 & 0 & 0.1 & 0 \\ 0 & 0.4 & 0 & 0.5 \\ 0 & 0 & 0.3 & 0 \\ 0.6 & 1.1 & 0.8 & 0 \\ 0 & 0.9 & 0.7 & 0 \end{bmatrix}. \tag{12}$$

As shown in (12), W has five rows and four columns, indicating that it records the relationships between four testing data and five training data. For the first column of W, it corresponds to the relationship between the first testing data and the five training data. We can observe that in the first column of W, the elements corresponding to the 1st and 4th training data are nonzero, indicating that the optimal $K$ value for the first testing data is 2, with the two nearest neighbors being the 1st and 4th training data. Similarly, we can determine that the optimal $K$ value for the second testing data is 3, with the corresponding 3 nearest neighbors being the 2nd, 4th, and 5th training data. For the third testing data, its optimal $K$ value is 4, and the corresponding four nearest neighbors are the 1st, 3rd, 4th, and 5th training data. Finally, for the fourth testing data, its optimal $K$ value is 1, with the nearest neighbor being the 2nd training data.

### C. Quantum KNN With K Value Selection and Neighbor Selection

In the previous section, we obtained the final model $W$ through (3) and alternating iterations. However, the time complexity of this process is relatively high. Therefore, in this section, we propose its quantum version. First, we perform reduced singular value decomposition on the training data matrix $X$, as shown in the following equation:

$$X = U\Sigma V = \sum_{i=1}^{r} \sigma_i u_i v_i^T \tag{13}$$

where $r \leq \min(n, d)$ represents the rank of matrix X, $U = \{u_i\}_{i=1}^r \in \mathbb{R}^{n\times r}$ and $V = \{v_i\}_{i=1}^r \in \mathbb{R}^{d\times r}$ are composed of left and right singular value vectors, and $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$. In order to facilitate the quantization of the algorithm, we convert (13) into the dirac symbolic form, i.e., $X = \sum_{i=1}^r \sigma_i |u_i\rangle\langle v_i|$. According to (9) and (13), we can obtain the following theorem.

*Theorem 1:* Given $X = \sum_{i=1}^r \sigma_i |u_i\rangle\langle v_i|$, the quantum state of model $W_j$ can be obtained as follows:

$$\left|W_j\right\rangle = \sum_{i=1}^{r} \frac{\sigma_i \beta_i \left\|Yb_j\right\|}{\sigma_i^2 + \lambda_1 D_j^{ii} + \lambda_2 Q^{ii}} |u_i\rangle \tag{14}$$

where $Yb_j = (Y^T)_j - eb_j = \sum_{i=1}^r \beta_i \|Yb_j\| |v_i\rangle$ is the linear combination of $Yb_j$ on the basis vector $|v_i\rangle$.

*Proof:* According to the dirac symbolic form of (13), i.e., $X = \sum_{i=1}^r \sigma_i |u_i\rangle\langle v_i|$, we can obtain $X^T = \sum_{i=1}^r \sigma_i |v_i\rangle\langle u_i|$. Furthermore, the following equation can be obtained:

$$\begin{aligned} XX^T &= \left(\sum_{i=1}^{r} \sigma_i |u_i\rangle\langle v_i|\right)\left(\sum_{i=1}^{r} \sigma_i |v_i\rangle\langle u_i|\right) \\ &= \sum_{i=1}^{r}\sum_{i=1}^{r} \sigma_i\sigma_i |u_i\rangle\langle v_i||v_i\rangle\langle u_i| \\ &= \sum_{i=1}^{r} \sigma_i^2 |u_i\rangle\langle u_i|. \end{aligned} \tag{15}$$

Because $Yb_j$ can be written as a linear combination on the basis vector $|v_i\rangle$, i.e., $Yb_j = \sum_{i=1}^r \beta_i \|Yb_j\| |v_i\rangle$. Therefore, the following equation can be obtained:

$$\begin{aligned} XYb_j &= \sum_{i=1}^{r} \sigma_i |u_i\rangle\langle v_i| \sum_{i=1}^{r} \beta_i \left\|Yb_j\right\| |v_i\rangle \\ &= \sum_{i=1}^{r} \sigma_i \beta_i \left\|Yb_j\right\| |u_i\rangle\langle v_i||v_i\rangle \\ &= \sum_{i=1}^{r} \sigma_i \beta_i \left\|Yb_j\right\| |u_i\rangle. \end{aligned} \tag{16}$$

By combining (9), (15), and (16), the following equation can be further obtained:

$$\begin{aligned} W_j &= \left(XX^T + \lambda_1 D_j + \lambda_2 Q\right)^{-1}\left(XYb_j\right) \\ &= \left\{ \begin{array}{l} \sum_{i=1}^r \sigma_i^2 |u_i\rangle\langle u_i| + \lambda_1 \sum_{i=1}^r D_j^{ii} |u_i\rangle\langle u_i| \\ +\lambda_2 \sum_{i=1}^r Q^{ii} |u_i\rangle\langle u_i| \end{array} \right\}^{-1} \\ &\quad \left[\sum_{i=1}^{r} \sigma_i \beta_i \left\|Yb_j\right\| |u_i\rangle\right] \\ &= \left\{\sum_{i=1}^{r} \left(\sigma_i^2 + \lambda_1 D_j^{ii} + \lambda_2 Q^{ii}\right) |u_i\rangle\langle u_i|\right\}^{-1} \\ &\quad \left[\sum_{i=1}^{r} \sigma_i \beta_i \left\|Yb_j\right\| |u_i\rangle\right] \\ &= \sum_{i=1}^{r}\sum_{i=1}^{r} \left(\sigma_i^2 + \lambda_1 D_j^{ii} + \lambda_2 Q^{ii}\right)^{-1} \sigma_i \beta_i \left\|Yb_j\right\| |u_i\rangle\langle u_i||u_i\rangle \\ &= \sum_{i=1}^{r} \frac{\sigma_i \beta_i \left\|Yb_j\right\|}{\sigma_i^2 + \lambda_1 D_j^{ii} + \lambda_2 Q^{ii}} |u_i\rangle. \end{aligned} \tag{17}$$

■

Next, inspired by the HHL algorithm [40], the proposed QKNN algorithm is divided into five steps. The detailed process is shown in the following.

1) Preparation of quantum initial states. First, we prepare three types of quantum registers, as shown in Fig. 3. From bottom to top, they are the first, second, and third types of quantum registers, corresponding to $Yb_j$, eigenvalues, and auxiliary quantum bits, respectively. We set their initial state to $(|0\rangle|Yb_1\rangle|0\rangle|Yb_2\rangle, \ldots, |0\rangle|Yb_m\rangle)(|0\rangle^{\otimes n_0})(|0\rangle^m)$. The preparation of $|0\rangle|Yb_1\rangle|0\rangle|Yb_2\rangle, \ldots, |0\rangle|Yb_m\rangle$ can refer to the method in [41], where $n_0$ represents the number of quantum bits to store the eigenvalues, and $m$ represents the number of quantum auxiliary bits, corresponding to the $m$ columns of $W$ where

$$\begin{aligned}&|0\rangle|Yb_1\rangle|0\rangle|Yb_2\rangle, \ldots, |0\rangle|Yb_m\rangle\\&= |0\rangle\left(\sum_{i=1}^{r}\beta_{1i}|v_i\rangle\right)|0\rangle\left(\sum_{i=1}^{r}\beta_{2i}|v_i\rangle\right), \ldots, |0\rangle\left(\sum_{i=1}^{r}\beta_{mi}|v_i\rangle\right).\end{aligned} \tag{18}$$

2) We convert the Hermitian matrix $X$ into a unitary operation $e^{iXt}$. If matrix $X$ is not Hermitian, we restructure it through the following formula:

$$\tilde{X} = \begin{bmatrix} 0 & X \\ X^T & 0 \end{bmatrix} \in R^{(n+d)\times(n+d)}. \tag{19}$$

At this point, the corresponding $W_j$ and $Yb_j$ are filled with the corresponding zero elements, namely, $\tilde{W}_j = \begin{pmatrix} 0 \\ W_j \end{pmatrix}$ and $\tilde{Yb}_j = \begin{pmatrix} Yb_j \\ 0 \end{pmatrix}$. For the preparation of $e^{iXt}$, if $X$ is a sparse matrix, it can be simulated by the method in [40]. Its time complexity is $O(s^4\kappa\varepsilon^{-1}\log_2(n))$, where $\kappa$ is the condition number of $X$, $s$ is the sparsity, and $\varepsilon$ is the error. If $X$ is not a sparse matrix, we can use the methods in [42] and [43] to effectively simulate. As this part of the work is not the focus of this article, we will not provide a detailed introduction.

3) *Phase Estimation:* We use phase estimation to transfer all the integer forms of the eigenvalues of $X$ to the basis vector of the stored bits. At this point, the quantum state of the entire system is

$$\begin{aligned}&\sum_{i=1}^{r}\beta_{1i}(|v_i\rangle|\sigma_i\rangle)|0\rangle\\&\quad\sum_{i=1}^{r}\beta_{2i}(|v_i\rangle|\sigma_i\rangle)|0\rangle, \ldots, \sum_{i=1}^{r}\beta_{mi}(|v_i\rangle|\sigma_i\rangle)|0\rangle.\end{aligned} \tag{20}$$

Since our model $W$ is a matrix, we perform quantum calculations to obtain the quantum states of each column. There are a total of $m$ columns, so $m$ phase estimation operations are required.

4) *Controlled Rotation:* This process is the core of the algorithm, and each rotation angle affects the solution of $W_j$. Specifically, we control the qubits of the third register through $|(\pm\sigma_i)/(n+d)\rangle$ and rotate them from $|0\rangle$ to $\sqrt{1 - C_1^2 h_1^2(\pm\sigma_i, \lambda_1, \lambda_2)}|0\rangle + C_1 h_1(\pm\sigma_i, \lambda_1, \lambda_2)|1\rangle$. where $C_1 = O(\max_{\lambda_i} h_1(\sigma_i, \lambda_1, \lambda_2))^{-1}$, $h_1(\sigma, \lambda_1, \lambda_2) \coloneqq ((n+d)\sigma)/(\sigma^2 + \lambda_1 D_j^{ii} + \lambda_2 Q^{ii})$. At this point, the quantum state of the entire system is

$$\begin{cases} \sum_{i=1}^{r}\beta_{1i}|v_i\rangle|\sigma_i\rangle\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_1^{ii}+\lambda_2 Q^{ii}}\right)|1\rangle \\ +\sqrt{1-\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_1^{ii}+\lambda_2 Q^{ii}}\right)^2}|0\rangle \\ \sum_{i=1}^{r}\beta_{2i}|v_i\rangle|\sigma_i\rangle\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_2^{ii}+\lambda_2 Q^{ii}}\right)|1\rangle \\ +\sqrt{1-\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_2^{ii}+\lambda_2 Q^{ii}}\right)^2}|0\rangle \\ \vdots \\ \sum_{i=1}^{r}\beta_{mi}|v_i\rangle|\sigma_i\rangle\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_3^{ii}+\lambda_2 Q^{ii}}\right)|1\rangle \\ +\sqrt{1-\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_3^{ii}+\lambda_2 Q^{ii}}\right)^2}|0\rangle \end{cases} \tag{21}$$

where the controlled rotation angle is

$$\begin{cases} \vartheta(\sigma_i) = \arcsin\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_1^{ii}+\lambda_2 Q^{ii}}\right) \\ \vartheta(\sigma_i) = \arcsin\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_2^{ii}+\lambda_2 Q^{ii}}\right) \\ \vdots \\ \vartheta(\sigma_i) = \arcsin\left(\frac{C_1(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_m^{ii}+\lambda_2 Q^{ii}}\right). \end{cases} \tag{22}$$

At this stage, all eigenvalues associated with quantum states are acquired, and these eigenvalues are converted from the stored bit basis vector to amplitudes.

5) *Inverse Phase Estimation and Measurement:* At this stage, we perform inverse phase estimation and auxiliary bit measurement, i.e., the eigenvalues on the amplitude of the stored bit are merged onto the $Yb_j$ bit, and when the ancillary qubit is measured to be in the specific state (i.e., $|1\rangle$), the quantum state of the solution can be obtained on the $Yb_j$ bit. At this stage, the quantum state of the entire system is

$$\begin{cases} |1\rangle\left(\sum_{i=1}^{r}\frac{C_1\beta_{1i}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_1^{ii}+\lambda_2 Q^{ii}}|v_i\rangle\right)\Big/\sqrt{\sum_{i=1}^{r}\left(\frac{C_1\beta_{1i}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_1^{ii}+\lambda_2 Q^{ii}}\right)^2} \\ |1\rangle\left(\sum_{i=1}^{r}\frac{C_1\beta_{2i}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_2^{ii}+\lambda_2 Q^{ii}}|v_i\rangle\right)\Big/\sqrt{\sum_{i=1}^{r}\left(\frac{C_1\beta_{2i}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_2^{ii}+\lambda_2 Q^{ii}}\right)^2} \\ \vdots \\ |1\rangle\left(\sum_{i=1}^{r}\frac{C_1\beta_{mi}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_m^{ii}+\lambda_2 Q^{ii}}|v_i\rangle\right)\Big/\sqrt{\sum_{i=1}^{r}\left(\frac{C_1\beta_{mi}(n+d)\sigma_i}{\sigma_i^2+\lambda_1 D_m^{ii}+\lambda_2 Q^{ii}}\right)^2}. \end{cases} \tag{23}$$

At this point, we only need to remove $|1\rangle$ to obtain the quantum state $|W_j\rangle$ of each column of $W$ (Here, represents the $j$th column, out of a total of $m$ columns.).

To facilitate the understanding of the algorithm's process, we present the pseudocode of Algorithm 1. The input of the algorithm includes the training data $X$, testing data $Y$, and tunable hyperparameters $\lambda_1$ and $\lambda_2$. The output is each column of $W$ and its corresponding quantum state $|W_j\rangle$. The first

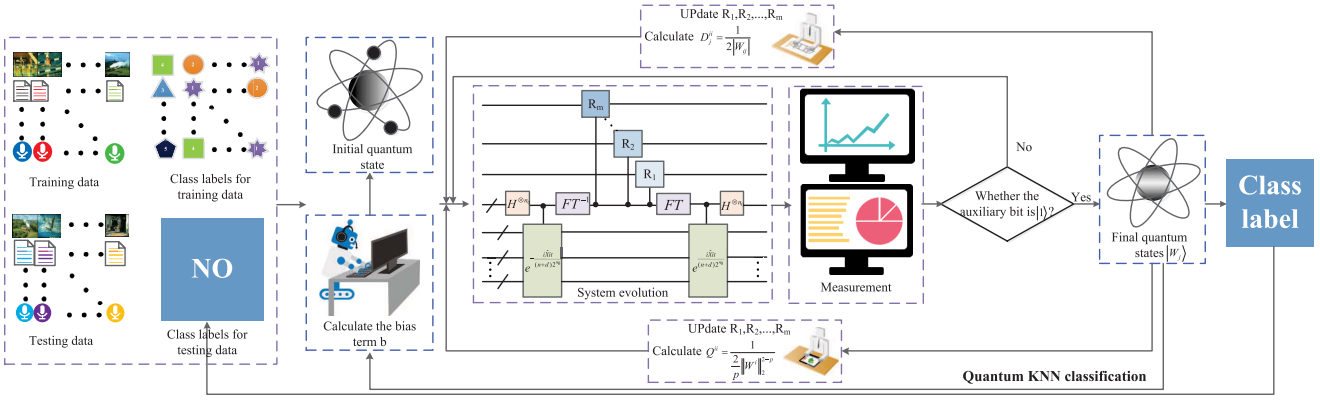



Fig. 2. Flowchart of the proposed QKNN.

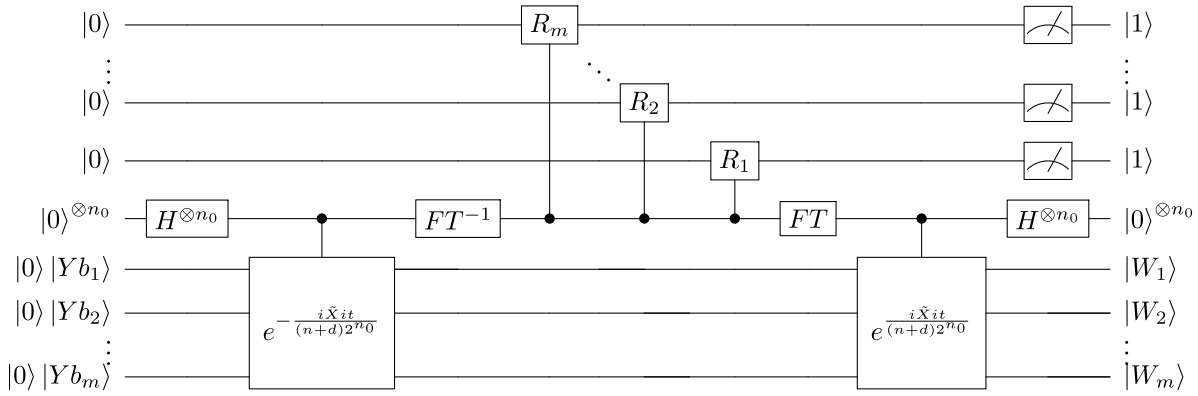



Fig. 3. Quantum circuit for solving $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$. From bottom to top, a total of three types of quantum registers are involved. The initial quantum states of the first type of registers are denoted as $|0\rangle|Yb_1\rangle|0\rangle|Yb_2\rangle, \ldots, |0\rangle|Yb_m\rangle$. The initial quantum states of the second type of quantum registers are denoted as $|0\rangle^{\otimes n_0}$. The initial quantum states of the third type of quantum registers are denoted as $|0\rangle^m$. After inputting the quantum initial states of the three quantum registers, from left to right, the phases estimation, controlled rotation, inverse phase estimation, and measurement operations are sequentially performed. When the auxiliary qubits in the third type of quantum registers are measured as $|1\rangle$, quantum states $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$ can be obtained. Here, $H$ represents the Hadamard gate. $R_1, R_2, \ldots, R_m$ is the controlled rotation operations corresponding to $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$, respectively. $FT$ and $FT^{-1}$ represent the quantum Fourier transform and its inverse, respectively. $n_0$ represents the number of quantum bits required to store eigenvalues.

and second steps are variable initialization, and $t$ denotes the number of iterations. Steps 4–7 compute the values of $b$, $D_j$, $Q$, and $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$. Step 9 indicates that the algorithm has converged, and the final values of variables $b$, $D_j$, $Q$, and $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$ are obtained when the convergence condition $(|\text{obj}(t+1) - \text{obj}(t)|)/(\text{obj}(t)) \leq 10^{-5}$ is met, where obj($t$) and obj($t+1$) represent the values of the objective function in the $t$-th and $(t+1)$th iterations, respectively. The specific process is shown in Figs. 2 and 3.

### D. Time Complexity Analysis

As described in Section I, the time complexity of current KNN algorithms mainly focuses on $K$ value selection and neighbor selection. In classical KNN algorithms, their time complexity is $O(mnd)$, where $m$, $n$, and $d$ represent the number of testing data, the number of training data, and the number of features in the data, respectively. Assuming $d = O(n)$, the time complexity of classical KNN can be further simplified to $O(mn^2)$. The time complexity of the proposed QKNN is mainly composed of phase estimation, amplitude amplification, and the number of iterations. According to [44], we can obtain that the time complexity of phase estimation is: $O(\|X\|^2_{\max}\text{poly}\log(n+d)\kappa^2/\varepsilon^3)$. By adding the time complexity of amplitude amplification and iteration, we can know that the time complexity of $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$ is $O(t\|X\|^2_{\max}\text{poly}\log(n+d)\kappa^3/\varepsilon^3 m)$, where $t$ is the number

**Algorithm 1:** Pseudo Code for Proposed Method

**Input**: Training data $\mathbf{X} \in \mathbb{R}^{n\times d}$, labels $l_x$ of the training data, testing data $\mathbf{Y} \in \mathbb{R}^{m\times d}$, adjustable parameters $\lambda_1$ and $\lambda_2$;;

**Output**: $W$ and $|W_1\rangle, |W_2\rangle, \ldots, |W_m\rangle$;

1 Initialize $t = 0$;
2 Random initialization matrix $W$;
3 **repeat**
4 Update $b$ according to Eq. (6);
5 Compute $D_j$ via $D_j^{ii} = \frac{1}{2|W_{ij}|+\varsigma} i = (1, 2, \ldots, n)$;
6 Compute $Q$ via $Q^{ii} = \frac{1}{\frac{2}{p}\|W^i\|_2^{2-p}+\varsigma} i = (1, 2, \ldots, n)$;
7 The quantum state of $|W_j\rangle$ is obtained by Fig. 3 *i.e.,* Eq. (14);
8 $t = t+1$ ;
9 **until** *convergence*;

TABLE I
TIME COMPLEXITY COMPARISON BETWEEN PROPOSED QKNN AND NON-QKNN

| Condition | non-quantum KNN | quantum KNN |
|---|---|---|
| $\kappa = O(\sqrt{n})$ | $O(mn^2)$ | $O(poly\log(n)n^{3/2}m)$ |
| $\kappa = poly\log(n)$ | $O(mn^2)$ | $O(poly\log(n)m)$ |

of iterations and $\kappa$ is the number of conditions (i.e., the ratio of the maximum singular value of $X$). $\varepsilon$ is the error term. Assuming $d = O(n)$, $t = O(1)$, $\|X\|_{\max} = O(1)$, and $1/\varepsilon = O(\text{poly}\log n)$, the time complexity of the proposed algorithm can be further written as $O(\text{poly}\log(n)\kappa^3 m)$. When $\kappa$ satisfies different situations, the time complexity comparison between the proposed QKNN algorithm and the non-QKNN algorithm is shown in Table I.

As shown in Table I, we can observe that the proposed algorithm can further achieve exponential acceleration on the basis of non-QKNN.

## IV. EXPERIMENT

In this section, we conducted a series of experiments using qiskit and MATLAB programming to, respectively, output the quantum and classical results of the proposed algorithm, thereby verifying the efficiency of the proposed algorithm.

### A. Experimental Setup

Due to the limitations of quantum computer hardware, we use the dataset in [45] and the public dataset Banknote[1] to verify the performance of the proposed quantum KNN. Specifically, we set training data $X = 1/4\begin{bmatrix} 15 & 9 & 5 & -3 \\ 9 & 15 & 3 & -5 \\ 5 & 3 & 15 & -9 \\ -3 & -5 & -9 & 15 \end{bmatrix}$, label $l_x = [1; 1; 2; 2]$ for training data, and testing data $Y = \begin{bmatrix} 14 & 8 & 5 & -3 \\ 8 & 14 & 3 & -5 \\ 4 & 4 & 15 & -9 \\ -4 & -4 & -9 & 15 \end{bmatrix}$. The label for testing data is the same as the label for training data, i.e., $l_y = [1; 1; 2; 2]$. Parameters $\lambda_1 = 1$ and $\lambda_2 = 1$. Due to the fact that the eigenvalues of training data X are 1, 2, 4, and 8. The eigenvectors corresponding to the eigenvalues are $u_1 = \begin{bmatrix} 0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}$, $u_2 = \begin{bmatrix} -0.5 \\ 0.5 \\ -0.5 \\ -0.5 \end{bmatrix}$, $u_3 = \begin{bmatrix} 0.5 \\ 0.5 \\ -0.5 \\ 0.5 \end{bmatrix}$, and $u_4 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ 0.5 \end{bmatrix}$. Therefore, we set each $y$ register to contain two qubits with initial quantum states of $|Yb_1\rangle$, $|Yb_2\rangle$, $|Yb_3\rangle$, and $|Yb_4\rangle$. The intermediate quantum register requires four qubits with initial quantum states of $|0\rangle|0\rangle|0\rangle|0\rangle$. In addition, we also need an auxiliary register with a quantum initial state of $|0\rangle$.

[1] http://archive.ics.uci.edu/ml

After completing these settings, we conducted the following four parts of experiments.

1) For each testing data, we output the quantum and nonquantum results for the first eight iterations of the algorithm. From these results, we determine the optimal $K$ value and the corresponding $K$ nearest neighbors for each testing data.
2) We conduct convergence experiments to verify the convergence of the algorithm.
3) We output the quantum circuit diagram for the algorithm in the final iteration. Additionally, we also output the controlled rotation angles for each iteration of the algorithm to aid readers in understanding.
4) Based on the output results from the quantum circuit and classical computation, we obtain the predicted label for each testing data. In addition, we compared the proposed algorithm with the QKNN in [46] on the Banknote dataset, thereby verifying the classification performance of the algorithm.

It is worth noting that the proposed algorithm had already converged by the 8th iteration. Therefore, all of our experiments only present the results from the first eight iterations.

### B. Analysis of Quantum and Classical Output Results

Figs. 4–7 show the quantum output results of the first to fourth testing data for the first eight iterations, respectively. Fig. 9 shows the classical results of all testing data for the first eight iterations. As we only measured three quantum bits, only the results corresponding to these three qubits are shown in Figs. 4–7, where two quantum bits correspond to each testing data and the other qubit is an ancillary qubit. In addition, it is only considered when the auxiliary bit is $|1\rangle$. From Fig. 4, in the first iteration, we can see that the ratio of the measurement probability is $|00\rangle_{y_1}|1\rangle_a : |01\rangle_{y_1}|1\rangle_a : |10\rangle_{y_1}|1\rangle_a : |11\rangle_{y_1}|1\rangle_a = 0.145 : 0.120 : 0.079 : 0.086$. In the classic output result (i.e., Fig. 9), the output of the first iteration is $[2.3361; -0.2541; -0.9744; -1.6464]$. This indicates that for the first testing data, both quantum and nonquantum results were given the maximum weight to the first training data in the first iteration. Unfortunately, due to the noise present in the quantum circuit and errors in the Hamiltonian simulation process, there is a slight deviation between the quantum results and classical results. However, this does not affect our ability to obtain the $K$ nearest neighbors of the data and the prediction accuracy. Similarly, for the first testing data, after the fifth iteration, its optimal $K$ value was obtained to be 1, and the nearest neighbor data was the first training data. From Fig. 4, it can also be seen that after the fifth iteration, the probability of $|00\rangle_{y_1}|1\rangle_a$ is the highest, relative to $|01\rangle_{y_1}|1\rangle_a$, $|10\rangle_{y_1}|1\rangle_a$, and $|11\rangle_{y_1}|1\rangle_a$. From Fig. 5, for the second testing data, in the first iteration, we can see that the ratio of the measurement probability is $|00\rangle_{y_1}|1\rangle_a : |01\rangle_{y_1}|1\rangle_a : |10\rangle_{y_1}|1\rangle_a : |11\rangle_{y_1}|1\rangle_a = 0.041 : 0.065 : 0.024 : 0.046$. In classical calculations, the first output result of the second testing data is $[0.1470; 1.6382; -0.5731; -1.1893]$. Starting from the sixth iteration, it is determined that the optimal $K$ value of the second testing data is 1, and the nearest neighbor is the second

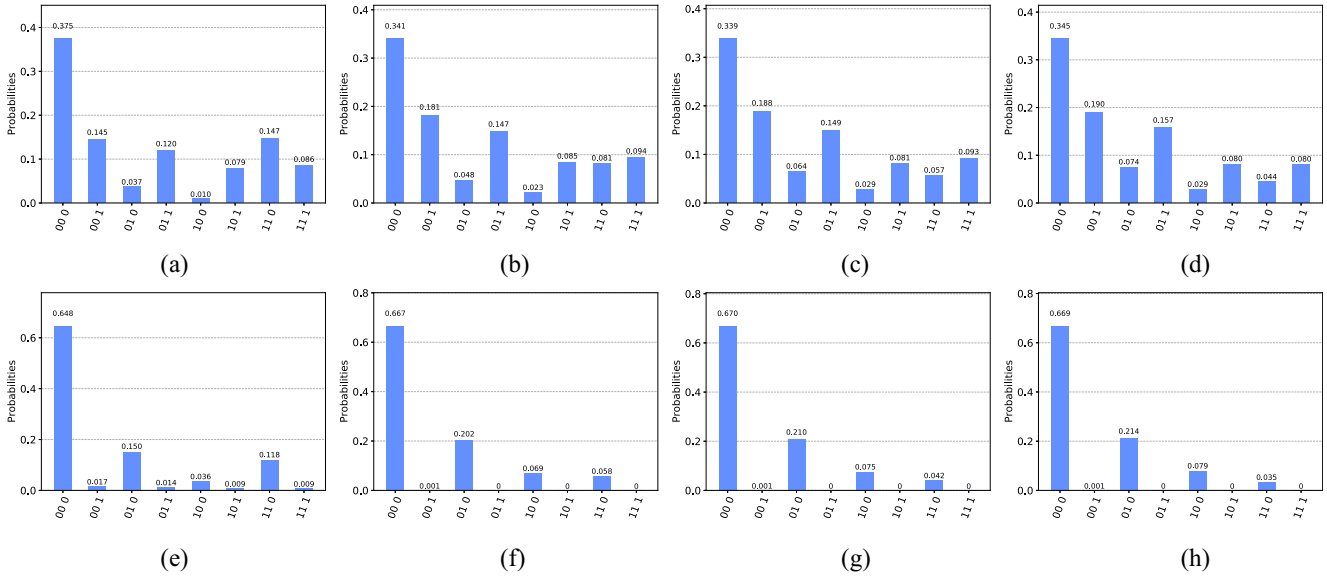



Fig. 4. Measurement results of $|W_1\rangle$ at each iteration. (a) $iter = 1$. (b) $iter = 2$. (c) $iter = 3$. (d) $iter = 4$. (e) $iter = 5$. (f) $iter = 6$. (g) $iter = 7$. (h) $iter = 8$.

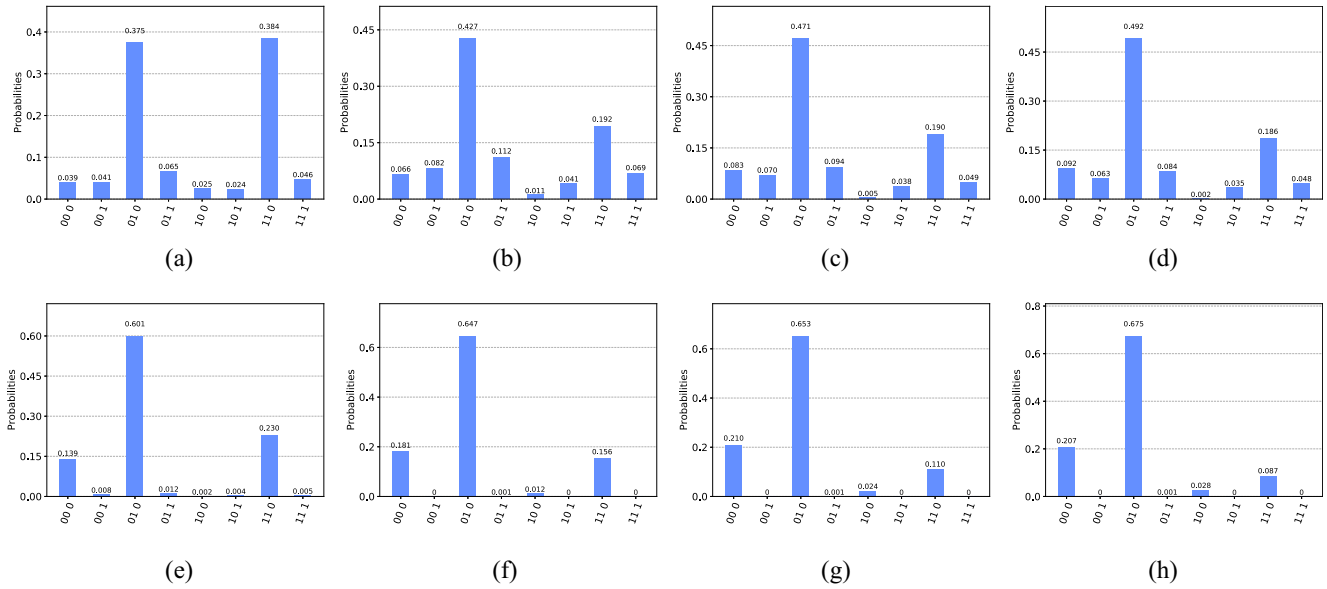



Fig. 5. Measurement results of $|W_2\rangle$ at each iteration. (a) $iter = 1$. (b) $iter = 2$. (c) $iter = 3$. (d) $iter = 4$. (e) $iter = 5$. (f) $iter = 6$. (g) $iter = 7$. (h) $iter = 8$.

TABLE II
CONTROLLED ROTATION ANGLE CORRESPONDING TO THE FIRST TESTING DATA

| iteration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Qubit1 | 0.8424 | 1.5541 | 1.6904 | 1.7348 | 0.1745 | 0.1744 | 0.1775 | 0.1790 |
| Qubit2 | 0.8014 | 0.6540 | 0.8042 | 0.8267 | 0.1624 | $2.3842\times10^{-8}$ | $2.384\times*10^{-8}$ | $2.3842\times10^{-8}$ |
| Qubit3 | 0.4640 | 0.4835 | 0.4875 | 0.4866 | 0.1614 | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ |
| Qubit4 | 0.2475 | 0.2491 | 0.2492 | 0.2491 | 0.1659 | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ |

training data. Fig. 6 shows the quantum output results of the first eight iterations of the third testing data. From Fig. 6, in the first iteration, we can see that the ratio of the measurement probability is $|00\rangle_{y_1}|1\rangle_a : |01\rangle_{y_1}|1\rangle_a : |10\rangle_{y_1}|1\rangle_a : |11\rangle_{y_1}|1\rangle_a = 0.029 : 0.024 : 0.100 : 0.091$. The corresponding classical result is $[-0.2131; -0.3674; 2.1429; -1.8633]$. The optimal $K$ value for the third testing data can be obtained as 1, with the third training data as its nearest neighbor. Similarly, according to Figs. 7 and 9, the optimal $K$ value for the fourth testing data is 1, with the fourth training data as its nearest neighbor.

Table VI shows a comparison between the proposed algorithm and the state-of-the-art QKNN algorithms. From

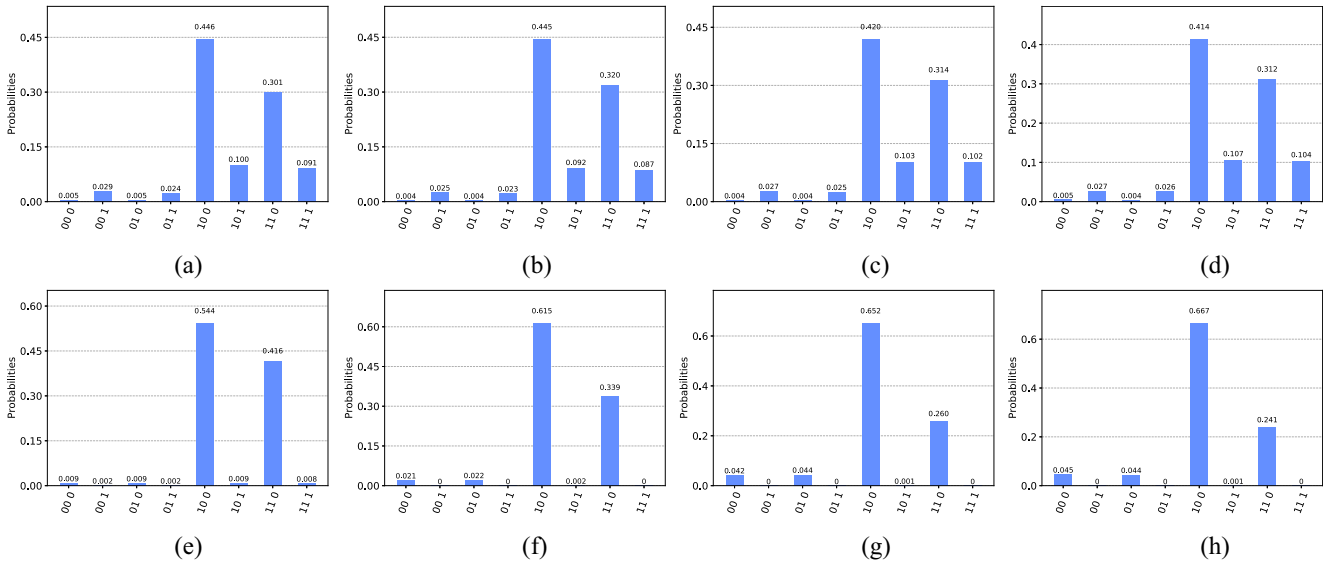



Fig. 6. Measurement results of $|W_3\rangle$ at each iteration. (a) $iter = 1$. (b) $iter = 2$. (c) $iter = 3$. (d) $iter = 4$. (e) $iter = 5$. (f) $iter = 6$. (g) $iter = 7$. (h) $iter = 8$.

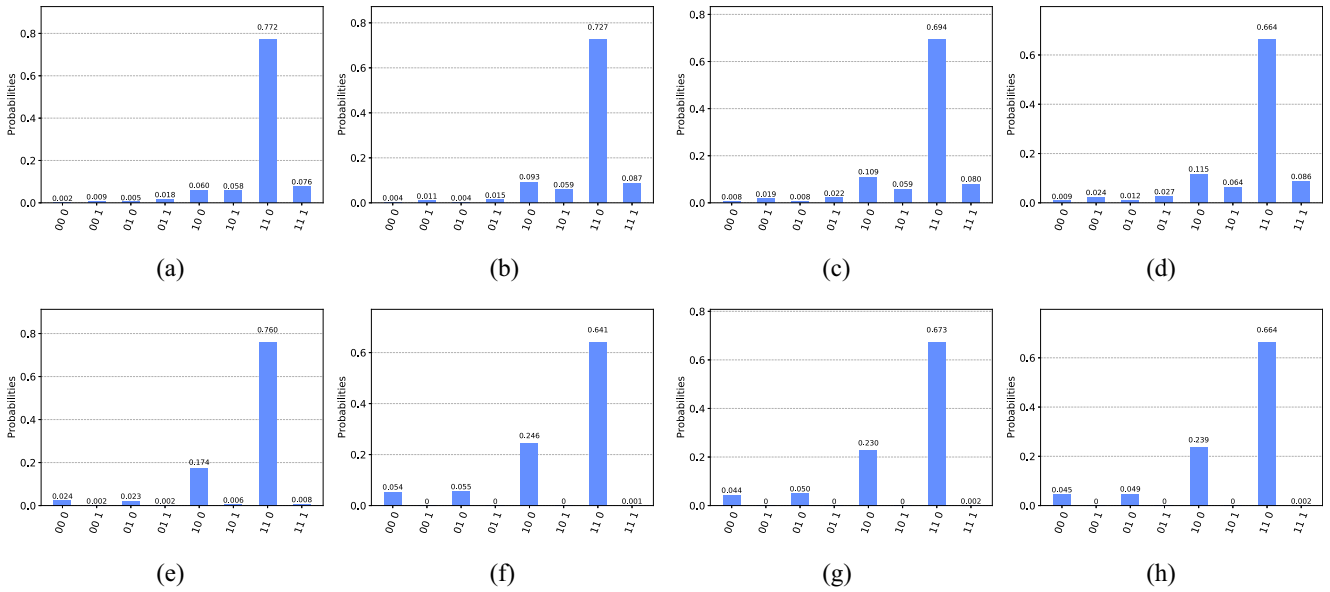



Fig. 7. Measurement results of $|W_4\rangle$ at each iteration. (a) $iter = 1$. (b) $iter = 2$. (c) $iter = 3$. (d) $iter = 4$. (e) $iter = 5$. (f) $iter = 6$. (g) $iter = 7$. (h) $iter = 8$.

TABLE III
CONTROLLED ROTATION ANGLE CORRESPONDING TO THE SECOND TESTING DATA

| iteration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Qubit1 | 0.5307 | 0.4371 | 0.1045 | 0.0011 | $3.0858\times10^{-6}$ | $1.1355\times10^{-8}$ | $5.9605\times10^{-9}$ | $5.9605\times10^{-9}$ |
| Qubit2 | 0.4402 | 0.9073 | 0.9486 | 0.9516 | 0.1905 | 0.1903 | 0.1935 | 0.1945 |
| Qubit3 | 0.3833 | 0.4731 | 0.4893 | 0.4896 | 0.1629 | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ |
| Qubit4 | 0.2221 | 0.2487 | 0.2493 | 0.2493 | 0.1661 | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ |

Table VI, we can see that the proposed quantum algorithm can simultaneously perform $K$ value selection and neighbor selection. Unlike existing QKNN algorithms, which use quantum search algorithms or distance metric quantization for neighbor selection, our proposed QKNN algorithm includes both $K$ value selection and neighbor selection in one-step quantum computation, achieving exponential speedup. Although Wiebe et al.'s algorithm can also achieve exponential speedup under certain conditions, its acceleration is only relative to the neighbor selection process and does not include $K$ value selection. Therefore, our proposed QKNN algorithm is a significant improvement as it combines both $K$ value selection and neighbor selection and achieves exponential speedup.

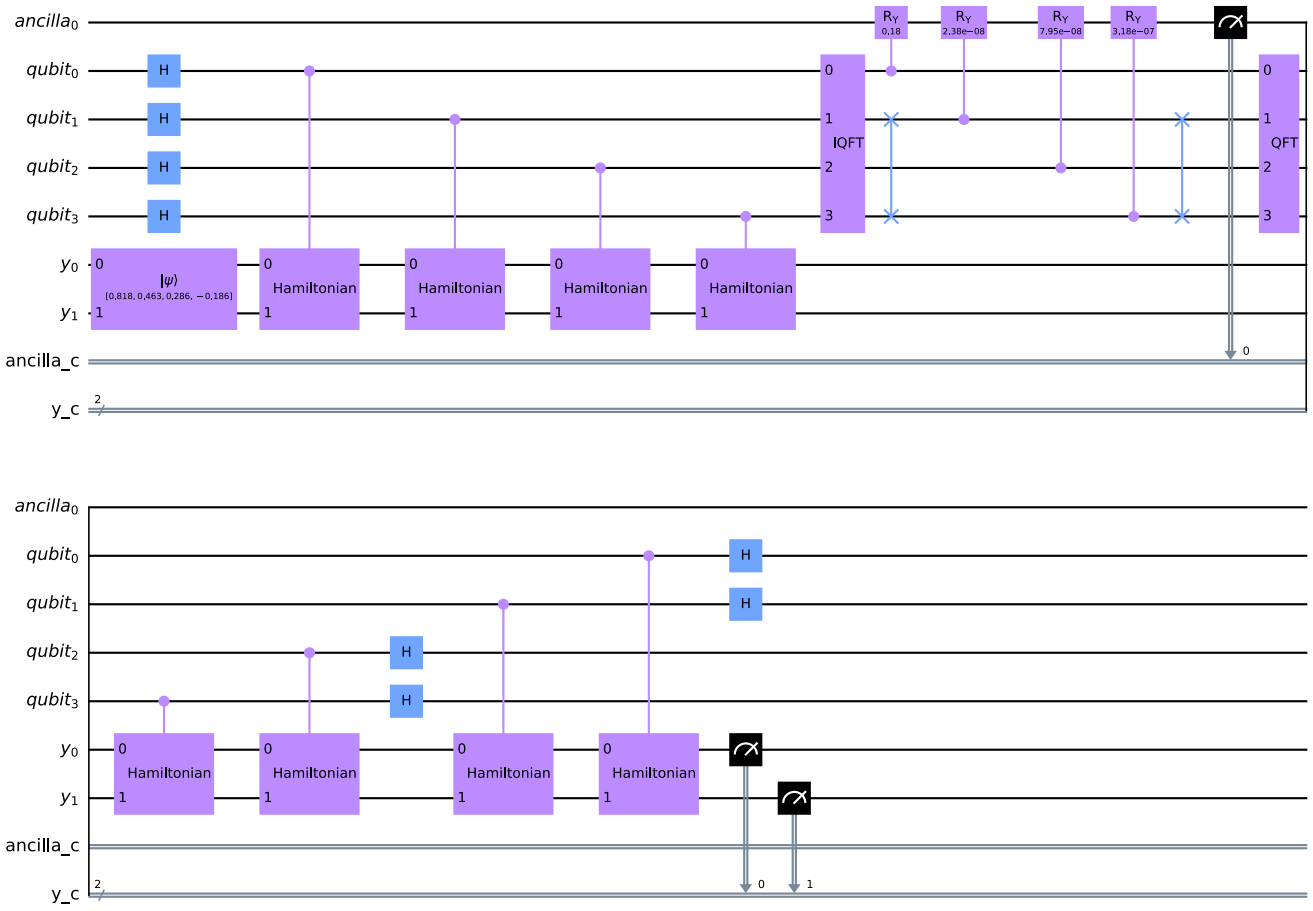



Fig. 8. Quantum circuit for solving $|W_1\rangle$ in experiment.

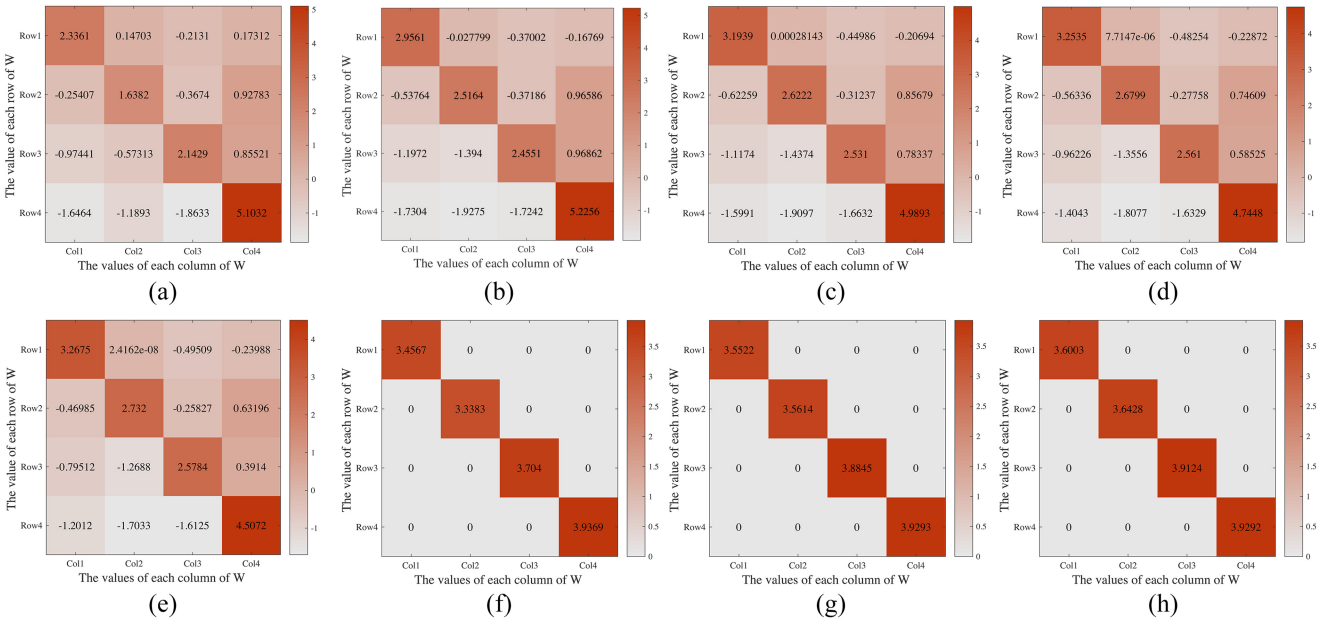



Fig. 9. Nonquantum results of $W$ at each iteration. (a) $iter = 1$. (b) $iter = 2$. (c) $iter = 3$. (d) $iter = 4$. (e) $iter = 5$. (f) $iter = 6$. (g) $iter = 7$. (h) $iter = 8$.

### C. Convergence Analysis and Quantum Circuit

Fig. 10 displays the value of the objective function at each iteration. As can be seen from Fig. 10, the algorithm reached convergence after five iterations, indicating its fast convergence. Combining Figs. 4–7, we also observe that the results from both quantum circuit outputs and classical computation outputs stabilize from the fifth iteration onward. Fig. 8 displays the circuit diagram for the first testing

TABLE IV
CONTROLLED ROTATION ANGLE CORRESPONDING TO THE THIRD TESTING DATA

| iteration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Qubit1 | 0.7408 | 0.5697 | 0.8165 | 0.9140 | 0.0949 | $5.9605\times10^{-9}$ | $5.9605\times10^{-9}$ | $5.9605\times10^{-9}$ |
| Qubit2 | 0.7374 | 0.7278 | 0.7408 | 0.7071 | 0.1365 | $2.3842\times10^{-8}$ | $2.3842\times10^{-8}$ | $2.3842\times10^{-8}$ |
| Qubit3 | 0.4531 | 0.4919 | 0.4940 | 0.4942 | 0.1647 | 0.1644 | 0.1656 | 0.1657 |
| Qubit4 | 0.2469 | 0.2493 | 0.2492 | 0.2491 | 0.1661 | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ | $3.1789\times10^{-7}$ |

TABLE V
CONTROLLED ROTATION ANGLE CORRESPONDING TO THE FOURTH TESTING DATA

| iteration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Qubit1 | 0.8901 | 0.4927 | 0.4868 | 0.5677 | 0.0609 | $5.9605\times10^{-9}$ | $5.9605\times10^{-9}$ | $5.9605\times10^{-9}$ |
| Qubit2 | 0.8090 | 0.8601 | 0.8795 | 0.8674 | 0.1702 | $2.3842\times10^{-8}$ | $2.3842\times10^{-8}$ | $2.3842\times10^{-8}$ |
| Qubit3 | 0.4757 | 0.4814 | 0.4846 | 0.4810 | 0.1582 | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ | $7.9473\times10^{-8}$ |
| Qubit4 | 0.2474 | 0.2499 | 0.2500 | 0.2499 | 0.1666 | 0.1665 | 0.1664 | 0.1664 |

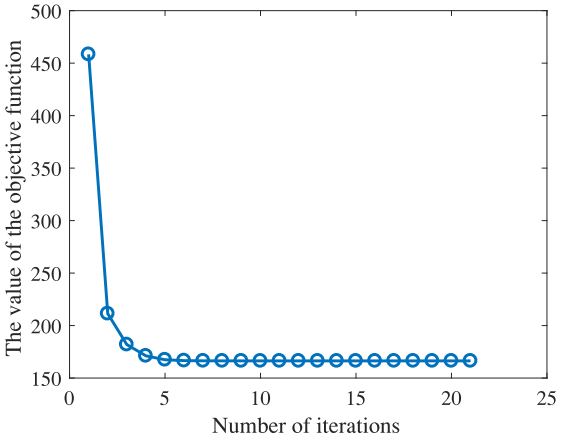



Fig. 10. Value of the objective function in each iteration.

data in the final iteration of the proposed algorithm. Since the quantum circuits differ only slightly in each iteration, with the main differences being the input quantum states $|Yb_1\rangle$, $|Yb_2\rangle$, $|Yb_3\rangle$, and $|Yb_4\rangle$, as well as the controlled rotation angles, we only show the circuit diagram for one iteration.

In Fig. 8, from bottom to top, there are three quantum registers. The first quantum register is initialized to the corresponding quantum state of the testing data. The second quantum register mainly stores the eigenvalues of the training data, and we use four qubits with their initial states being $|0\rangle$, $|0\rangle$, $|0\rangle$, and $|0\rangle$. The third register is an auxiliary quantum register, and its initial state is $|0\rangle$. After the initial state preparation, the corresponding unitary operation of the training data is obtained by the Hamiltonian simulation. Then, phase estimation, controlled rotation, and inverse phase estimation are performed sequentially. Finally, the auxiliary qubit is measured, and when $|1\rangle$ is measured, the quantum state of the final output $W$ can be obtained.

### D. Controlled Rotation Angle

The core of the proposed algorithm is to obtain quantum results for each iteration based on different controlled rotation angles. It is equivalent to $RY(\theta)$ rotation, as follows:

$$RY(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{24}$$

where $\theta = 2\arcsin\left[\sigma_i/(\sigma_i^2 + \lambda_1 D_m^{ii} + \lambda_2 Q^{ii})\right]$. Due to the fact that the training data has four eigenvalues, we need to perform four $RY(\theta)$ rotations for each testing data. Additionally, each testing data needs to correspond to an auxiliary qubit, so in one iteration, we require 16 controlled rotations. The controlled rotation angles for each testing data in each iteration are shown in Tables II–V. From Tables II–V, it can be observed that for the first testing data, starting from the 6th iteration, the controlled rotation angles for each qubit remain almost unchanged, with the first qubit's rotation angle ranging from $0.177\pm0.004$. The controlled rotation angles for the remaining three qubits remain almost constant. This indicates that after the 6th iteration, the controlled rotation angles remain almost unchanged, which is consistent with the convergence of the algorithm. For the second testing data, starting from the 6th iteration, the controlled rotation angles change relatively little. The controlled rotation angles for the first, third, and fourth qubits remain almost constant, while the second qubit's rotation angle ranges from $0.192 \pm 0.003$. Similarly, for the third and fourth testing data, starting from the 6th iteration, the controlled rotation angles change relatively little. For example, the rotation angle range for the third qubit corresponding to the third testing data is $0.165\pm0.001$. The rotation angle range for the fourth qubit corresponding to the fourth testing data is $0.1664 \pm 0.0001$.

### E. Classification Result Analysis

For the dataset in [45], based on the quantum results (Figs. 4–7) and the classical results (Fig. 9), we can obtain the

TABLE VI
CHARACTERISTICS COMPARISON OF THE QKNN ALGORITHMS

| method | Whether there are quantum circuits | Function of classification | Exponential acceleration or not | Square acceleration or not | Quantization of neighbor selection | Quantization of K value selection |
|---|---|---|---|---|---|---|
| Wiebe *et al.* [2] | ✓ | ✓ | ✓ | × | ✓ | × |
| Ruan *et al.* [3] | ✓ | ✓ | × | × | ✓ | × |
| Dang *et al.* [6] | ✓ | ✓ | × | ✓ | ✓ | × |
| Basheer *et al.* [5] | ✓ | ✓ | × | ✓ | ✓ | × |
| Zhang *et al.* [7] | ✓ | ✓ | × | ✓ | ✓ | × |
| Tian and Baskiyar [8] | × | ✓ | × | ✓ | ✓ | × |
| Gao *et al.* [9] | ✓ | ✓ | × | ✓ | ✓ | × |
| Li *et al.* [36] | ✓ | ✓ | × | ✓ | ✓ | × |
| Feng *et al.* [4] | ✓ | ✓ | × | ✓ | ✓ | × |
| Proposed | ✓ | ✓ | ✓ | × | ✓ | ✓ |

optimal $K$ value and $K$ nearest neighbors for each testing data. Specifically, the optimal $K$ value for the first testing data is 1, and its nearest neighbor is the first training data. Similarly, the nearest neighbors for the second, third, and fourth testing data are the second, third, and fourth training data, respectively. Consequently, we can deduce that the class labels for these four testing data are 1, 1, 2, and 2, respectively. This achieves a 100% accuracy.

In addition, classification experiments were conducted on the publicly available Banknote dataset. The comparative algorithm QKNN only achieved a classification accuracy of 70.83%, while the proposed algorithm achieved a significantly higher accuracy of 87.50%. This outcome suggests that the proposed algorithm exhibits superior classification performance. This phenomenon can be attributed to the proposed algorithm's ability to learn the optimal $K$ value for each testing data, thereby mitigating the influence of inappropriate $K$ values on the classification performance of the algorithm.

## V. CONCLUSION

This article introduces a novel QKNN algorithm that determines the optimal $K$ value and its corresponding $K$ nearest neighbors for each testing data by manipulating rotation angles. Specifically, the KNN objective function is formulated using least square loss, $l_{2,p}$, and $l_1$ regularization terms. Subsequently, a new quantum circuit is proposed to solve this objective function and acquire the quantum state of variables storing the optimal $K$ value and the indices of the corresponding $K$ nearest neighbors for the testing data. Finally, a series of experiments are conducted to demonstrate the efficiency of the proposed algorithm.

The core of the proposed method lies in leveraging quantum technology to address the issues of $K$ value selection and neighbor selection in KNN. To some extent, this enhances the QKNN algorithm. In addition, the proposed method can handle objective functions with sparse regularization terms, which is of great significance in quantum computing. As far as we know, a current quantum technology can only solve problems in the form of $Xw = y$, where $X$ is a known data matrix and $y$ is the corresponding class label. Clearly, $w = X^{-1}y$. The core of our proposed quantum algorithm can solve objective functions containing sparse regularization terms, as shown in (3). This capability can be very helpful for quantum computing to solve objective functions based on sparse learning.

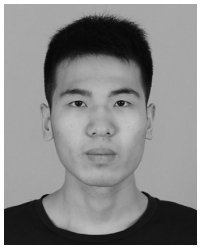
However, due to the impact of noise, the outcomes of the proposed algorithm have exhibited deviations in quantum outputs. Hence, for future investigations, we have identified two primary research directions.

1) We intend to delve into the development of an anti-noise QKNN algorithm, aiming to ensure that the algorithm's outcomes remain unaffected by noise.
2) Employing insights from the structure and functioning of quantum circuits, we will construct novel quantum denoising algorithms to address this challenge.

## REFERENCES

[1] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.
[2] N. Wiebe, A. Kapoor, and K. M. Svore, "Quantum nearest-neighbor algorithms for machine learning," *Quantum Inf. Comput.*, vol. 15, nos. 3-4, pp. 318–358, 2015.
[3] Y. Ruan, X. Xue, H. Liu, J. Tan, and X. Li, "Quantum algorithm for k-nearest neighbors classification based on the metric of hamming distance," *Int. J. Theor. Phys.*, vol. 56, pp. 3496–3507, Aug. 2017.
[4] C. Feng, B. Zhao, X. Zhou, X. Ding, and Z. Shan, "An enhanced quantum $K$-nearest neighbor classification algorithm based on polar distance," *Entropy*, vol. 25, no. 1, p. 127, 2023.
[5] A. Basheer, A. Afham, and S. K. Goyal, "Quantum $k$-nearest neighbors algorithm," 2020, *arXiv:2003.09187*.
[6] Y. Dang, N. Jiang, H. Hu, Z. Ji, and W. Zhang, "Image classification based on quantum $K$-nearest-Neighbor algorithm," *Quantum Inf. Process.*, vol. 17, pp. 1–18, Aug. 2018.
[7] Y. Zhang, B. Feng, W. Jia, and C.-Z. Xu, "An improved quantum nearest-Neighbor algorithm," in *Proc. 9th Int. Conf. Comput. Eng. Netw.*, 2021, pp. 405–413.
[8] Z. Tian and S. Baskiyar, "Fake news detection: An application of quantum $K$-nearest neighbors," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, 2021, pp. 1–6.
[9] L.-Z. Gao, C.-Y. Lu, G.-D. Guo, X. Zhang, and S. Lin, "Quantum $K$-nearest neighbors classification algorithm based on mahalanobis distance," *Front. Phys.*, vol. 10, p. 1063, Oct. 2022.
[10] J. Li et al., "Quantum $k$-fold cross-validation for nearest neighbor classification algorithm," *Physica A Stat. Mechan. Appl.*, vol. 611, Feb. 2023, Art. no. 128435.
[11] B. Li, S. Yu, and Q. Lu, "An improved $k$-nearest neighbor algorithm for text categorization," 2003, *arXiv:cs/0306099*.
[12] P. Nair and I. Kashyap, "Classification of medical image data using $K$ nearest neighbor and finding the optimal $K$ value," *Int. J. Sci. Technol. Res.*, vol. 9, no. 4, pp. 221–226, 2020.
[13] M. Saputra, H. Mawengkang, and E. Nababan, "Gini index with local mean based for determining $K$ value in $k$-nearest neighbor classification," *J. Phys., Conf. Ser.*, vol. 1235, no. 1, 2019, Art. no. 012006. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1235/1/012006/meta

[14] E. Laksono, A. Basuki, and F. Bachtiar, "Optimization of $K$ value in KNN algorithm for spam and ham email classification," *Jurnal (RESTI) (Rekayasa Sistem Dan Teknologi Informasi)*, vol. 4, no. 2, pp. 377–383, 2020.
[15] L. Juan, "TKNN: An improved KNN algorithm based on tree structure," in *Proc. 7th Int. Conf. Comput. Intell. Security*, 2011, pp. 1390–1394.
[16] S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning $K$ for KNN classification," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 3, pp. 1–19, 2017.
[17] A. Gong and Y. Liu, "Improved KNN classification algorithm by dynamic obtaining $K$," in *Proc. Int. Conf., Electron. Commerce, Web Appl., Commun., (ECWAC)*, 2011, pp. 320–324.
[18] Y. Ling, X. Zhang, and Y. Zhang, "Improved KNN algorithm based on probability and adaptive $K$ value," in *Proc. 7th Int. Conf. Comput. Data Eng.*, 2021, pp. 34–40.
[19] A. J. Gallego, J. R. Rico-Juan, and J. J. Valero-Mas, "Efficient $K$-nearest neighbor search based on clustering and adaptive $k$ values," *Pattern Recognit.*, vol. 122, Feb. 2022, Art. no. 108356.
[20] N. García-Pedrajas, J. A. R. Del Castillo, and G. Cerruela-García, "A proposal for local $k$ values for $K$-nearest neighbor rule," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 470–475, Feb. 2015.
[21] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient kNN classification with different numbers of nearest neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1774–1785, May 2018.
[22] S. Zhang and J. Li, "KNN classification with one-step computation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2711–2723, Mar. 2023.
[23] S. Zhang, J. Li, and Y. Li, "Reachable distance function for KNN classification," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7382–7396, Jul. 2023.
[24] N. Rastin, M. Z. Jahromi, and M. Taheri, "A generalized weighted distance $K$-nearest neighbor for multi-label problems," *Pattern Recognit.*, vol. 114, Jun. 2021, Art. no. 107526.
[25] J. Gou, L. Du, Y. Zhang, and T. Xiong, "A new distance-weighted k-nearest neighbor classifier," *J. Inf. Comput. Sci*, vol. 9, no. 6, pp. 1429–1436, 2012.
[26] T. Liao, Z. Lei, T. Zhu, S. Zeng, Y. Li, and C. Yuan, "Deep metric learning for $K$ nearest neighbor classification," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 264–275, Jan. 2021.
[27] C. Zhang, P. Zhong, M. Liu, Q. Song, Z. Liang, and X. Wang, "Hybrid metric $K$-nearest neighbor algorithm and applications," *Math. Problems Eng.*, vol. 2022, Jan. 2022, Art. no. 8212546. [Online]. Available: https://doi.org/10.1155/2022/8212546
[28] K. Syaliman, E. Nababan, and O. Sitompul, "Improving the accuracy of $K$-nearest neighbor using local mean based and distance weight," *J. Phys., Conf. Ser.*, vol. 978, no. 1, 2018, Art. no. 012047.
[29] C.-H. Liu, "Nearly optimal planar $K$ nearest neighbors queries under general distance functions," *SIAM J. Comput.*, vol. 51, no. 3, pp. 723–765, 2022.
[30] A. R. Lubis, M. Lubis, and A. Khowarizmi, "Optimization of distance formula in $K$-nearest neighbor method," *Bull. Elect. Eng. Inform.*, vol. 9, no. 1, pp. 326–338, 2020.
[31] J. Gou, H. Ma, W. Ou, S. Zeng, Y. Rao, and H. Yang, "A generalized mean distance-based $K$-nearest neighbor classifier," *Expert Syst. Appl.*, vol. 115, pp. 356–372, Jan. 2019.
[32] A. Zulehner, A. Paler, and R. Wille, "An efficient methodology for mapping quantum circuits to the IBM QX architectures," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 7, pp. 1226–1236, Jul. 2018.
[33] V. Saravanan and S. M. Saeed, "Data-driven reliability models of quantum circuit: From traditional ML to graph neural network," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 5, pp. 1477–1489, May 2023.
[34] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Phys. Rev. Lett.*, vol. 79, no. 2, p. 325, 1997.
[35] J. Li, M. Alam, and S. Ghosh, "Large-scale quantum approximate optimization via divide-and-conquer," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 6, pp. 1852–1860, Jun. 2023.
[36] J. Li, S. Lin, K. Yu, and G. Guo, "Quantum $K$-nearest neighbor classification algorithm based on hamming distance," *Quantum Inf. Process.*, vol. 21, no. 1, p. 18, 2022.
[37] Y. Xu, Q. Zhu, Z. Fan, M. Qiu, Y. Chen, and H. Liu, "Coarse to fine $K$ nearest neighbor classifier," *Pattern Recognit. Lett.*, vol. 34, no. 9, pp. 980–986, 2013.
[38] J. Gou, W. Qiu, Z. Yi, Y. Xu, Q. Mao, and Y. Zhan, "A local mean representation-based $K$-nearest neighbor classifier," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 3, pp. 1–25, 2019.
[39] X. Li, H. Zhang, R. Wang, and F. Nie, "Multiview clustering: A scalable and parameter-free bipartite graph fusion method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 330–344, Jan. 2020.
[40] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Phys. Rev. Lett.*, vol. 103, no. 15, 2009, Art. no. 150502.
[41] C.-H. Yu, F. Gao, and Q.-Y. Wen, "An improved quantum algorithm for ridge regression," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 858–866, Mar. 2019.
[42] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nat. Phys.*, vol. 10, no. 9, pp. 631–633, 2014.
[43] I. Cong and L. Duan, "Quantum discriminant analysis for dimensionality reduction and classification," *New J. Phys.*, vol. 18, no. 7, 2016, Art. no. 073011.
[44] P. Rebentrost, A. Steffens, I. Marvian, and S. Lloyd, "Quantum singular-value decomposition of nonsparse low-rank matrices," *Phys. Rev. A*, vol. 97, no. 1, 2018, Art. no. 012327.
[45] Y. Cao, A. Daskin, S. Frankel, and S. Kais, "Quantum circuit design for solving linear systems of equations," *Mol. Phys.*, vol. 110, nos. 15-16, pp. 1675–1680, 2012.
[46] V. T. Hai, P. H. Chuong, and P. T. Bao, "New approach of KNN algorithm in quantum computing based on new design of quantum circuits," *Informatica*, vol. 46, no. 5, pp. 95–103, 2022.



**Jiaye Li** is currently pursuing the Ph.D. degree with Central South University, Changsha, China.

His research interests include machine learning, quantum machine learning, and deep learning.

**Jian Zhang** received the B.Eng. degree in computer science from the National University of Defense Technology, Changsha, China, in 1998, and the M.Eng. and Ph.D. degrees in computer science from Central South University, Changsha, in 2002 and 2007, respectively,

He is currently a Professor with the School of Computer Science and Engineering, Central South University. His research interests include optimization theory, cyberspace security, cloud computing, and cognitive radio technology.

**Jilian Zhang** born in 1977. He received the Ph.D. degree from Singapore Management University, Singapore, in 2014.

He is an Associate Professor. His main research interests include data management, information security, and machine learning.

Dr. Zhang is a member of China Computer Federation.

**Shichao Zhang** (Senior Member, IEEE) received the Ph.D. degree from Deakin University, Burwood VIC, Australia, in 2002.

He is a China National Distinguished Professor with Guangxi Normal University, Guilin, China. He has published 90 international journal papers and over 70 international conference papers. He is a CI for 18 competitive national grants. His research interests include data mining and big data.

Dr. Zhang serves/served as an associate editor for four journals.