Quantum Algorithm for K-Nearest Neighbors Classification Based on the Metric of Hamming Distance

Article *in* International Journal of Theoretical Physics · November 2017

DOI: 10.1007/s10773-017-3514-4

5 authors, including:

Heng Liu
Anhui University of Technology
41 PUBLICATIONS **339** CITATIONS
SEE PROFILE

Jianing Tan
Southeast University (China)
9 PUBLICATIONS **57** CITATIONS
SEE PROFILE

Xi Li
Southeast University (China)
10 PUBLICATIONS **54** CITATIONS
SEE PROFILE

Int J Theor Phys
DOI 10.1007/s10773-017-3514-4

Quantum Algorithm for K-Nearest Neighbors Classification Based on the Metric of Hamming Distance

Yue Ruan[1,2] · Xiling Xue[1] · Heng Liu[1] · Jianing Tan[3] · Xi Li[3]

Received: 12 May 2017 / Accepted: 16 August 2017

Abstract K-nearest neighbors (KNN) algorithm is a common algorithm used for classification, and also a sub-routine in various complicated machine learning tasks. In this paper, we presented a quantum algorithm (QKNN) for implementing this algorithm based on the metric of Hamming distance. We put forward a quantum circuit for computing Hamming distance between testing sample and each feature vector in the training set. Taking advantage of this method, we realized a good analog for classical KNN algorithm by setting a distance threshold value t to select $k - nearest$ neighbors. As a result, QKNN achieves $O(n^3)$ performance which is only relevant to the dimension of feature vectors and high classification accuracy, outperforms Llyod's algorithm (Lloyd et al. 2013) and Wiebe's algorithm (Wiebe et al. 2014).

✉ Yue Ruan
yue_ruan@163.com

Xiling Xue
stmxue@163.com

Heng Liu
hengliusky@aliyun.com

Jianing Tan
jn_tan@163.com

Xi Li
230169107@seu.edu.cn

[1] School of Computer Science and Technology, Anhui University of Technology, Maanshan 243005, China

[2] Key Laboratory of Computer Network and Information Integration, Southeast University, Ministry of Education, Nanjing 210096, China

[3] School of Computer Science and Engineering, Southeast University, Nanjing 210096, China

Published online: 23 August 2017

Keywords Quantum algorithm · K-nearest neighbors algorithm · Hamming distance · Classification · Machine learning

1 Introduction

Classification (the broader understanding of it should include clustering), has been studied as a core issue in machine learning for many decades. In recent years, the research of such issues meets real challenges for the larger and larger dataset to be processed, which is known as the term of Big Data. The main challenge faced is the computational inefficiency of classical machine learning algorithms processing data of such huge volume. The common method to respond to this challenge is to layout these algorithms in Cloud [1]. With the aid of computing power of Cloud, such as Map-Reduce framework of google [2], these problems have been resolved to some extent. But, the storage and management of data in distributed heterogeneous networks have also brought about some new problems, such as the modification of canonical algorithms applied for the cloud framework[3], the reliability and error tolerance of cloud system [4], and the safety of privacy data in cloud computing [5–8]. All these problems are not easy tackled and become hot topics in computer science today.

Are there other approaches to deal with this challenge? Physicists present another perspective. By exploiting the superposition and entanglement properties of quantum state, quantum information processing technology can compress the representation space of data exponentially and speed up classical algorithms inherently[9]. The idea of doing machine learning task fused into quantum computational properties seems to be promising. Benefited from the works of Llyod [10–12], Wiebe [13, 14] and other pioneers [15–22] , this idea attracts more attention and gradually develops into a crossover research field — quantum machine learning.

In this paper, we present a quantum version for a concrete machine learning algorithm — k-nearst neighbors algorithm (KNN). KNN algorithm is an important and basic algorithm for classification, and also a sub-routine in various complicated machine learning algorithm.

The following parts are organized as follows: In Section 2, we will depict some realted works firstly, involving the classical KNN algorithm, the virtue of quantum machine learning, esp. on the high efficiency of distance computing, and the trick of computing Hamming distance. And then, we will describe the proposed quantum KNN algorithm (QKNN) in Section 3. In Section 4, we will discuss the time performance and classification accuracy of QKNN compared to analogous works presented by Llyod [11] and Wiebe [13]. Finally, we will draw the conclusion and give our insight of quantum machine learning.

2 Related Works

2.1 K-Nearest Neighbors Algorithm

KNN is a commonly used algorithm for supervised machine learning. The idea of its working mechanism is very simple: Give a testing sample, find its k nearest neighbors based on some distance metric, and then determine its category according to the information of these

neighbors. As a general rule, the algorithm uses "majority voting" to the end. That is, the testing sample is labeled as the leading category tag of its k nearest neighbors.

Figure 1 illustrates the principle of this algorithm. When $k = 1$, the testing sample (indicated as a question mark) is labeled as the category "blue star". When $k = 3$ and $k = 5$, the testing sample is labeled as the majority category "red triangle". Obviously, k is a very important factor. The classification result is different as we set k to different value. We have to note that if we set $k = 1$, KNN will be degenerated into the closest neighbor algorithm. This simplified assumption does not work well in practice, esp. in the Big Data scenario. Because in large data sets of the real world, unavoidable outliers (for example, polluted data) may lead to faulty judgment. In contrast, "majority voting" tends to be effective from a statistical view.

2.2 The Virtue of Quantum Machine Learning

The virtues of Machine Learning fused into quantum properties (Quantum Machine Learning, QML) are embodied in the following two aspects:

One is the storage scale (represention space) can be reduced exponentially by exploiting the superposition properity of quantum states. For example, a n qubit state $|\phi_1\phi_2\cdots\phi_n\rangle$ can be written as

$$|\phi_1\phi_2\cdots\phi_n\rangle = \sum_{i=0}^{2^n-1} c_i|i\rangle \qquad st.\ \sum |c_i|^2 = 1 \tag{1}$$

This equation shows that, in the quantum computer, all the binary number from the set $\{0, 1, ..., i, ..., 2^n - 1\}$ coexists in a n qubit quantum register with the corresponding probability $|c_i|^2$. However, in a classical computer, a n bit register can just store one number from the set $\{0, 1, ..., 2^n - 1\}$ exclusively.

The other virtue is the acceleration of algorithm execution speed by quantum parallelism. Also inherited from the superposition property of quantum state, the unitary evolution of the closed quantum systems can operate on each item of a quantum superposition state

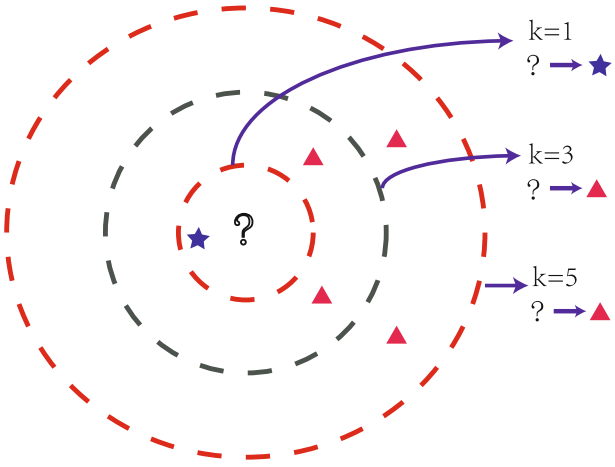

Fig. 1 The illustration of KNN algorithm

simultaneously. As shown in (2), supposing unitary operator U_f realizes the computing function $f(x)$, then U_f has accomplished all the computing task $f(x)$ by inputting each variable x from $|00\cdots0\rangle$ to $|11\cdots1\rangle$ simultaneously. In classical algorithm, the computing of $f(x)$ from the same scale of input variables x needs to be done by 2^n cycles or to be done by the parallel work of 2^n CPUs.

$$U_f\left(\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\right)=\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}U_f|x\rangle=\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|f(x)\rangle \tag{2}$$

Obviously, the reduction of data space and the acceleration of algorithm execution speed appear to be especially important and meaningful in Big Data scenario. But we have to note, in general, there is no effective manner to retrieve all the $f(x)$ after one run of the quantum algorithm. The only way to get useful information from a quantum state is measurement, which will lead to the collapse of quantum state and loss of most execution results of the quantum algorithm. If we want to get all the $f(x)$, we must run the algorithm many times and do measurement many times. So, how to retrieve useful information from a quantum state with high performance is a key issue and needs real ingenuity.

Buhrman takes good advantage of quantum parallelism, proposes a quantum computing trick of calculating the distance of two vectors with high performance [23]. Figure 2 illustrates this trick, where $|0\rangle$ is an auxiliary qubit, through left H gate, it will be changed to $\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$. Then under the control of $|1\rangle$, the circuit will swap two vectors $|x\rangle$ and $|y\rangle$, i.e. $|xy\rangle\rightarrow|yx\rangle$. Finally, we can get (3) at the right end of the circuit:

$$|0_{anc}\rangle|x\rangle|y\rangle\rightarrow\frac{1}{2}|0_{anc}\rangle\left(|xy\rangle+|yx\rangle\right)+\frac{1}{2}|1_{anc}\rangle\left(|xy\rangle-|yx\rangle\right) \tag{3}$$

If we measure auxiliary qubit alone, then the probability of final state in the ground state $|0\rangle$ is:

$$P(|0_{anc}\rangle)=\frac{1}{2}+\frac{1}{2}|\langle x|y\rangle|^2 \tag{4}$$

$|\langle x|y\rangle|$ is named as *fidelity* in quantum information theory and also named as *cosine distance* in classical machine learning. Obviously, if $|x\rangle$ and $|y\rangle$ have maximum distance, i.e. orthogonality, this probability is $\frac{1}{2}$; if $|x\rangle$ and $|y\rangle$ has minimum distance, i.e. overlap, this probability is 1. We should notice that the estimation of this probability is irrelevant to the dimension of the vectors. The higher the dimension of the vector, the higher efficiency of the quantum solution is. Lloyd points out even the cost of quantum state preparation were considered, the performance of computing distance by this quantum trick would be more efficient than classical manner [11].

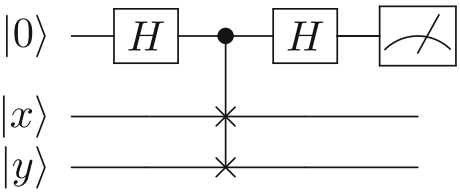

Fig. 2 Swap test for distance computing

Based on this trick, popular distance metric Euclidean distance can be easliy computed ($Euclidean\ distance = \sqrt{(2 - 2|\langle x|y\rangle|)}$). And because distance metric is the core of similarity measurement, this trick is used widely in quantum machine learning. Wiebe takes advantage of this trick, and also uses two existing algorithm [24, 25] to implement a simplified version for KNN algorithm — nearest neighbor algorithm [13]. Lloyd also use this trick to implement a variant KNN algorithm — nearest centroid algorithm [11]. In the quantum support vector machine realized by Rebentrost et al., they also use this trick to transform the computing of kernel matrix to computing the partial trace of density matrix [21].

2.3 Computing Hamming Distance

Inspired by the aforementioned trick and the work of Schuld [22], we design a quantum KNN algorithm based on the metric of Hamming distance. In order to articulate our algorithm, as the background, we review the definition of Hamming distance and introduce a quantum circuit adopted to comupte this distance here.

Hamming distance is defined as counting the number of positions at which the corresponding symbols of two bit vectors of equal length are different. For example, the Hamming distance: 0110 $\leftrightarrow$ 0001 has a distance of 3; while the Hamming distance: 0110 $\leftrightarrow$ 1110 has a distance of 1. This may look unnatural in determining the similarity of two natural feature vectors. But in practice, Hamming distance is used widely in document classification, image classification, etc. [26–28]. Mapping a natural vector to a bit vector by well-defined hash function, simple KNN classifiers in Hamming space are competitive with sophisticated discriminative classifiers, including SVMs and neural networks [27].

In quantum machine learning algorithm, if we map the features of object to the ground quantum states in Hilbert space, then it is easier to select k nearest neighbors of the testing sample by computing Hamming distance among these states, which will improve the performance further by avoiding some time-consuming operations in manipulating general quantum states such as tomography or phase estimation. This assertion will be externalized after we describe the whole algorithm. Here, we first depict an incremental circuit, which is adopted as the core module to compute Hamming distance in our algorithm.

The circuit is proposed by Kaye [29] (shown as Fig. 3). It realizes the incrementation operation of a number a, i.e. $a = a + 1$. The number a is described as a[0..n-1]. And incrementation by 1 means the flipping from the least significant qubit. If a[i] flips from 1 to 0,

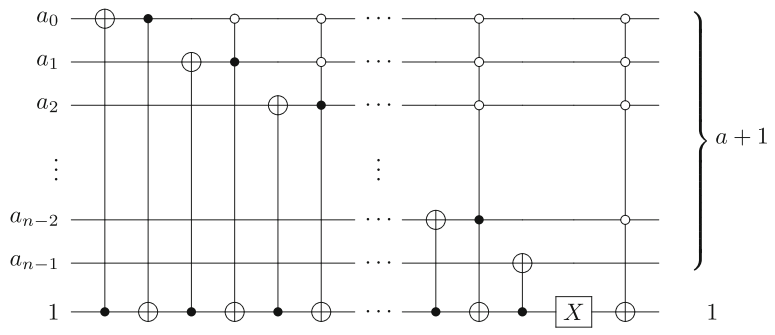

Fig. 3 Quantum $a + 1$ circuit

the addition would continue. If a[i] flips from 0 to 1, which means no carry qubit is produced, the addition should stop. The ancillary qubit in the circuit can be viewed as a "flag" which signals the first time a qubit flips from 0 to 1. It should be reset to 1 for the next run of addition. The work flow of this circuit can be depicted as the following pseudo-code:

```
i = 0;
Do
    if a[i]==1 then {
        a[i]: 1 → 0;
        i++;
    }
    else
        a[i]: 0 → 1;
Until (a[i]: 0 → 1)
```

3 Quantum K-Nearest Neighbors Algorithm

With the knowledge of the previous section, we can describe the QKNN algorithm now. This algorithm is intended to determine a testing sample belongs to which class in the training set.

3.1 Preliminary Steps

Step 1: Extract the feature of the training set and store them as bit vectors by exploiting the method of Ref [27],

Step 2: Map and parepare these bit vectors to quantum ground state straightforwardly, i.e. $0 \rightarrow |0\rangle$ and $1 \rightarrow |1\rangle$.

After these two steps, the training set is indicated by N feature vectors $|v^p\rangle$, $p = 1, ..., N$ and the corresponding class $c^p \in \{1, ..., l\}$, which can be written as $\{|v_1^p ... v_n^p, c^p\rangle\} \in \mathbb{H}_2^{\otimes n} \otimes \mathbb{H}_l$.

Step 3: Construct a training set superposition.

$$|\mathcal{T}\rangle = \frac{1}{\sqrt{N}} \sum_p |v_1^p ... v_n^p, c^p\rangle \tag{5}$$

3.2 Main Steps

Step 1: Input a testing sample, taking the same method to transform it to a quantum state $|x\rangle$ (a normalized n-dimensional feature vector $|x_1 ... x_n\rangle$).

Step 2: Prepare the unclassified quantum state $|x_1 ... x_n\rangle$ in the first register, prepare the training set $|\mathcal{T}\rangle$ in the second register, prepare an ancillary qubit $|0\rangle$ in the last register. The result can be written as $|\phi_0\rangle$.

$$|\phi_0\rangle = \frac{1}{\sqrt{N}} \sum_p |x_1 ... x_n; v_1^p ... v_n^p, c^p; 0\rangle \tag{6}$$

Step 3: Record the difference between $|x_1...x_n\rangle$ and each $|v_1^p...v_n^p\rangle$ in the training set, store the result $|d_1^p...d_n^p\rangle$ in the first register and reverse the value. For example, if $|x_1...x_n\rangle = |0010\rangle$, $|v_1^p...v_n^p\rangle = |1010\rangle$, then the final result $|d_1^p...d_n^p\rangle = |0111\rangle$. The reason that we reverse the value is just a mathematical trick. Its function will be demonstrated in the following steps.

$$|\phi_2\rangle = \prod_k X(x_k)CNOT(x_k, v_k^p)|\phi_1\rangle = \frac{1}{\sqrt{N}}\sum_p |d_1^p...d_n^p; v_1^p...v_n^p, c^p; 0\rangle \quad (7)$$

Where, CNOT(a, b)-gate overwrites the first entry a with 0 if $a = b$ and else with 1. X gate is used to reverse the value.

Step 4: In this step, we will compute Hamming distance in terms of $|d_1^p...d_n^p\rangle$, and label the k-nearest neighbors of the testing sample (modify the corresponding ancillary qubit, $|0\rangle \rightarrow |1\rangle$) according to a distance threshold value t. The operation of this step can be defined as a certain unitary operation U, which realizes:

$$|\phi_3\rangle = U|\phi_2\rangle = \frac{1}{\sqrt{N}}\left(\sum_{p\in\Omega}|d_1^p...d_n^p; v_1^p...v_n^p, c^p; 1\rangle + \sum_{p\notin\Omega}|d_1^p...d_n^p; v_1^p...v_n^p, c^p; 0\rangle\right) \quad (8)$$

Where set Ω contains certain indexes p which indicates the Hamming distance between $|x\rangle$ and the p^{th} sample in the training set $\leq t$.

In step 3, each d_i^p has recorded the difference at the corresponding position i. Hence, the Hamming distance can be computed as $\sum_i d_i^p$. As we recall, In Section 2, we introduce the quantum circuit of $a = a + 1$. Then the key step for adding d_i^p can be realized by taking advantage of this circuit as Fig 4a, which is reduced as Fig 4b.

Taking inC_k as the core module, then $\sum_i d_i^p$ can be gotten by invoking this module n times. As we also recall, in step 3, the value of d_i^p is reversed, so $Hamming\ distance \leq t$ can be depicted as follows:

$$\sum_i d_i^p \geq n - t \quad (9)$$

Suppose $2^{k-1} \leq n \leq 2^k$, if we set a variable $l = 2^k - n$, then the condition $Hamming\ distance \leq t$ can be finally derived as:

$$\sum_i d_i^p + l \geq n + l - t \Longrightarrow \sum_i d_i^p + l + t \geq 2^k \quad (10)$$

This equation means that if we set initial $a = l + t$, then the condition of $Hamming\ distance \leq t$ can be determined by whether the addition of $\sum_i d_i^p + a$ overflow or not. Once the addition is done, we select the $\lceil logt \rceil$ most significant qubits and use quantum OR gate to get the signal $COND^p$, which indicates the condition of $Hamming\ distance \leq t$. The quantum OR gate is shown in Fig. 5 and the overall circuit is shown in Fig. 6.

Step 6: Define a projection operator $\Gamma = \mathbb{1} \otimes |1\rangle\langle 1|$, apply it on $|\phi_3\rangle$, and renormalize the result, we can get:

$$|\phi_4\rangle = \Gamma|\phi_3\rangle = \alpha\sum_{p\in\Omega}|d_1^p...d_n^p; v_1^p...v_n^p, c^p; 1\rangle \qquad st. \sum|\alpha|^2 = 1 \quad (11)$$

α is the renormalized amplitude of each component in $|\phi_4\rangle$. Their values are same because projection does not change the uniform superposition property of $|\phi_3\rangle$.

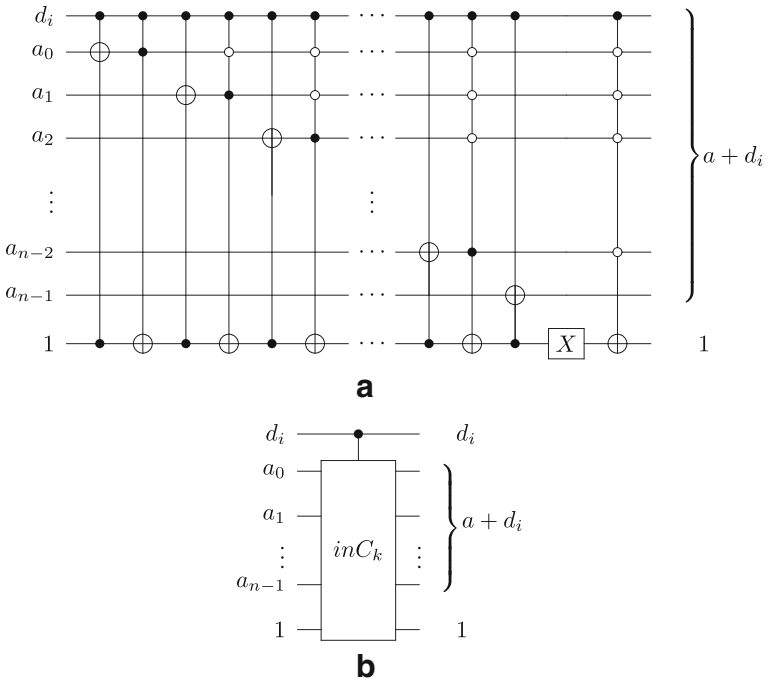

Fig. 4 The quantum circuit of $a + d_i$

Now, $|\phi_4\rangle$ is composed of $|v^p\rangle$ whose distance are no more than t to the testing sample $|x\rangle$. Measure c^p alone, we can get the category to which $|x\rangle$ belongs.

4 Discussion

4.1 Performance Analysis

This algorithm includes two parts: preliminary steps and main steps. In preliminary steps, because each sample in the training set should be extracted features and mapped to quantum state, the time performance of these steps is positively correlated with the number of samples in the traning set, at least $O(N)$. But it should be noted the aim of these steps is to generate the training set superpostion, which does not change in the main steps. That means we just execute these steps once before the first run of this algorithm. So, the time cost of preliminary steps can be ignored with the repeated execution of the algorithm.

In the main part, each step is executed in quantum parallelism, so the time cost is irrelevant to the training set size. It is only relevant to the dimension of the feature vectors n. In these steps, the step 4 computing Hamming distance is the most time-consuming operation. The cost of the core module (Fig. 4) is depicted as the following equation, which is measured by the number of "elementary gates" $\{NOT, CNOT, Toffoli\}$ [29].

$$the\ cost\ of\ inC_k = \begin{cases} 1, & k = 1 \\ 10, & k = 2 \\ 2k^2 + k - 5, & k \geq 3 \end{cases} \tag{12}$$

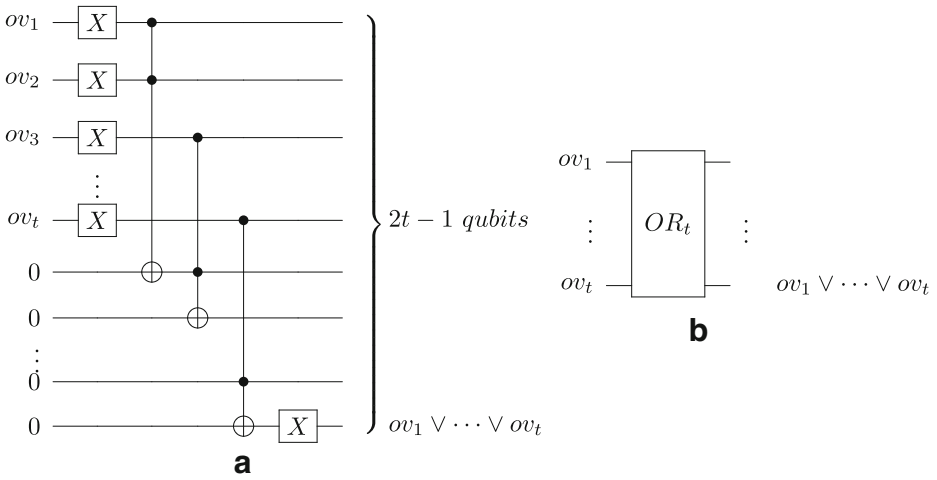

Fig. 5 The quantum OR circuit

In Fig. 6, the sub-circuit that generates the $\lceil logt \rceil$ largest qubits invokes module inC_k n times. The sub-circuit that generates $CONDD^p$ has $\lceil logt \rceil + 1$ NOT gates and $\lceil logt \rceil - 1$ Toffoli gates. Hence, the cost of the total circuit is:

$$n(2n^2 + n - 5) + (\lceil logt \rceil + 1) + (\lceil logt \rceil - 1) \tag{13}$$

The last step is to determine the category to which the testing sample belongs by measuring the c^p alone. As we know, $|\phi_4\rangle$ has k items (k-nearest neighbors) and $k \ll N$ in general. So the time cost can be ignored compared to big N. The total time cost of this algorithm is $\sim O(n^3)$.

Compared to Lloyd algorithm [11] Wiebe's algorithm [13], because they use similar processes like quantum amplitude estimation or Grover algorithm to get the final result [24, 25, 30], the time cost is relevant to the training set size N. The former is $O(logNn)$, the latter is $O(\sqrt{N}logN)$ (require feature vectors are sparse). In Big Data scenario, the dimension n is far less than the dataset size N. Hence, Our algorithm has a huge performance benefit.

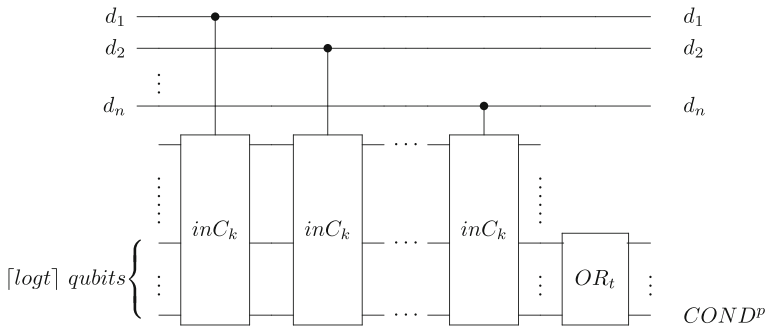

Fig. 6 The quantum circuit to generate $CONDD^p$

4.2 Classification Accuracy

Whether Lloyd algorithm [11] or Wiebe's algorithm [13], they may both lead to inaccurate classification for the simplification of the application scenario. Wiebe's algorithm simplifies the classification of the testing sample only in terms of its nearest neighbor. As Fig. 1 shows, the nearest neighbor "blue star" is an outlier indeed. In such case, Wiebe's algorithm will lead to an error. Lloyd algorithm uses the distance to nearest-centroid (central vector of the cluster) to determine the category, but nearest-centroid may distort the classification result. Imagine that the cluster {A} is dense but cluster {B} is sparse. Then even if $|x - mean(A)| \leq |x - mean(B)|$, it may be much more likely that the testing vector x should be assigned to B because the probability of a large deviation from the centroid is much greater for {B} than {A}.

Different from the aforementioned two algorithms, QKNN is a complete analog of the original KNN algorithm. To illustrate the point, we reduce the final state $|\phi_4\rangle$ to a simple form by ignoring other parts:

$$|\phi_4\rangle = \alpha_1|v^1, c^1\rangle + \alpha_2|v^2, c^2\rangle + ... + \alpha_k|v^k, c^k\rangle \quad \sum |\alpha_i|^2 = 1 \tag{14}$$

The remaining work is to determine which c^i is the major category amongest k nearest neighbors. Measuring c^i alone, the probabilty of getting category p is $\sum_{i\in\{c^i=p\}} |\alpha_i|^2$. Obviously, amongest $k - nearest$ neighbors, more $|v^i\rangle$ belongs to category p, bigger this probability is. So, the measurement probability of getting the final classification result is a good analog for "majority voting" of original KNN algorithm.

The above discussion shows that QKNN should be having a higher classification accuracy. This assertion is confirmed on the numerical experiments on real dataset MNIST http://yann.lecun.com/exdb/mnist/. The MNIST digit database is a benchmark image dataset of ten handwritten digits (0 to 9). To QKNN, we use binary hash function presented in Ref. [27] to map raw MNIST images to 64-bit codes and set $k = 20$ (20 nearest neighbors) to classify handwritten digits. Then, we select 10%, 20%, ..., 90% images as training set and

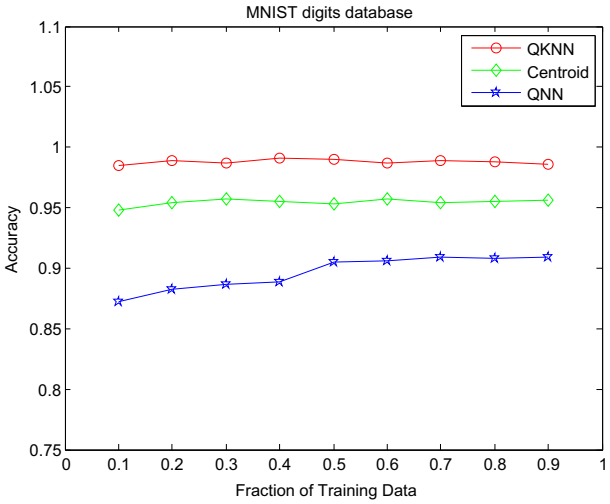

Fig. 7 The classification accuracy of QKNN, Centroid and QNN

the rest images as testing images. From the results as Fig. 7 shows, QKNN outperforms Centroid and QNN as we estimated.

5 Conclusion

In this paper, we presented a quantum algorithm for performing k-nearest neighbors (QKNN) classification based on the metric of Hamming distance. First, we developed a method for computing Hamming distance inspired by the quantum circuit presented by Kaye. And owing to this method, the performance of QKNN can be highly improved in Big Data scenario because the time cost is only relevant to the dimension of feature vectors and is irrelevant of dataset size. Second, QKNN is a good analog for classical KNN algorithm, which avoid the defect of the simplified assumption of the task of classification. In summary, QKNN outperforms Centroid and QNN on time performance and classification accuracy.

Quantum machine learning presents a new perspective, sheds light on the new possibility of doing classification or other traditional machine learning tasks with high performance. QKNN is a step toward this approach. Beyond computational speedups, a more interesting question is whether quantum computation allows new classes of learning algorithms that do not have natural classical analogs. The answer to this question may reveal potentials and limitations of quantum physics on the ability to learn from data and bring wider/deeper application of quantum machine learning.

Acknowledgements This work is supported by the National Natural Science Foundation of China (Grant No. 61170321,61502101), Natural Science Foundation of Jiangsu Province, China (Grant No. BK20140651), Natural Science Foundation of Anhui Province, China (Grant No. 1608085MF129, 1708085MF162), Foundation for Natural Science Major Program of Education Bureau of Anhui Province (Grant No. KJ2015ZD09) and the open fund of Key Laboratory of Computer Network and Information Integration in Southeast University, Ministry of Education, China (Grant No. K93-9-2015-10C).

References

1. Armbrust, M., Fox, A., Griffith, R., et al.: A view of cloud computing. Commun. ACM. **53**(4), 50–58 (2010)
2. Schölkopf, B., Platt, J., Hofmann, T.: Map-Reduce for machine learning on multicore. In: Proceedings of the 2006 Conference Advances in Neural Information Processing Systems 19, pp. 281–288. MIT Press (2007)
3. Low, Y., Bickson, D., Gonzalez, J., et al.: Distributed GraphLab: a framework for machine learning and data mining in the cloud. Proceedings of the VLDB Endowment **5**(8), 716–727 (2012)
4. Liu, Q., Cai, W., Shen, J., et al.: A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment. Security Commun. Netw. **9**(17), 4002–4012 (2016)
5. Xia, Z., Wang, X., Zhang, L., et al.: A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. IEEE Trans. Inf. Forensics Secur. **11**(11), 2594–2608 (2016)
6. Fu, Z., Ren, K., Shu, J., et al.: Enabling personalized search over encrypted outsourced data with efficiency improvement. IEEE Trans. Parallel Distrib. Syst. **27**(9), 2546–2559 (2016)
7. Fu, Z., Sun, X., Liu, Q., et al.: Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEICE Trans. Commun. **E98-B**(1), 190–200 (2015)
8. Xia, Z., Wang, X., Sun, X., et al.: A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans. Parallel Distrib. Syst. **27**(2), 340–352 (2015)

9. Nielsen, M.A., Chuang, I.L.: Quantum computation and quantum information [M]. Cambridge University Press, Cambridge (2000)
10. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum principal component analysis. Nature Physics. **10**(9), 631–633 (2014)
11. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411 (2013)
12. Lloyd, S., Garnerone, S., Zanardi, P.: Quantum algorithms for topological and geometric analysis of data. Nat. Commun. **7**, 10138 (2016)
13. Wiebe, N., Kapoor, A., Svore, K.: Quantum nearestneighbor algorithms for machine learning. arXiv:1401.2142 (2014)
14. Wiebe, N., Granade, C., Ferrie, C., et al.: Quantum Hamiltonian learning using imperfect quantum resources. Phys. Rev. A **89**(4), 042314 (2014)
15. Trugenberger, D.A.: Quantum pattern recognition. Quantum Inf. Process. **1**(6), 471–493 (2002)
16. Aïmeur, E., Brassard, G., Gambs, S.: Machine learning in a quantum world. In: Advances in Artificial Intelligence Lecture Notes in Computer Science, vol. 4013, pp. 431–442 (2006)
17. Hentschel, A., Sanders, B.C.: Machine learning for precise quantum measurement. Phys. Rev. Lett. **104**(6), 063603 (2010)
18. Gammelmark, S., Mølmer, K.: Quantum learning by measurement and feedback. New J. Phys. **11**, 033017 (2009)
19. Bisio, A., Chiribella, G., D'Ariano, G.M., et al.: Optimal quantum learning of a unitary transformation. Phys. Rev. A **81**(3), 032324 (2010)
20. Lu, S., Braunstein, S.L.: Quantum decision tree classifier. Quantum Inf. Process. **13**(3), 757–770 (2014)
21. Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum support vector machine for big data classification. Phys. Rev. Lett. **113**(13), 130503 (2014)
22. Schuld, M., Sinayskiy, I., Petruccione, F.: Quantum computing for pattern classification. In: Proceedings 13th Pacific Rim International Conference on Artificial Intelligence (PRICAI), pp. 208–220 (2014)
23. Buhrman, H., Cleve, R., Watrous, J., et al.: Quantum fingerprinting. Phys. Rev. Lett. **87**(16), 167902 (2001)
24. Brassard, G., Høyer, P., Mosca, M., et al.: Quantum amplitude amplification and estimation. arXiv:0005055[quant-ph/] (2000)
25. Dürr C., Høyer, P.: A quantum algorithm for finding the minimum. arXiv:9607014[quant-ph/] (1996)
26. Manevitz, L.M., Yousef, M.: One-class SVMs for document classification. J. Mach. Learn. Res. **2**(12), 139–154 (2001)
27. Norouzi, M., Fleet, D.J., Salakhutdinov R.R.: Hamming distance metric learning [C]. In: Advances in Neural Information Processing Systems (NIPS). 061–1069 (2012)
28. Rai, H., Yadav, A.: Iris recognition using combined support vector machine and Hamming distance approach. Expert Syst. Appl. **41**(2), 588–593 (2014)
29. Kaye, P.: Reversible addition circuit using one ancillary bit with application to quantum computing. arXiv:0408173v2[quant-ph] (2004)
30. Grover, L.K.: Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett. **79**(2), 325–328 (1997)