New Approach of KNN Algorithm in Quantum Computing Based on New Design of Quantum Circuits

Vu Tuan Hai[1], Phan Hoang Chuong[1] and Pham The Bao[2]
E-mail: haivt@uit.edu.vn, chuongph@uit.edu.vn, ptbao@sgu.edu.vn

[1]University of Information Technology, Vietnam National University - Ho Chi Minh City (VNUHCM)
Ho Chi Minh City 700000, Vietnam

[2]Saigon University, Ho Chi Minh City 700000, Vietnam

Keywords: quantum computing, machine learning, k-nearest neighbor

Received: June 27, 2021

Quantum computing is rising as a new type of computer that theoretically reduces the time complexity of classical methods exponentially because of the nature of superposition. It is promising to reduce run times on large datasets in machine learning (ML). Meanwhile, k-nearest neighbor (kNN) is a simple ML algorithm that can be translated to a quantum version (QkNN) to perform classification tasks efficiently. Here, we show a new version of QkNN which has a speed up in time complexity by using a new design of quantum circuits called integrated swap test. This quantum circuit can load two N - dimensional states and calculate the fidelity between them. Results show that QkNN allows us to take a different approach to the ML field.

Povzetek:

1 Introduction

Important progress was made in the fields of quantum computation [1, 2, 3] and machine learning [4, 5, 6, 7]. Normally, when a machine learning algorithm is loaded with large data, it shows the limitations of classical computing. Meanwhile, quantum computation provides us with a modern computing model. The integration of these two fields has resulted in a new field - quantum machine learning (QML). QML uses superposition and entanglement to create methods that can handle large data even more quickly than classical computers. Not only can QML outperform its classical counterpart in terms of speed, but it can also handle quantum data effectively. Several classical ML algorithms are ported to quantum versions, such as quantum k-means clustering (QkM) [8], quantum k nearest neighbor (QkNN) [11] quantum support vector machine (QSVM) [9], quantum principal component analysis (QPCA) [10], ... However, there is a gap in accuracy between QML and traditional ML. So, in this paper, we will propose some methods for optimizing one of the QML algorithms (the QkNN algorithm) to achieve nearly equal accuracy to its classical version.

Although these QkNN algorithms have their merits, they also have several limitations. For example, the method presented in Ref. [12] just measures the overlap between two binary states. The algorithm in Ref. [13] attempts to load the database with M data points into the circuit, which demands a greater number of qubits than is usual. The methods proposed in Ref. [14] have a limit since they can only perform binary classification. Besides, the algorithm presented in Ref. [12] requires complete knowledge of the test state which is limited when dealing with quantum data. The research in Ref. [11] deals with the pure quantum problems that can not be applied while quantum computing is in NISQ era [15]. Almost all materials that attempt to explain the benefit of QkNN on real-world data (Iris [16], Breast Cancer [17]) still have worse results than the classical version, such as Refs. [19, 20], ...

This paper is divided into three sections. Sec. 2 reviews the background of our QkNN algorithm, including classical kNN [21], swap test procedure [22], and divide-and-conquer algorithm for quantum state preparation [20], which we used to improve QkNN. Sec. 3 will be the main of this paper which describes our methodology. Sec. 4 shows the main result, the proof of the reliability of our proposed circuit, and its application in QkNN which test on the Iris dataset. Sec. 5 discusses the conclusion and potential future works.

2 Related works

2.1 Classical kNN

kNN[21] is a supervised classical machine learning algorithm used to identify test states (say $|v\rangle$) whose labels must be determined, it is based on the premise: "Two states that are similar to each other are more likely to belong to the same class or pattern". Since it is a straightforward algorithm, kNN enables us to reason about the structure of the data we are dealing with. The test and training states are also N - dimensional complex states. The working mechanism is simple: find $|v\rangle$'s k nearest neighbors based on some distance metric between it and training states (say U), and then decide its mark based on the details of these neighbors through majority voting. For the requirements of kNN, any concept of a distance metric may be used, the most common is Euclidean distance $d(|u\rangle, |v\rangle)$ with $|u\rangle \in U$, defined as Eq. (1).

$$d(|u\rangle, |v\rangle) = \sqrt{\sum_{i=0}^{N-1} (u_i - v_i)^2} \tag{1}$$

Here u_i, v_i are $|u\rangle$, $|v\rangle$ components, respectively. When we get into the quantum world, the popular alternative choice for the distance measure is fidelity $F(u, v)$. Fidelity between two such normalized states $|u\rangle$ and $|v\rangle$ is defined as Eq. (2) [23]:

$$F(\rho, \sigma) = \left(\text{Tr}\left(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right)\right)^2 \tag{2}$$

with $\rho = |u\rangle\langle u|$ and $\sigma = |v\rangle\langle v|$. In case $\hat{u}$, $\hat{v}$ are pure state, the fidelity function can simply as:

$$\begin{aligned} &F(\hat{u}, \hat{v}) = |\langle \hat{u}|\hat{v}\rangle|^2 \\ &\text{with } |\hat{u}\rangle = \frac{|u\rangle}{\|x\|} \text{ and } |\hat{v}\rangle = \frac{|v\rangle}{\|v\|}. \end{aligned} \tag{3}$$

It ranges from 0 (if the states are orthogonal or completely distinguishable) to 1 (if the states are equal). As a result, the greater the fidelity between the two states is, the closer they will be. If we get the square root of $F(\hat{u}, \hat{v})$, it will return the cosine similarity between $|u\rangle$ and $|v\rangle$, $(|u\rangle, |v\rangle)$, defined as Eq. (4).

$$\begin{aligned} &\sqrt{F(\hat{u}, \hat{v})} = |\langle u|v\rangle| = (|u\rangle, |v\rangle) \\ &= \frac{\sum_{i=0}^{N-1} u_i v_i}{\sum_{i=0}^{N-1} u_i^2 \sum_{i=0}^{N-1} v_i^2}. \end{aligned} \tag{4}$$

The kNN algorithm is composed of the steps which are presented in Algorithm. 1.

Although the kNN algorithm is simple to understand and implement, it has some limitations. kNN can easily become intractable for classical computers as the number of dimensions of training states increases. Classifying an N - dimensional test state by applying it to M training states needs about $\mathcal{O}(M)$ multiplication operations. Furthermore, there is no general approach for determining k so hyperparameter tuning is typically used to determine the best possible k [18]. So the total complexity $\mathcal{O}(kM)$. Meanwhile, QkNN will take only $\mathcal{O}(\sqrt{kM})$ if it combine with quantum search algorithm [30]. The complexity analysis will be discussed more in Sec. 3.2.

2.2 Swap test procedure

The swap test is introduced in Ref. [22] can return the correct value of fidelity between two quantum states $|\psi\rangle$ and $|\phi\rangle$ as mentioned in Eq. 2 by a lot of measurements. In order to implement the swap test, we need at least three registers prepared in states $|0\rangle$, $|\psi\rangle$ and $|\phi\rangle$ as Fig. 1.

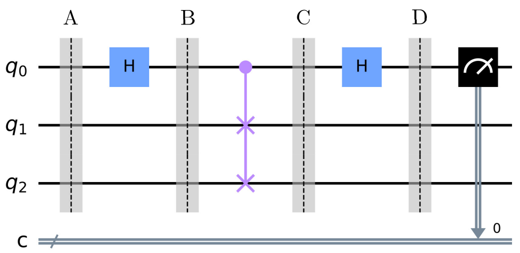


Figure 1: Circuit diagram for swap test. H is the Hadamard gate. First register will be measured, second and third registers respond for load $|\psi\rangle$ and $|\phi\rangle$, respectively.

At point A, the initial system's state is simply:

$$|R_A\rangle = |0\rangle|\psi\rangle|\phi\rangle \text{ or } |0, \psi, \phi\rangle \text{ for short.} \tag{5}$$

After that, the Hadamard gate convert first qubit q_0 into the superposition state:

$$H|q_0\rangle = H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \tag{6}$$

Thus, at point B, the whole system is in the state:

$$|R_B\rangle = \frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\psi\rangle|\phi\rangle). \tag{7}$$

Whereas the action of Fredkin gate (or CSWAP gate) between point B and point C reads:

$$\begin{aligned} C_S|0\rangle|\psi\rangle|\phi\rangle &= |0\rangle|\psi\rangle|\phi\rangle, \\ C_S|1\rangle|\psi\rangle|\phi\rangle &= |1\rangle|\phi\rangle|\psi\rangle. \end{aligned} \tag{8}$$

So at point C, we have:

$$\begin{aligned} &C_S|R_B\rangle = |R_C\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\phi\rangle|\psi\rangle). \end{aligned} \tag{9}$$

Finally, we apply Hadamard gate on q_0 again that make system into:

$$\begin{aligned} |R_D\rangle = &\frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\phi\rangle|\psi\rangle) \\ &+\frac{1}{\sqrt{2}}(|0\rangle|\phi\rangle|\psi\rangle - |1\rangle|\psi\rangle|\phi\rangle) \end{aligned} \tag{10}$$

When we measure the first qubit, there are two possible states that we can receive:

Algorithm 1 kNN

Input: training states $U = u_0, u_1, \dots u_{(M-1)}$ and its labels $U_L = u_{L_0}, u_{L_1}, \dots, u_{L_{(M-1)}}$, test states (unlabeled) $V = v_0, v_1, \dots, v_{\left(M'-1\right)}$ and hyperparameter k.

Output: set of test labels $V_L = (v_{L_0}, v_{L_1}, \dots, v_{L_{\left(M'-1\right)}})$.

```
1:  function PREDICT
2:      for $i \leftarrow 0$ to $M^{'} - 1$ do
3:          Load all training states $U$ and test states $v_i$ on the register.
4:          for $j \leftarrow 0$ to $M - 1$ do
5:              Calculate $(\text{dist}_{(v_i)})_j = \text{dist}(u_j, v_i)$.
6:          end for
7:          Sort $\text{dist}_{(v_i)} = \{(\text{dist}_{(v_i)})_0, ...(\text{dist}_{(v_i)})_{((M-1)}\}$ (descending or ascending
8:          Choose $k$ neighbors which are nearest to $v_i$.
9:          Conduct majority voting and assign the label $v_{L_i}$ of the majority to the test point.
10:     end for
        return $V_L = (v_{L_0}, v_{L_1}, \dots, v_{L_{\left(M'-1\right)}})$
11: end function
```

$$\begin{aligned}
&P(q_0 = 0) \text{ or } P(0) \\
&= \frac{1}{2}(\langle\psi|\langle\phi| + \langle\phi|\langle\psi|)\frac{1}{2}(|\psi\rangle|\phi\rangle + |\phi\rangle|\psi\rangle) \\
&= \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2 \\
&P(q_0 = 1) \text{ or } P(1) \\
&= \frac{1}{2}(\langle\psi|\langle\phi| - \langle\phi|\langle\psi|)\frac{1}{2}(|\psi\rangle|\phi\rangle - |\phi\rangle|\psi\rangle) \\
&= \frac{1}{2} - \frac{1}{2}|\langle\psi|\phi\rangle|^2
\end{aligned} \tag{11}$$

The final result now is:

$$F(\psi, \phi) = P(0) - P(1) = |\langle\psi|\phi\rangle|^2. \tag{12}$$

If $|\psi\rangle$ and $|\phi\rangle$ are orthogonal, $|\langle\psi|\phi\rangle|^2 = 0$, that mean $P(0) = P(1) = 0.5$. Otherwise, if $|\psi\rangle$ and $|\phi\rangle$ are identical, $|\langle\psi|\phi\rangle|^2$ = 1, that mean $P(0) = 1, P(1) = 0$.

2.3 Prepare quantum state

Normally, loading quantum states requires the quantum system a lot of gates because the state $|\psi\rangle = c_0|0\rangle + ... + c_{N-1}|N-1\rangle$ (or $\sum_{j=0}^{N-1} c_j|j\rangle$ has the dimensional N which frequently greater than 2, and components in it are belong to the complex space.

An easiest approach is to encode m - bits of binary data into qubit strings. Let $d_j = \{d_j^{(k)}\}_{k=1}^{m} (d_j^{(k)} = 0, 1; j = 0, \dots, N-1)$ be the binary bit-strings, the equation for encoding can be described by Ref. [24]:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle|d_j^{(m)}...d_j^1\rangle. \tag{13}$$

This method of encoding is simple and can represent any state by using a series of NOT gate. The downside is that it not only does not use superposition but also needs more qubits - mN qubits for one state to encode N components in it. The greater m is, the more precisely the component is defined.

Another way which is provided in Ref. [28, 29, 20] will help us use fewer qubits a lot, this devised method is based on quantum forking [25]. The basic theory is to break a problem into sub-problems of the same class and then combine the solutions of these sub-problems to offer the solution to the original problem. The main idea is building the quantum state in each level of the state tree in a top-down or divide-and-conquer strategy. At each step, they divide the input into bi-dimensional sub-vectors, load it on qubit and repeat until reached the desired quantum state.

In case of loading N - dimensional real vectors, these method [28, 29] take only $\mathcal{O}(\log_2 N)$ qubits and $\mathcal{O}(N)$ computation cost. But it have an exponential depth in relation to the number of qubits. Another method is comes from Ref. [20] which has overall depth $\mathcal{O}(\log_2^2 N)$ but using $\mathcal{O}(N)$ qubits.

Three above methods proposed a trade-off between the number of qubits and circuit depth, when using more qubits we have a short execution time and vice versa. For fitting with the NISQ device and simulation's resources, we will choose Ref. [20] as the loading component for dealing with states that have small or medium dimensions.

3 Methodology

3.1 Integrated swap test circuit

As discussed in the previous section, if we want to perform the calculation about the fidelity between two N - dimensional states like $|\psi\rangle$ and $|\phi\rangle$, we must have a quantum circuit that has two separate functions. The first will load $2N$ components $\{\psi_i\}_{i=0}^{N-1}$, $\{\phi_i\}_{i=0}^{N-1}$. The second will apply the N - swap test to measure and calculate the probability of value 0 or 1 on the first qubit, $\sqrt{P(0) - P(1)}$ will be

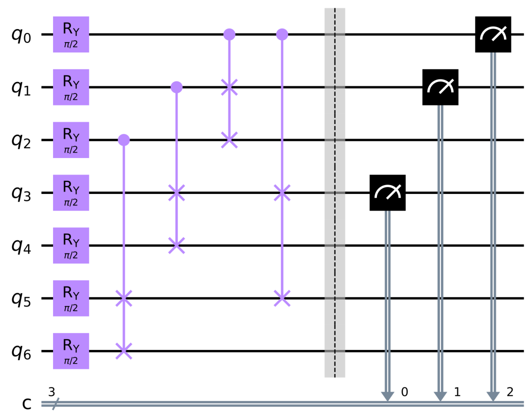


Figure 2: A circuit generated by the divide-and-conquer strategy described in Ref. [20] that encode the 8–dimensional state $|\psi\rangle = [\frac{1}{\sqrt{8}}\ \frac{1}{\sqrt{8}} \dots \frac{1}{\sqrt{8}}]^T$.

the final result. The idea to create a quantum circuit that meets the above requirement is to combine the circuit in Ref. [20], $\log_2 N$ (or n) Fredkin gates and two Hadamard gates. The outcome from the circuits in Ref. [20] will be the input of the second component. This new design of the quantum circuit whose name is integrated swap test, is described in Fig. 3.

The total qubits that need to be used in this circuit in detail is $(2(N-1)+1)$ for three sub - circuits. The first sub - circuit also the first qubit, will be measured to calculate the probability of value 0 or 1 on c (a classical channel, show in Fig. 3). The second and third sub - circuits (v_1 and v_2) have $(N-1)$ qubits which is from index 1 to $(N-1)$ and from N to $2(N-1)$ use to encode $|\psi\rangle$ and $|\phi\rangle$, respectively. It continued to be active on wires which has indices:

$$\begin{aligned} I_{v_1} &= \{i^2 | 1 \le i \le n\} \text{ for } |\psi\rangle, \\ I_{v_2} &= \{i^2 + (N-1) | 1 \le i \le n\} \text{ for } |\phi\rangle. \end{aligned} \quad (14)$$

Our n Fredkin gates are attached on:

$$\begin{aligned} I^n_{C_S} = \{ & I_{C_S} | I_{C_S} = (0, (i_{v_1})_j, (i_{v_2})_j), \\ & i_{v_1} \in I_{v_1}, i_{v_2} \in I_{v_2}, j = 1, \dots, n)\}. \end{aligned} \quad (15)$$

Control qubits in n Fredkin gates always have index 0, the second i_{v_1} and the third i_{v_2} in the tuple I_{C_S} are the target qubits which swap each other if the first qubit has value 1, nothing happened otherwise. Two Hadamard gates remain on the first wire like the original swap test.

About the complexity, the number of qubits used to compute 2 N - dimensional states is still $\mathcal{O}(N)$, and the circuit depth is $\mathcal{O}(\log_2^2 N + \log_2 N + 1)$ which is the sum of loading stage and N - swap test stage.

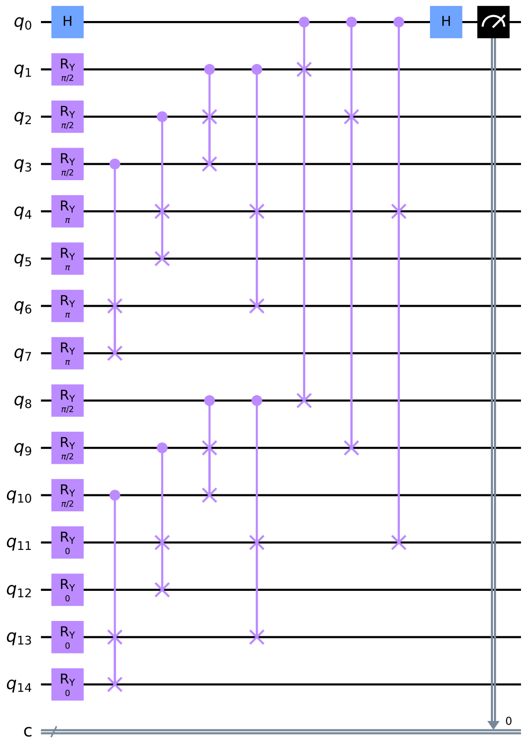


Figure 3: Our new circuit that can calculate the fidelity between two 8 - dimensional states $|\psi\rangle = [0\ 1\ 0\ 1\ 0\ 1\ 0\ 1]^T$ and $|\phi\rangle = [1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]^T$.

3.2 Hybrid QkNN algorithm when using integrated swap test circuit

Formally, our QkNN algorithm still consists of many steps that are described in Algorithm 1 but it has other crucial points. Not like other QkNN versions that deal with pure quantum problems [12, 13, 11], we use the idea that can combine the advantages of classical computing and quantum computing. The classical computer will do its best job including load states (Step 1 in Fig. 4), sort descending (Step 6 in Fig. 4), and mark label for test state (Step 7 in Fig. 4). The quantum computer only does the calculating fidelity task that outperforms the classical computer in case the dimension of states increases exponentially.

In the first step, the classical computer will put a training state u_j and a test state v_i into the integrated swap test circuit. This step will be repeated $MM^{'}$ times with M and $M^{'}$ are the cardinality of the training set and test set, respectively. In the second and third steps, the quantum computer creates phases $\{\omega_i\}_{i=0}^{N-2}$ for the R_y gates in the circuit, these gates will be recreated $MM^{'}$ times. For the pair (u_j, v_i), the circuit must perform - measure n_{iter} times to reduce the noise from the real environment, it finally returns the average of all results. The fifth, sixth and seventh steps are not hard to understand, it is completely similar to the classical kNN. All of the steps from 2 to 7 described above will be repeated until the test set does not have any unlabeled states.

Because all computation is done during prediction, there is no training phase, and we have $\mathcal{O}(1)$ for both time and space. The prediction part is split into 2 parts: classical and quantum. Computing the distances between all train states and each test state will consume about $\mathcal{O}(M\log_2^2 N)$, when taking k nearest neighbors will depend on the classical algorithm. If we use the Brute-force algorithm, the complexity is $\mathcal{O}(kM)$ or use some Sort algorithm, the complexity is $\mathcal{O}(M\log M)$. So the total complexity when predicting $M^{'}$ test states is $\mathcal{O}(M^{'}(M\log_2^2 N + kM) + 1)$ or $\mathcal{O}(M^{'}(M\log_2^2 N + M\log M) + 1)$.

In some research about fully quantum kNN, the above complexities can be reduced to $\mathcal{O}(M^{'}M(\log_2^2 N + \sqrt{kM}) + 1)$ by replacing Brute-force or Sort algorithm by Grover algorithm [12, 13]. Obviously, this way will make our circuit deeper.

4 Experiments

To evaluate the proposed methods, we perform two experiments. In the first experiment, we prove that the proposed circuit can be applied in some algorithms. In the second experiment, we use this circuit in our new QkNN on Iris dataset.

We use Qiskit SDK [27] (version 0.24.1) in all experiments because of its completeness and ease of use, the maximum number of qubits is 31 on Intel core i5-5250U and 8 GB RAM classical computer. The maximum of dimensional of state is 15, so $2(N-1)+1 < 31$ and $N < 16$. The number of shots is 16384.

The codes in this paper, including documentation, are available on Ref. [26].

4.1 Proof of reliability of integrated swap test circuit

We evaluate the integrated swap test circuit on various iterations from 0 to 10000 and the number of dimensions $\{2^k\}_{k=1}^3$. As in the Fig. 5, we easily see that the error is quite chaotic in 0 - 1000 iterations. It tends to be in an order and gets smaller according to the iterations from 1000 to 10000. For the 2 - dimensional state, the error is acceptable with Standard error (RE) between 0.25% and 0.75%. But in case of higher dimensional state, the average RE are $0.27\%, 1.01\%$ and 5.34% for 2, 4 and 8 - dimensional respectively, it increases exponentially. We can explain this phenomenon as: the higher dimensional state, smaller the value of components. Meanwhile, we can use only a finite number of shots ($n_{\text{shot}} = 16384$), the smallest division is $\frac{1}{16384}$ that can not be reduced anymore, note that $P(0) = \frac{n_{(c=0)}}{n_{\text{shot}}}$ and $P(1) = \frac{n_{(c=1)}}{n_{\text{shot}}}$.

4.2 Hybrid QkNN on Iris dataset

Since the number of qubits in the quantum circuit are restricted, the resources for describing the states are limited. Currently, the way the quantum circuit works is that one circuit is made M times per test data point and $MM^{'}$ times in the whole predicting stage. That means the number of training states and test states is not the largest limiting factor because the quantum circuit can be recreated once the machine runs out of memory. The limiting factor is the dimensionality, the memory accepted with these values at 8. Values not following the relation of $N = 2^n$ can be chosen but can be considered a waste of qubit resources.

The Iris dataset is 4 - dimensional and consists of 150 examples in total of three classes. It consists of length and width measurements of sepals and petals from three different flowers: Iris Setosa, Iris Versicolour, and Iris Virginica. In short, the dimensionality does not need to be changed, and the instances of each class will be equal to each other. The values of a state $|x\rangle$ must to be encoded to quantum state through dividing components $\{x_i\}_{i=0}^3$ by norm $|x|$ before conducting the experiment.

After prepared data, the QkNN may be configured to begin classifying it. As mentioned in Sec. 4, Qiskit used in this project so the circuits will be simulated with the **qasm_simulator**. This simulator would run the circuit as if it were running on a quantum computer, which means it will have to run several times in order to reach the state provided by the quantum circuit. The setups of experiment include:

1. Using full features of Iris dataset (4 features).
2. Number of training states and test states will be $n_{\text{train}} = \{2^i\}_{i=3}^7$ and $n_{\text{test}} = \{\lfloor 0.3 * 2^i \rfloor\}_{i=3}^7$ respectively. The train set and test set are not overlapped together.
3. Conduct on a variety of k number of neighbors with $k = \{1, 3, 5, 7\}$. In case of k is large compared to n_{train} ($k = \{5, 7\}$ and $n_{\text{train}} = \{8, 16\}$), the experiment will not be conducted because the kNN will produce an underfitted model.

The first thing we can see is that the accuracy of classical and quantum method varies. From the metrics in Tab. 1, it is observable that on the Iris dataset, Hybrid QkNN outperforms classical kNN in some cases ($k = \{3, 5\}$) but averagely classical algorithm is still better than Hybrid QkNN. When comparing with recent QkNN [13, 19], our method was outperformed in almost all cases, these old methods have struggled when dealing with a large number of train states. On the other hand, Hybrid QkNN runs faster than a little bit, the total run-times shown as Tab. 3. The old QkNN have more disadvantage in that the train set must be small, all train states was loaded on the Oracles at the same times. So the experiments on $n_{\text{train}} = 128$ can not be conducted because the algorithm makes the simulation run out of memory quickly.

Some of the facts in Tab. 2 also indicate that the classical kNN and Hybrid QkNN both tend to increase as n_{train} increases. This phenomenon can be explained by looking at the confusion matrices in Fig. 7.

It is apparent that in the confusion matrices, accuracy which is smaller than 100% is always in the case of the truth label is belong to class 2 (Virginica) but our algorithm labeled class 1 (Versicolor). This comes from the fact that we encoded the normal state in terms of quantum state as shown from Fig. 6. a to Fig. 6. b, all states now occupy on the surface of $4-$ dimensional ball and make the ratio between the interference space between the states labeled 1 and the states labeled 2 and the whole space of all states is larger. Meanwhile, the space of the states labeled 0 is quite separate. That leads to the states of these two labels being confused.

5 Conclusion

In this paper, we present the Hybrid QkNN algorithm, this algorithm uses the new design of quantum circuits that can achieve higher functionality compared to its previous version (swap test). One of the most important advantages of the Hybrid QkNN is that it is capable of handling a large number of train states without many qubits because it is combined with a classical computer to reduce a lot of tasks. The task that classical computer accomplishes currently do not have their corresponding quantum version so it is still performed well on a classical computer.

In our experiment, we simulated the Hybrid QkNN to solve the problem of classifying the Iris dataset. The results showed that our method can provide a high degree of accuracy that is nearly equal to the classical kNN. The reasons which make this algorithm does not achieve the same or higher accuracy come from the fact that we are forced to encode the normal state into the quantum state and the number of shots is limited. This can be fixed in the future if we have a more efficient simulator or simply experiment on a real quantum computer which is considered by experts to exist in the next few decades or earlier [15].

Acknowledgement

This research was supported by The VNUHCM-University of Information Technology's Scientific Research Support Fund under Grant No. D1-2022-08.

References

[1] Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J., Barends, R., Boixo, S., Broughton, M., Buckley, B., Buell, D. & Others. Hartree-Fock on a superconducting qubit quantum computer. *Science*. **369**, 1084-1089 (2020). https://doi.org/10.1126/science.abb9811

[2] Wu, S., Chan, J., Guan, W., Sun, S., Wang, A., Zhou, C., Livny, M., Carminati, F., Di Meglio, A., Li, A. & Others. Application of quantum machine learning using the quantum variational classifier method to high energy physics analysis at the LHC on IBM quantum computer simulator and hardware with 10 qubits. *Journal Of Physics G: Nuclear And Particle Physics*. **48**, 125003 (2021). https://doi.org/10.1088/1361-6471/ac1391

[3] Alexeev, Y., Bacon, D., Brown, K., Calderbank, R., Carr, L., Chong, F., DeMarco, B., Englund, D., Farhi, E., Fefferman, B. & Others. Quantum computer systems for scientific discovery. *PRX Quantum*. **2**, 017001 (2021). https://doi.org/10.1103/PRXQuantum.2.017001

[4] Dogan, A. & Birant. D. Machine learning and data mining in manufacturing. *Expert Systems With Applications*. **166** pp. 114060 (2021). https://doi.org/10.1016/j.eswa.2020.114060

[5] Greener, J., Kandathil, S., Moffat, L. & Jones, D. A guide to machine learning for biologists. *Nature Reviews Molecular Cell Biology*. **23**, 40-55 (2022). https://doi.org/10.1038/s41580-021-00407-0

[6] Shorten, C., Khoshgoftaar, T. & Furht, B. Deep Learning applications for COVID-19. *Journal Of Big Data*. **8**, 1-54 (2021). https://doi.org/10.1186/s40537-020-00392-9

[7] Ma, W., Liu, Z., Kudyshev, Z., Boltasseva, A., Cai, W. & Liu, Y. Deep learning for the design of photonic structures. *Nature Photonics*. **15**, 77-90 (2021). https://doi.org/10.1038/s41566-020-0685-y

[8] Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. *ArXiv Preprint ArXiv:1307.0411*. (2013)

[9] Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum support vector machine for big data classification. *Physical Review Letters*. **113**, 130503 (2014). https://doi.org/10.1103/PhysRevLett.113.130503

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

[11] Wiebe, N., Kapoor, A. & Svore, K. Quantum Algorithms for Nearest-Neighbor Methods for Supervised and Unsupervised Learning. *Quantum Information And Computation*. **15**, 318-358 (2015). https://doi.org/10.48550/arXiv.1401.2142

[12] Ruan, Y., Xue, X., Liu, H., Tan, J. & Li, X. Quantum algorithm for k-nearest neighbors classification based on the metric of hamming distance. *International Journal Of Theoretical Physics*. **56**, 3496-3507 (2017). https://doi.org/10.1007/s10773-017-3514-4

k - nearest neighbors	1	3	5	7	Average
Classical kNN	98.7%	92.8%	96.7%	98.2%	96.6%
QkNN [13, 19]	70.2%	71.6%	75.7%	50.7%	67.1%
Our method	95.1%	**97.3**%	**97.4**%	94.5%	95.7%

Table 1: A comparison between classical kNN, QkNN [13, 19] and our method (Hybrid QkNN) on the same dataset based on k. The accuracy of each k was calculated averagely on n_{train} from 16 to 128.

n_{train}	8	16	32	64	128	Average
Classical kNN	100%	91.7%	92.6%	96.0%	100%	96.6%
QkNN [13, 19]	100%	100%	49.5%	43.4%	*	67.1%
Our method	100%	**100**%	91.7%	92.1%	97.3%	95.7%

Table 2: A comparison between classical kNN, QkNN [13, 19] and our method (Hybrid QkNN) on the same dataset based on n_{train}. The accuracy of each n_{train} was calculated averagely on k from 1 to 7.

[13] Basheer, A., Afham, A. & Goyal, S. Quantum k -nearest neighbors algorithm. *ArXiv Preprint ArXiv:2003.09187*. (2020). https://doi.org/10.48550/arXiv.2003.09187

[14] Fastovets, D., Bogdanov, Y., Bantysh, B. & Lukichev, V. Machine learning methods in quantum computing theory. *International Conference On Micro-and Nano-Electronics 2018*. **11022** pp. 110222S (2019). https://doi.org/10.1117/12.2522427

[15] Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum*. **2** pp. 79 (2018). https://doi.org/10.22331/q-2018-08-06-79

[16] Fisher, R. The use of multiple measurements in taxonomic problems. *Annals Of Eugenics*. **7**, 179-188 (1936). https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

[17] Zwitter, M. & Soklic, M. Breast Cancer dataset. (1988), data retrieved from UCI Machine learning.

[18] Zhang, S., Li, X., Zong, M., Zhu, X. & Cheng, D. Learning k for knn classification. *ACM Transactions On Intelligent Systems And Technology (TIST)*. **8**, 1-19 (2017). https://doi.org/10.1145/2990508

[19] Kok, D., Caron, S. & Acun, A. Building a quantum kNN classifier with Qiskit: theoretical gains put to practice. (2021)

[20] Araujo, I., Park, D., Petruccione, F. & Silva, A. A divide-and-conquer algorithm for quantum state preparation. *Scientific Reports*. **11**, 1-12 (2021). https://doi.org/10.1038/s41598-021-85474-1

[21] Cunningham, P. & Delany, S. k-Nearest neighbour classifiers-A Tutorial. *ACM Computing Surveys (CSUR)*. **54**, 1-25 (2021). https://doi.org/10.1145/3459665

[22] Buhrman, H., Cleve, R., Watrous, J. & De Wolf, R. Quantum fingerprinting. *Physical Review Letters*. **87**, 167902 (2001). https://doi.org/10.1103/PhysRevLett.87.167902

[23] Jozsa, R. Fidelity for mixed quantum states. *Journal Of Modern Optics*. **41**, 2315-2323 (1994). https://doi.org/10.1080/09500349414552171

[24] Mitarai, K., Kitagawa, M. & Fujii, K. Quantum analog-digital conversion. *Physical Review A*. **99**, 012301 (2019). https://doi.org/10.1103/PhysRevA.99.012301

[25] Park, D., Sinayskiy, I., Fingerhuth, M., Petruccione, F. & Rhee, J. Parallel quantum trajectories via forking for sampling without redundancy. *New Journal Of Physics*. **21**, 083024 (2019). https://doi.org/10.1088/1367-2630/ab35fb

[26] Hai, V. Source code and experiment results. (2022), Github, https://github.com/vutuanhai237/HybridQkNN

[27] Quantum, I. Qiskit open-source. (2021), Github, https://github.com/Qiskit

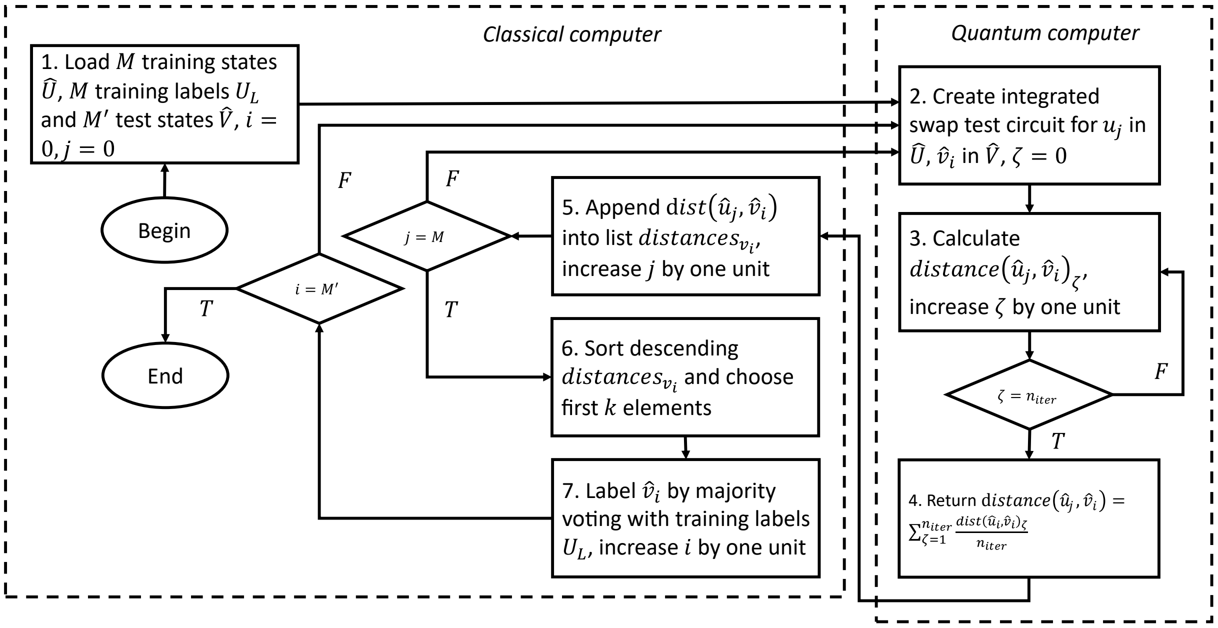
[28] Iten, R., Colbeck, R., Kukuljan, I., Home, J. & Christandl, M. Quantum circuits for isometries. *Physical Review A*.**93**, 032318 (2016). https://doi.org/10.1103/PhysRevA.93.032318

[29] Park, D., Petruccione, F. & Rhee, J. Circuit-based quantum random access memory for classical data. *Scientific Reports*. **9**, 1-8 (2019). https://doi.org/10.1038/s41598-019-40439-3

[30] Grover, L. A fast quantum mechanical algorithm for database search. *Proceedings Of The Twenty-eighth Annual ACM Symposium On Theory Of Computing*. pp. 212-219 (1996). https://doi.org/10.1145/237814.237866

n_{train}	8	16	32	64	128
QkNN [13, 19]	7s	51s	464s	4138s	*
Our method	8s	**32s**	**257s**	**2835s**	45600s

Table 3: Total of run-times between QkNN [13, 19] and our method (Hybrid QkNN) on the same dataset based on n_{train}. **The experiments of QkNN [13, 19] on $n_{train} = 128$ can not be conducted because the algorithm makes the simulation run out of memory.*



Classical computer
Quantum computer
1. Load M training states $\widehat{U}$, M training labels U_L and M' test states $\widehat{V}$, $i = 0, j = 0$
Begin
End
$i = M'$
$j = M$
F
F
T
T
2. Create integrated swap test circuit for u_j in $\widehat{U}$, $\hat{v}_i$ in $\widehat{V}$, $\zeta = 0$
3. Calculate $distance(\hat{u}_j, \hat{v}_i)_\zeta$, increase ζ by one unit
$\zeta = n_{iter}$
F
T
4. Return $distance(\hat{u}_j, \hat{v}_i) = \sum_{\zeta=1}^{n_{iter}} \frac{dist(\hat{u}_i, \hat{v}_i)_\zeta}{n_{iter}}$
5. Append $dist(\hat{u}_j, \hat{v}_i)$ into list $distances_{v_i}$, increase j by one unit
6. Sort descending $distances_{v_i}$ and choose first k elements
7. Label $\hat{v}_i$ by majority voting with training labels U_L, increase i by one unit

Figure 4: Hybrid QkNN's diagram, a version of QkNN with collaboration between classical computer and quantum computer.

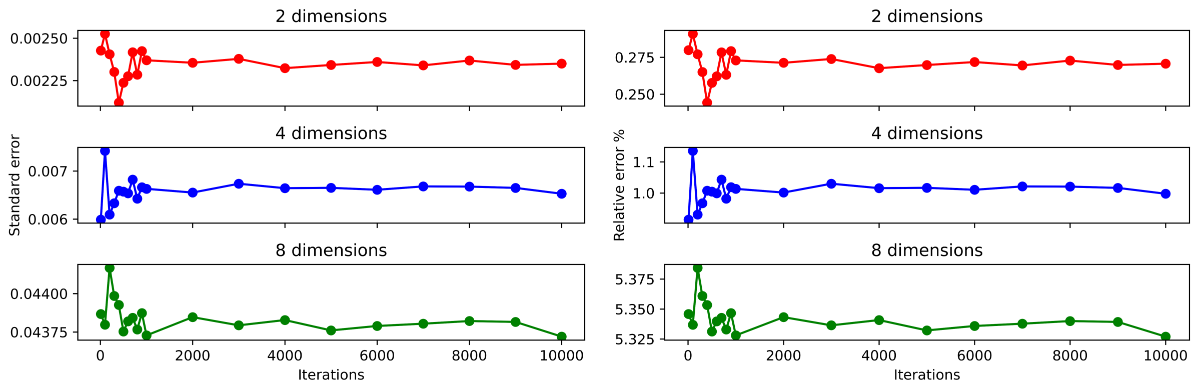
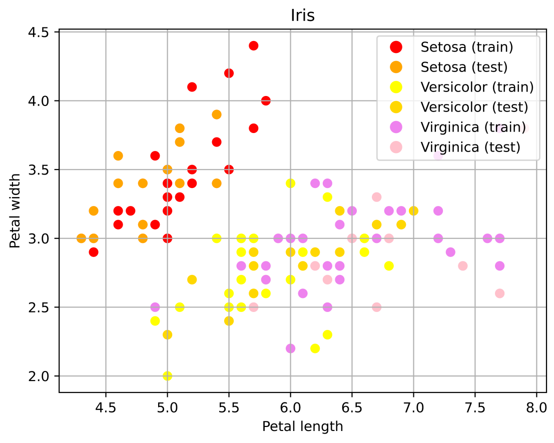
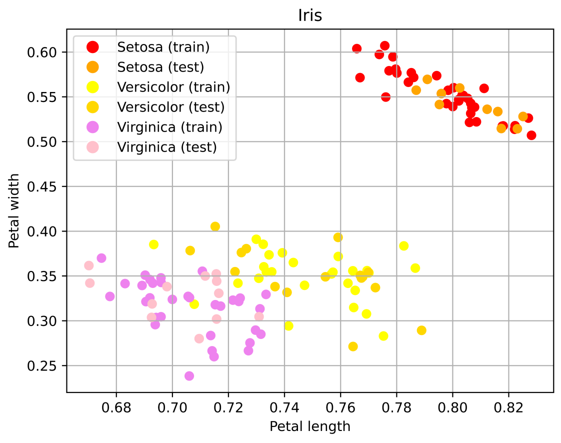


Figure 5: Standard error (SE) and relative error (RE) when calculating fidelity between two states, are defined as: $SE = \frac{\sum_{i=0}^{n_{\text{iter}}-1} |f_{\text{predict}_i} - f_{\text{truth}}|}{n_{\text{iter}}}$ and $RE = \frac{SE}{(\frac{\sum_{i=0}^{n_{iter}-1} f_{\text{predict}}}{n_{\text{iter}}})} * 100(\%)$.



a) The normal Iris



b) The encoded Iris

Figure 6: Visualisation of the encoding method on the dataset (Sepal width and Sepal height do not display in this figure). The untouched dataset (a), and the dataset encoded to be loaded into the quantum circuit (b).

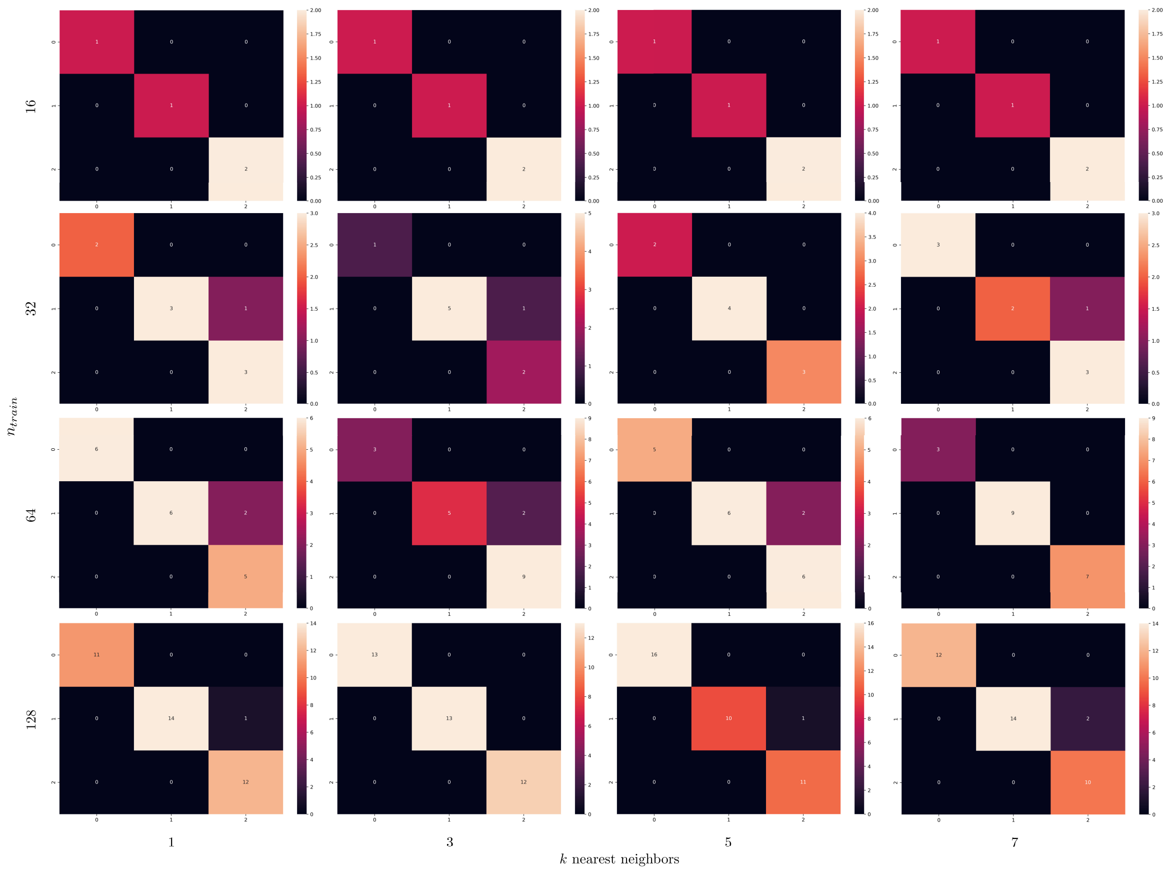


Figure 7: The confusion matrices on different n_{train} and k nearest neighbors. The color of boxes ranges from white to black corresponding for high to low integer values.