Quantum k-nearest neighbors algorithm

Afrad Basheer,[1,2,*] A. Afham,[1,3,†] and Sandeep K. Goyal[3,‡]

[1] *Centre for Quantum Software, University of Technology, Sydney, Australia*
[2] *Chennai Mathematical Institute, H1 SIPCOT IT Park, Kelambakkam, Tamil Nadu 603103, India*
[3] *Department of Physical Sciences, Indian Institute of Science Education & Research (IISER) Mohali, Sector 81 SAS Nagar, Manauli PO 140306 Punjab India.*

(Dated: May 6, 2021)

One of the simplest and most effective classical machine learning algorithms is the k-nearest neighbors algorithm (kNN) which classifies an unknown test state by finding the k nearest neighbors from a set of M train states. Here we present a quantum analog of classical kNN – quantum kNN (QkNN)– based on fidelity as the similarity measure. We show that QkNN algorithm can be reduced to an instance of the quantum k-maxima algorithm, hence the query complexity of QkNN is $O(\sqrt{kM})$. The non-trivial task in this reduction is to encode the fidelity information between the test state and all the train states as amplitudes of a quantum state. Converting this amplitude encoded information to a digital format enables us to compare them efficiently, thus completing the reduction. Unlike classical kNN and existing quantum kNN algorithms, the proposed algorithm can be directly used on quantum data thereby bypassing expensive processes such as quantum state tomography. As an example, we show the applicability of this algorithm in entanglement classification and quantum state discrimination.

I. INTRODUCTION

Quantum machine learning (QML) [1–4] is a recent offspring of quantum computing and machine learning (ML). One of the proposed applications of QML is using quantum computers to speed up ML tasks [5, 6]. In a sort of converse manner, ML has proven itself adept at problems in physics [7–10]. We have also seen the realisation of quantum versions of several (classical) ML algorithms [11–13]. Along the same vein, we propose a quantum version of the k-nearest neighbor algorithm [14] in this paper. We also present two problems of interest to the quantum computing community as an application of our algorithm, namely entanglement classification and quantum state discrimination.

Our algorithm is essentially a k-maxima finding algorithm [15] to find the k states (from a total of M given states) which have the maximum fidelity (or dot product) with the state to be classified. We provide an explicit construction of the oracle required to perform the k-maxima finding algorithm. We require the information regarding the states to be classified (test states) and supplied data (train states) in the form of circuits capable of preparing these states.

We envision our algorithm to be a direct quantum analogy to the (classical) kNN in case of classical data - supply the circuits and obtain the identities of the k nearest states. We require no classical description (writng out the amplitudes in some basis) of the states and thereby circumvent expensive processes such as tomography, which would be required if one were to do a classical kNN on quantum data. Moreover, since we use Grover search to find the k-maxima set, the complexity of our algorithm is $O(\sqrt{kM})$ providing a quadratic speedup over any classical kNN algorithm.

kNN algorithm is a simple supervised ML algorithm used extensively for pattern recognition and classification [14]. This algorithm rests on the assumption that two points close to each other are more likely to be of the same type. In this algorithm, the computer is provided with a set of train states (vectors) whose class labels are known. The test state (vector) with the unknown label is compared with the train states, and the k number of nearest neighbors of the train states are identified for the given test state. The label of the test state is determined upon majority voting.

The expensive step in the kNN algorithm is to determine the distance between the test state and all the train states. Each state (train or test) is represented by a vector of real (or complex) numbers. As the number of train states and the size of the state vectors increases, kNN becomes more expensive. To classify vectors of dimension N by comparing it to a set of train vectors of cardinality M, we need to carry out $O(MN)$ operations. This gives the classical kNN algorithm a complexity of $O(MN)$.

Previous work in quantum versions of kNN include [16–20]. Ruan et al. [17] uses Hamming distance as the metric to estimate the k nearest neighbors. Dang et al. [19] uses quantum k nearest neighbors algorithm proposed in [18] for image classification. Of all the previous papers mentioned, the work by Chen et al. [18] seems to be the closest to ours with the same overall query complexity of $O(\sqrt{kM})$.

However, there are a few features that distinguishes our work from the above mentioned works. Firstly, we envision our algorithm to be used directly on quantum data (though it can also be used on 'classical data'), which allows us to classify states without having their explicit classical description. Instead, we require the circuits ca-

* Afrad.M.Basheer@student.uts.edu.au
† afham@student.uts.edu.au; afham.a.acad@gmail.com
‡ skgoyal@iisermohali.ac.in

pable of preparing the test state and train states. We also use fidelity and dot product as a measure of similarity and we demonstrate problems where fidelity can be used to carry out classification.

In this article, we propose a novel quantum k-nearest neighbor (QkNN) algorithm, a quantum analog of classical kNN algorithm. Utilising existing algorithms such as the Swap test [21], k-minimum finding algorithm [22] by Durr et al., quantum phase estimation, and the recent quantum analog-to-digital conversion algorithm [23], we construct an algorithm capable of classifying states without the requirement of having a classical description of it. This is particularly useful in cases where the data to be classified are inherently quantum and thereby we can bypass expensive processes such as tomography [24] to learn the description of state in question. We also provide two applications of our algorithm, namely entanglement classification of pure states and quantum state discrimination.

In Section III, we present our quantum k nearest neighbors algorithm which uses fidelity as a measure of closeness. Then, in Section IV, we present a variation of the algorithm that uses dot product instead of fidelity for classification. Both of these algorithms build upon generalisations of algorithms from [23]. Section V comprises the query complexity of our algorithm which we show to be $O(\sqrt{kM})$. We then present in Section VI two problems where our QkNN algorithm can be utilised - the problem of entanglement classification and a problem analogous to quantum state discrimination. This brings us to our second contribution. We show, through numerical experiments, that quantum states have a nearest neighbors structure as far as entanglement is concerned - nearby states have similar entanglement properties. Specifically, we considered the entanglement entropy as a measure of entanglement. This indicates that our QkNN algorithm can classify a state on the basis of its entanglement without having its explicit classical description.

Finally we conclude in Section VII while also discussing the kind of problems our algorithm would be adept at solving.

II. BACKGROUND

A. Classical kNN algorithm

Let $\{u_n\}$ be a collection of vectors of unknown labels, which we call *test states*. The aim is to assign these test states' labels as accurately as possible. To do this, the k-nearest neighbors (kNN) algorithm requires a collection of vectors $\{v_m\}$ of the same dimension whose labels are known to us. We shall call these states *train states*. kNN assigns labels to each of the test state by first computing the k nearest neighbors of the test state. Then a majority voting is carried out among these k nearest neighbors Ties are resolved in different ways, such as assigning the label of the nearest training point, or the label of a random training point among the k nearest neighbors. Successful applications of include [25, 26]. Being a simple algorithm, kNN also allows us to reason about the structure of the data we are working with.

Let the test states and the trains states are r-dimensional real or complex vectors. Any bona fide definition of a distance measure can be used for the purpose of kNN algorithm. Most common distance measures include Euclidean distance $\mathrm{d}(\mathbf{u},\mathbf{v})$ and cosine similarity $\mathrm{c}(\mathbf{u},\mathbf{v})$ (which reduces to inner product for normalised states), which are defined as:

$$\mathrm{d}(\mathbf{u},\mathbf{v}) = \left(\sum_i^r |u_i - v_i|^2\right)^{1/2}, \tag{1}$$

$$\mathrm{c}(\mathbf{u},\mathbf{v}) = \frac{\langle \mathbf{u},\mathbf{v}\rangle}{\|\mathbf{u}\|\cdot\|\mathbf{v}\|}. \tag{2}$$

Here, $\mathbf{u}$ and $\mathbf{v}$ are r-dimensional complex vectors.

In quantum information theory, the fidelity function, though not a metric, $\mathrm{F}(\cdot,\cdot)$ can be used to assign a notion of nearness between two quantum states beloning to the same space. For arbitrary quantum states ρ,σ belonging to the same space,the fidelity between them is

$$\mathrm{F}(\rho,\sigma) = \mathrm{Tr}\left(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right) \tag{3}$$

For pure quantum states $\rho = |u\rangle\langle u|, \sigma = |v\rangle\langle v|$, the fidelity function simplfies to

$$\mathrm{F}(u,v) = |\langle u|v\rangle|^2. \tag{4}$$

Even though the fidelity is not a metric, one can see that the metric Bures distance, defined over quantum states,

$$\mathrm{B}(u,v) = 2\left(1 - \sqrt{\mathrm{F}(u,v)}\right), \tag{5}$$

is a monotonous function of the fidelity. Therefore, finding the k-nearest neighbors of any quantum state with respect to the Bures distance is same as finding the k states with the largest fidelity to the chosen quantum state. The rationale behind kNN is that data points that are close together, with respect to some distance measure, must be similar. Formally, the kNN algorithm consists of the following steps (see Figure (1)):

1. For each test state (whose label is to be determined), compute its distance to the train states whose labels are known.
2. Choose the k number of neighbors which are nearest to the test state.
3. Assign the label using majority voting.

Although the kNN algorithm is simple to understand and easy to implement, the algorithm has its drawbacks. As the number of train data points and the dimension of

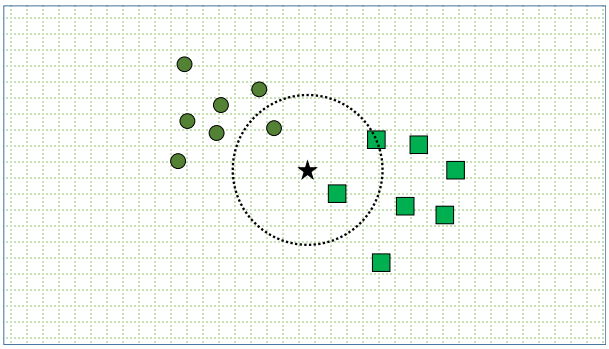

FIG. 1. Choosing a $k = 3$ neighborhood. Here circle and square represents two different classes and star represents the unknown state whose label is to be determined. On choosing $k = 3$, we classify it as a 'square' point.

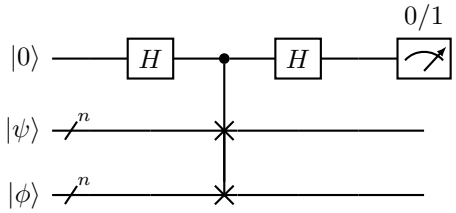

FIG. 2. Circuit for Swap test.

the state vectors grows, kNN can quickly turn intractable for classical computers. Classification of an N dimensional test state by comparing with M train states requires $O(MN)$ multiplication operations. Furthermore, there is no general way of choosing k and usually, hyperparameter tuning is done to choose the best possible k [27].

B. Quantum k-maxima finding algorithm

Durr and Hoyer describes, based on Grover search algorithm, an algorithm in [15] which can be used to find the minimum of an unsorted list of size M with complexity $O(\sqrt{M})$. Through a simple modification, one can use the same algorithm to find the maximum instead of the minimum. A generalisation of the algorithm can be found in [28], which can be used to find the k smallest elements in a table $T = [T_0, \ldots, T_{M-1}]$ of M elements in time $O(\sqrt{kM})$. A simple explanation of the algorithm can be found in [29].

The general idea of the algorithm is to start with randomly chosen k indices and use Grover search to find and replace the chosen indices with ones that have a higher table value. This process is repeated until we end up with the k highest values in the table. The algorithm is as follows:

1. Initialise a set $A = \{i_1, \ldots, i_k\}$ with randomly chosen k indices from the list of M indices.
2. Repeat the following forever:
   (a) Select threshold index y from A randomly.
   (b) Using Grover search, find index y' which is not present in A, such that $T_{y'} > T_y$. This can be seen as using Grover search on the Boolean function
   $$f_{y,A}(j) = \begin{cases} 1 & : T_j > T_y \text{ and } j \notin A \\ 0 & : \text{otherwise.} \end{cases} \tag{6}$$
   (c) Replace y with y' in the set A.

Note that step 2b is the only quantum subroutine of the algorithm. Essentially, we use Grover search to find an index not present in A and has its table value greater than the table value of the threshold index. Also, instead of randomly sampling an index from A in step 2a, one can choose the index

$$y = \underset{i \in A}{\operatorname{argmin}}\, T_i.$$

This method would ensure a stopping criterion for the algorithm, that is, repeat step 2 until step 2b cannot be carried out. This is because, if we cannot find an index, which is not in A and has a higher table value than the index with the minimum table value in A, then we should have all the k indices with the largest table values already present in A.

C. Swap Test

The swap test [21] is a quantum algorithm that can be used to statistically estimate the fidelity $\mathrm{F}(\psi, \phi) = |\langle\psi|\phi\rangle|^2$ between two arbitrary n qubit pure states $|\psi\rangle$ and $|\phi\rangle$. The three register gate in circuit 2 is the *controlled swap* (CSWAP) gate whose action is defined by

$$\begin{aligned} \text{CSWAP}|0\rangle|\psi\rangle|\phi\rangle &= |0\rangle|\psi\rangle|\phi\rangle, \\ \text{CSWAP}|1\rangle|\psi\rangle|\phi\rangle &= |1\rangle|\phi\rangle|\psi\rangle. \end{aligned} \tag{7}$$

To implement the swap test between states $|\psi\rangle$ and $|\phi\rangle$, we need three registers prepared in states $|0\rangle$, $|\psi\rangle$, and $|\phi\rangle$, respectively. The initial combined state of the three registers is $|0\rangle|\psi\rangle|\phi\rangle$. We then implement circuit 2.

At the end of the circuit, the measurement probabilities of the first register are

$$\Pr(0) = \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2, \tag{8}$$

$$\Pr(1) = \frac{1}{2} - \frac{1}{2}|\langle\psi|\phi\rangle|^2. \tag{9}$$

The quantity $\Pr(0) - \Pr(1)$ gives us the desired fidelity.

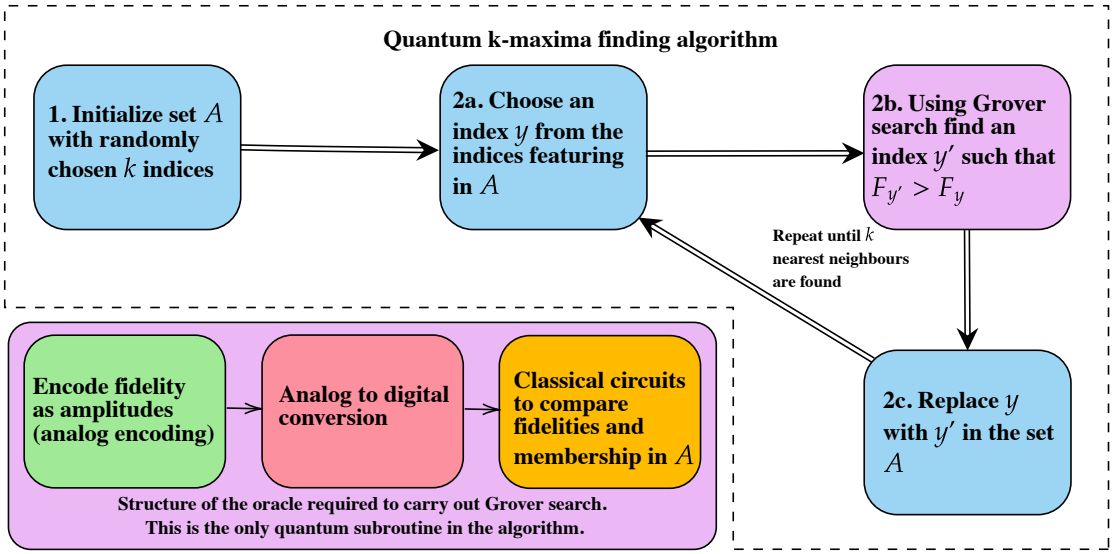

FIG. 3. An overview of the QkNN algorithm. The general approach is to use the quantum k-maxima finding algorithm to find the k nearest neighbors of the test state. The idea is to start with a set A of randomly chosen k indices and then replacing each of the k indices with the index of another train state having a higher fidelity with the test state. The crucial step here is to prepare an oracle capable of performing the required Grover search subroutine. The oracle should be capable of comparing fidelity values (pair-wise) between the test state and two arbitrary train states. Such an oracle is constructed by first extracting the fidelity using Swap test and encoding it as amplitudes of a quantum state. Then, an analog to digital conversion of the amplitudes is carried out which results in the fidelity being encoded as digital (bit-string) states. Once this is done, we perform a series of classical operations using Toffoli gates to compare the fidelities of test state with a train state whose index is in A and a train state whose index does not feature in A.

D. Quantum Analog-Digital Conversion (QADC) algotihm

Mitarai et al. describes a set of algorithms in [23] to carry out analog-digital conversions within a quantum circuit. We provide a description of what the algorithm does here and refer to Appendix (A) for more details for the sake of brevity.

Let $\sum_{i=0}^{d-1} c_i|i\rangle$ be an arbitrary quantum state. Let $\{r_0, \ldots, r_{d-1}\}$ be bitstrings that denote the best b-bit approximation of $\{|c_i|, \ldots, |c_{d-1}|\}$ respectively. An m-bit abs-QADC algorithm can transform thee *analog encoded* state $\sum_{i=0}^{d-1} c_i|i\rangle$ to the *digital encoded* state $\frac{1}{\sqrt{M}} \sum_{i=0}^{d-1} |i\rangle|r_i\rangle$.

Let $\{r_0, \ldots, r_{d-1}\}$ be bitstrings that denote the best b-bit approximation of $\{\mathrm{Re}(c_i), \ldots, \mathrm{Re}(c_{d-1})\}$ respectively. An m-bit real-QADC algorithm transforms the *analog encoded* state $\sum_{i=0}^{d-1} c_i|i\rangle$ and to the *digital encoded* state $\frac{1}{\sqrt{M}} \sum_{i=0}^{d-1} |i\rangle|r_i\rangle$.

In the coming sections, we show that a slight variation to the circuits of these algorithms give rise to circuits that form a part of circuits capable of carrying out a quantum kNN algorithm.

III. QUANTUM k NEAREST NEIGHBORS ALGORITHM USING FIDELITY

Some of the basic expectations from this algorithm is as follows:

- This algorithm should be able to find the fidelity between the test states and all the train states.
- The algorithm should be capable of performing comparison between two different fidelities.
- The algorithm should be more efficient than known algorithms.

We begin with definitions and notations. Let $\mathcal{H}$ be the n-qubit Hilbert space of dimension $N = 2^n$ and let $|\psi\rangle \in \mathcal{H}$ be the unknown *test state* whose label is to be determined. Let

$$\{|\phi_j\rangle : j \in \{0, \ldots, M-1\}\} \subset \mathcal{H} \tag{10}$$

be a collection of M *train states* whose labels are known to us. For the sake of convenience, we assume $M = 2^m$ for some positive integer m. The idea is to find the k nearest neighbors of $|\psi\rangle$ from the train states and then through majority voting, assign $|\psi\rangle$ a label. Let $F_j \equiv \mathrm{F}(\psi, \phi_j) = |\langle\psi|\phi_j\rangle|^2$ be the fidelity between the test state $|\psi\rangle$ and the j^{th} train state $|\phi_j\rangle$ and define

$$F = [F_0, \ldots, F_{M-1}] \tag{11}$$

to be a table of length M containing the fidelities with the test state $|\psi\rangle$ and all the train states $\{|\phi_j\rangle\}$.

A. A brief summary of the algorithm

Note that the problem of finding the k-nearest neighbors of a test state $|\psi\rangle$ can be reduced to an instance of k-maximum finding algorithm carried out on the table (11). The only step that requires a quantum circuit in the quantum k-maximum finding algorithm is the Grover search subroutine given in step 2b. One can see that to achieve this, we should be able to prepare the circuit that carries out the oracle transformation

$$\mathcal{O}_{y,A}|j\rangle|0\rangle = \begin{cases} |j\rangle|1\rangle & : F_j > F_y \text{ and } j \notin A \\ |j\rangle|0\rangle & : \text{otherwise.} \end{cases} \tag{12}$$

where $y, j \in \{0, \ldots, M-1\}$ are arbitrary indices. So, then QkNN algorithm using fidelity as the similarity measure is:

1. Using $\mathcal{O}_{y,A}$ as the required oracle for Grover search in step 2b, use k-maxima finding algorithm to find the k indices $\{j_1, \ldots, j_k\}$ whose states $\{\phi_{j_1}, \ldots, \phi_{j_k}\}$ have the maximum fidelity with the test state.

2. Conduct a majority voting among the k states and assign $|\psi\rangle$ the label of the majority.

The non-trivial part in the above steps is the realisation of the oracle $\mathcal{O}_{y,A}$. We briefly discuss this oracle in the next subsection and provide an explicit construction of it after that.

B. On the oracle $O_{y,A}$

Let b be the number of qubits which is required to store the binary representation of the fidelity values. Then, we require $2m+3b+2n+4$ qubits to realize this oracle. Let $f_{y,A}$ be the Boolean function defined as

$$f_{y,A}(j) = \begin{cases} 1 & : F_j > F_y \text{ and } j \notin A \\ 0 & : \text{otherwise} \end{cases} \tag{13}$$

That is, the fidelity F_j must be greater than F_y and j should not feature in the threshold index set A. The action of $\mathcal{O}_{y,A}$ may be concisely stated as

$$\mathcal{O}_{y,A}|j\rangle|0\rangle = |j\rangle|f_{y,A}(j)\rangle. \tag{14}$$

Roughly speaking, the way to construct the oracle O_y is as follows

1. Construct an operator $\mathcal{F}$ capable of the transformation

$$\mathcal{F}|j\rangle|0\rangle = |j\rangle|F_j\rangle \tag{15}$$

for arbitrary $j \in \{0, \ldots, M-1\}$. Here $|F_j\rangle$ is the computational basis state which is the (b-bit) binary representation of F_j. This step can be broken down into two.

   (a) First, we perform the transformation

$$\mathcal{E}^{\text{amp}}|j\rangle|0\rangle = |j\rangle|\Psi_j\rangle \tag{16}$$

   where $|\Psi_j\rangle$ is a state with information regarding F_j encoded in its amplitudes. We achieve this using the Swap test algorithm.

   (b) We now perform the transformation

$$\mathcal{E}^{\text{dig}}|j\rangle|\Psi_j\rangle = |j\rangle|F_j\rangle. \tag{17}$$

   This can be thought of as an analog to digital conversion as we are converting the fidelity information from amplitudes of $|\Psi_j\rangle$ to a digital format $|F_j\rangle$. We use a slightly modified version of the abs-QADC algorithm to achieve this.

   We see that $\mathcal{F} = \mathcal{E}^{\text{dig}}\mathcal{E}^{\text{amp}}$.

2. Consider two pairs of registers, *index, fidelity; index′, fidelity′* initialised as $|j\rangle_{\text{in}}|0\rangle_{\text{fid}}|y\rangle_{\text{in'}}|0\rangle_{\text{fid'}}$. Apply $\mathcal{F}$ (step 1) on each of the two pairs of registers.

$$\xrightarrow{\mathcal{F}_{\text{in,fid}}\mathcal{F}_{\text{in',fid'}}} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|y\rangle_{\text{in'}}|F_y\rangle_{\text{fid'}}. \tag{18}$$

   where $\mathcal{F}_{\text{in,fid}}$ denotes the gate $\mathcal{F}$ applied on the index and fidelity registers.

3. Now that we have our information in digital format, we may use a series of classical gates to realise the function $f_{y,A}$ (13). Note that any classical operation can be simulated in a quantum setting using Toffoli gates (refer page 29 of [30]). Let $\mathcal{C}$ denote the operator achieving (13). Uncomputing the irrelevant registers, we obtain

$$\xrightarrow{\mathcal{C},\text{uncompute}} |j\rangle|f_{y,A}(j)\rangle. \tag{19}$$

   For the sake of brevity, we do not further expand upon the form of $\mathcal{C}$, just like with other gates, here. That shall be done next.

We now provide an explicit construction of this oracle.

C. Constructing the oracle $\mathcal{O}_{y,A}$

We begin with the assumption that we are provided with circuits of *state preparation* oracles $\mathcal{V}, \mathcal{W}$ of the form

$$|0^n\rangle \xrightarrow{\mathcal{V}} |\psi\rangle, \tag{20}$$

$$|j\rangle|0^n\rangle \xrightarrow{\mathcal{W}} |j\rangle|\phi_j\rangle, \tag{21}$$

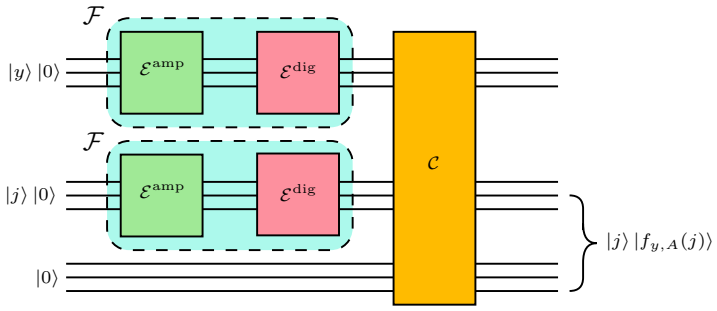

FIG. 4. A breakdown of the oracle $\mathcal{O}_{y,A}$. In two pairs of registers correspoding to index j and y, apply $\mathcal{F}$ separately to obtain digital encoding of F_j and F_y. Using classical circuits (through Toffoli gates) compare F_j and F_y, and flip an ancilla qubit from $|0\rangle$ to $|1\rangle$ if $F_j > F_y$. Uncomputing irrelevant registers, we obtain $|j\rangle|f_{y,A}(j)\rangle$. Note that in the explicit construction of the oracle, there is an uncomputation procedure of one $\mathcal{F}$ coming in between the required classical operations. But for simplicity of explaining the circuit, we have avoided it in this diagram.

for $j \in \{0, \ldots, M-1\}$. That is, we do not require the classical description of these states. Instead we require the circuits. Assuming such oracles are provided, we now describe the construction of the oracle $\mathcal{O}_{y,A}$. The construction of this oracle is based on the abs-QADC circuit from [23]. In that circuit, to compute the absolute values of the coordinates of the state, we apply Swap test in superposition with the state and standard basis vectors. In the quantum kNN setting, we apply swap test in superposition with test state and the train states. We now go on to show the correctness of such a protocol and a method to build the required oracle $\mathcal{O}_{y,A}$ using it. The method is as follows.

1. Initialise four registers named *index, train, test, B* of sizes $m, n, n, 1$ respectively, where $n = \log N$ and $m = \log M$.

$$|j\rangle_{\text{in}}|0^{\otimes n}\rangle_{\text{tr}}|0^{\otimes n}\rangle_{\text{tst}}|0\rangle_B. \tag{22}$$

2. Apply $\mathcal{W}$ on train register

$$\xrightarrow{\mathcal{W}_{\text{in,tr}}} |j\rangle_{\text{in}}|0\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}}|0\rangle_B. \tag{23}$$

3. Now apply $\mathcal{V}$ on test register to obtain

$$\xrightarrow{\mathcal{V}_{\text{tst}}} |j\rangle_{\text{in}}|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}}|0\rangle_B. \tag{24}$$

4. Apply the swap test circuit (sans measurement) between train register and test register with B as the control qubit. The state is then

$$\begin{aligned}\xrightarrow{\text{SwapTest}}\ &\frac{1}{2}|j\rangle_{\text{in}}\Big[\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} + |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|0\rangle_B \\ &+\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} - |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|1\rangle_B\Big] \\ &\equiv |j\rangle_{\text{in}}|\Psi_j\rangle_{\text{tr,tst},B},\end{aligned} \tag{25}$$

where we have defined

$$\begin{aligned}|\Psi_j\rangle = \frac{1}{2}\Big[&\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} + |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|0\rangle_B \\ &+\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} - |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|1\rangle_B\Big].\end{aligned} \tag{26}$$

Define U to be the combined unitary transformations of steps 3 and 4 (refer Figure. 5). If one now measures the register B, one would see the probabilities as

$$\Pr(B=0) = \frac{1+F_j}{2}, \tag{27}$$

$$\Pr(B=1) = \frac{1-F_j}{2}. \tag{28}$$

The information regarding fidelity is now encoded in the amplitudes. Therefore gates from steps 2-4 makes up the $\mathcal{E}^{\text{amp}}$ operator given in (1a). We must now convert it into a 'digital' format which can be further utilised.

5. To this end, construct a gate

$$G = U_{\text{tr,tst},B}\mathcal{W}_{\text{in,tr}}S_{0_{\text{tr,tst},B}}\mathcal{W}^{\dagger}_{\text{in,tr}}U^{\dagger}_{\text{tr,tst},B}Z_B, \tag{29}$$

where Z_B denotes the application of the Z gate on register B and $S_0 = \mathbb{1} - 2|0\rangle\langle 0|$ (refer Figure 6). This operator can be seen as the operator G used in the abs-QADC algorithm [23], with the CNOT gates replaced by the train data preparation oracle $\mathcal{W}$. The action of G on the current state can be written as controlled action of operators G_j:

$$G|j\rangle_{\text{in}}|\Psi_j\rangle_{\text{tr,tst},B} = |j\rangle_{\text{in}}\Big(G_j|\Psi_j\rangle_{\text{tr,tst},B}\Big), \tag{30}$$

where

$$G_j = U_{\text{tr,tst},B}S_jU^{\dagger}_{\text{tr,tst},B}Z_B, \tag{31}$$

$$S_j = \mathbb{1} - 2\Big(|\phi_j\rangle\langle\phi_j|_{\text{tr}} \otimes |0\rangle\langle 0|_{\text{tst},B}\Big). \tag{32}$$

6. $|\Psi_j\rangle_{\text{tr,tst},B}$ can be decomposed into two eigenstates of G_j, namely $|\Psi_{j+}\rangle$ and $|\Psi_{j-}\rangle$, corresponding to the eigenvalues $e^{\pm i2\pi\theta_j}$, respectively. Here, $\sin(\pi\theta_j) = \sqrt{\frac{1}{2}(1+F_j)}$ and $\theta \in [1/4, 1/2)$ (refer Appendix E). The decomposition is given as

$$|\Psi_j\rangle = \frac{-i}{\sqrt{2}}(e^{i\pi\theta_j}|\Psi_{j+}\rangle - e^{-i\pi\theta_j}|\Psi_{j-}\rangle), \tag{33}$$

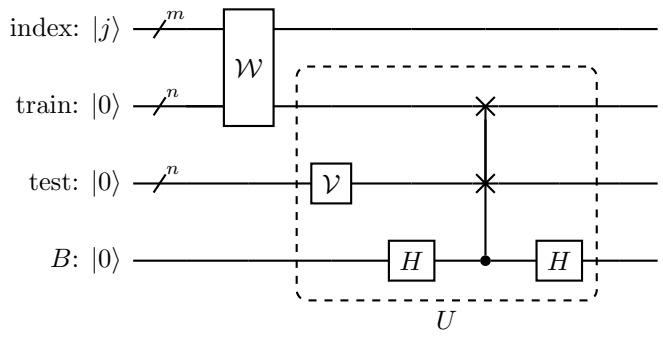

FIG. 5. Circuit for steps 1-4.

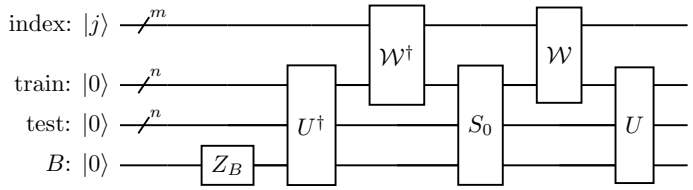

FIG. 6. Constructing the operator G as defined in (29).

7. We now have an operator G which has the fidelity value F_j stored in its eigenvalues. To get a b-bit binary representation of θ_j, we now run the phase estimation algorithm on G. To this end, we bring the *phase register* containing b qubits and run the phase estimation algorithm:

$$\xrightarrow{\text{PhaseEst.}} \frac{-i}{\sqrt{2}}|j\rangle_{\text{in}}\Big[e^{i\pi\theta_j}|\theta_j\rangle_{\text{ph}}|\Psi_{j+}\rangle_{\text{tr,tst},B} - e^{-i\pi\theta_j}|1-\theta_j\rangle_{\text{ph}}|\Psi_{j-}\rangle_{\text{tr,tst},B}\Big] \equiv |j\rangle_{\text{in}}|\Psi_{j,\text{AE}}\rangle_{\text{ph,tr,tst},B}. \tag{34}$$

where we have defined the combined state of all registers except index register after estimation to be

$$|\Psi_{j_{\text{AE}}}\rangle_{\text{ph,tr,tst},B} = \frac{-i}{\sqrt{2}}\Big(e^{i\pi\theta_j}|\theta_j\rangle_{\text{ph}}|\Psi_{j+}\rangle_{\text{tr,tst},B} - e^{-i\pi\theta_j}|1-\theta_j\rangle_{\text{ph}}|\Psi_{j-}\rangle_{\text{tr,tst},B}\Big). \tag{35}$$

Here, $|\theta_j\rangle_{\text{ph}}$ and $|1-\theta_j\rangle_{\text{ph}}$ are b-qubit states storing b-bit binary representation of θ_j and $1-\theta_j$ respectively.

8. Introducing a separate register, named *fid*, compute $F_j = 2\sin^2(\pi\theta_j)-1$ using quantum arithmetic from theorem C. Note that $\sin(\pi\theta_j) = \sin(\pi(1-\theta_j))$, and F_j is uniquely recovered. Then our total state is

$$\xrightarrow{\text{quantum arithmetics}} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|\Psi_{j,\text{AE}}\rangle_{\text{ph,tr,tst},B} \tag{36}$$

9. Uncompute everything in registers ph, tr, tst and B to get

$$\xrightarrow{\text{uncompute ph, tr, tst, } B} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}} \tag{37}$$

Now, we have successfully converted the fidelity values from amplitudes to digital format. Therefore Steps 5-9 makes up the operator $\mathcal{E}^{\text{dig}}$ given in (1b). 2-9 gives the construction of the gate $\mathcal{F}$ given in (15). We now have an operator capable of the transformation $|j\rangle|0\rangle \xrightarrow{\mathcal{F}} |j\rangle|F_j\rangle$ for arbitrary index j.

10. On separate registers, named index$'$ and fidelity$'$, initialised as $|y\rangle_{\text{in}'}|0\rangle_{\text{fid}'}$, apply $\mathcal{F}$, to obtain

$$\xrightarrow{\mathcal{F}} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|y\rangle_{\text{in}'}|F_y\rangle_{\text{fid}'} \tag{38}$$

11. Add an extra qubit Q_1 and apply the classical comparison gate

$$J|a\rangle|b\rangle|0\rangle = \begin{cases}|a\rangle|b\rangle|1\rangle & : a > b\\ |a\rangle|b\rangle|0\rangle & : a \le b,\end{cases} \tag{39}$$

on registers fid and fid$'$ to get the state

$$\xrightarrow{J} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|y\rangle_{\text{in}'}|F_y\rangle_{\text{fid}'}|g(j)\rangle_{Q_1} \tag{40}$$

where

$$g(j) = \begin{cases}1 & : F_j > F_y\\ 0 & : F_j \le F_y,\end{cases} \tag{41}$$

The qubit Q_1 will mark all indices j such that $F_j > F_y$.

12. Uncompute the registers in$'$ and fid$'$ to obtain the state

$$\xrightarrow{\text{uncompute in}', \text{ fid}'} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|g(j)\rangle_{Q_1} \tag{42}$$

13. Add an extra qubit Q_2 and for every $i_l \in A$, apply the gate $D^{(i_l)}$ of the form

$$D^{(i_l)}|j\rangle|0\rangle = \begin{cases}|j\rangle|1\rangle & : j = i_l\\ |j\rangle|0\rangle & : j \ne i_l.\end{cases} \tag{43}$$

on registers index and Q_2 to get the state

$$\xrightarrow{\left(D^{(i_1)}\cdots D^{(i_k)}\right)_{\text{in},Q_1}} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|g(j)\rangle_{Q_1}|\chi_A(j)\rangle_{Q_2} \tag{44}$$

where $\chi_A(j) = 1$ if $j \in A$ and 0 otherwise, is the indicator function of the set A. That is, the sequence of operators $D^{(i_1)}\cdots D^{(i_k)}$ marks all the indices that are already in the threshold index set

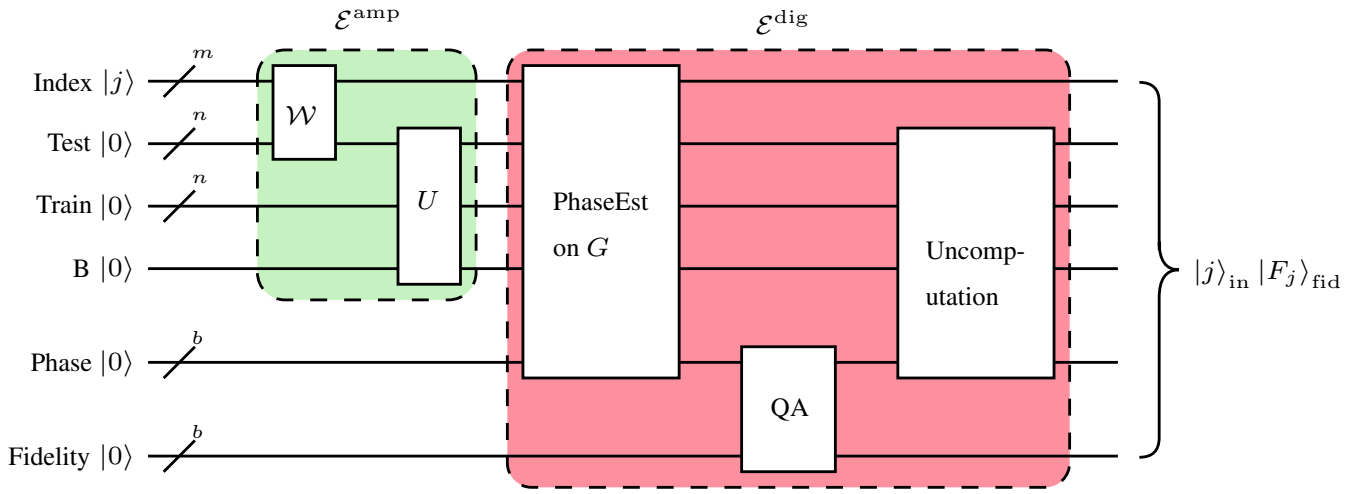

FIG. 7. Detailed construction of the operator $\mathcal{F}$ defined in (15). We first use $\mathcal{E}^{\text{amp}}$ to encode the fidelity in the amplitudes. See Figure (5) for details on U. Having the fidelity F_j encoded as amplitudes, we want to convert them into a digital format. To this end we use $\mathcal{E}^{\text{dig}}$, which comprises of phase estimation on the operator G (Eq. (29)) (refer Figure 8 for an explicit construction), which returns a state that must undergo further arithmetics (realised using quantum arithmetics) before it has a digital representation of F_j stored in the *Fidelity* register. We represent these circuits as QA. Finally, uncomputing every register except the Index and Fidelity registers, we have the required state.

A as we would like to avoid these indices so as to not have repetition in our top k neighbors. These gates can be realized using classical gates with an addition of an extra m-qubit register, which we uncomputed and recycled from step 12 (refer page 29 of [30]).

14. Add an extra qubit Q_3. Then apply an X gate on Q_2 and a Toffoli gate with controls Q_1, Q_2 and target Q_3. This results in the state

$$\xrightarrow{X,\text{Toffoli}} |j\rangle_{\text{in}}|F_j\rangle_{\text{fid}}|g(j)\rangle_{Q_1}|\chi_A(j)\rangle_{Q_2}|f_{y,A}(j)\rangle_{Q_3} \quad (45)$$

Note that ultimately we are trying to construct an oracle that should be able to mark indices j which have $g(j) = 1$ as well as $\chi_A(j) = 0$. An X gate combined with a Toffoli gate will flip the target qubit if one of the input qubits is 0 and the other is 1.

15. Uncomputing every register except index and Q_3, we have

$$\xrightarrow{\text{uncompute}} |j\rangle_{\text{in}}|f_{y,A}(j)\rangle_{Q_3} \quad (46)$$

Since the construction of this circuit does not depend on j, we now have an operator that does the aforementioned transformation.

$$\mathcal{O}_{y,A}|j\rangle|0\rangle = |j\rangle|f_{y,A}(j)\rangle. \quad (47)$$

This completes the construction of the oracle (12). We may now use this oracle in the k-maxima finding algorithm to find the k nearest neighbors of a test state based on fidelity.

IV. QUANTUM k NEAREST NEIGHBORS ALGORITHM USING DOT PRODUCT

Now we briefly discuss how one can construct a QkNN algorithm that utilises dot product $X(u, v) \equiv \langle u|v\rangle$ instead of fidelity, which is usually the case with real-world applications. Note that for QkNN using fidelity, we are performing Swap test between test state and all train states in superposition to analog-encode the fidelity information F_j, and we then use abs-QADC algorithm to digitise this information.

For QkNN using dot product, we simply replace Swap test with Hadamard test (refer Appendix D). That is we perform Hadamard test between the test state and all the train states in superposition to analog-encode the dot product information X_j, and then use a similar modification to the real-QADC algorithm to digitise this information.

The steps of this procedure are extremely similar to QkNN using fidelity, and therefore we refrain from presenting it here and instead present it in full detail in Appendix E.

V. COMPLEXITY ANALYSIS

The fidelity based QkNN requires $O(\sqrt{kM})$ calls to the oracle $\mathcal{O}_{y,A}$. The oracle contains two uses of the $\mathcal{F}$ circuit, which is a slight modification of the abs-QADC algorithm. Therefore the complexity of executing the oracle is similar to the abs-QADC algorithm. The circuit of the oracle contains $O(1/\epsilon)$ controlled $\mathcal{V}$ and $\mathcal{W}$ gates and $O((\log^2 N)/\epsilon)$ single and 2 qubit gates, where $\epsilon =$

2^{-b}. So, one can execute the whole fidelity based QkNN procedure with $O(\sqrt{kM})/\epsilon)$ calls to the data preparation oracles $\mathcal{V}$ and $\mathcal{W}$ and $O(\sqrt{kM}(\log^2 N)/\epsilon)$ single and 2 qubit gates and $O(k\sqrt{kM})$ gates $D^{(i_l)}$, each of which can be realized in $O(\log(M))$ with an extra register of m qubits. Each J gate can be realized in $O(\log b)$ and the total number of J gates required is $O(\sqrt{kM})$.

To perform $\mathcal{F}$ once, we need $m+b$ qubits and another $2n+b+1$ qubits which are uncomputed at step 9. Considering the fact that the uncomputed qubits can be recycled, applying $\mathcal{F}$ on two separate registers will require $2(m+b)+2n+b+1$ qubits. The classical operations would require 3 extra qubits. Therefore, in total, QkNN with fidelity as the similarity measure will require $2m+3b+2n+4$ qubits, as well as possible ancillary qubits required for classical computation and quantum arithmetics. In a small scale setting, to classify 3 qubit states with 128 train states, using QkNN with fidelity as similarity measure, we will require $24+3b$ qubits, where b is the required precision, along with ancillary qubits required. To classify 10 qubit states using QkNN, and $2^{20} \approx 1,000,000$ train states, we will require $64+3b$ qubits.

The dot product based QkNN requires has the same query complexity as the fidleity based QkNN. To perform this operation, one would require $2m+3b+n+4$ qubits as well as the ancillary qubits required to perform the classical operations and quantum arithmetics.

VI. APPLICATIONS

We present two scenarios where our quantum kNN algorithm can be applied. In both the scenarios, we utilise the fact that we can classify an unknown test state.

A. Entanglement classification

1. *Entanglement classes*

In this section, we discuss the entanglement classes in pure n-partite quantum states. For simplicity, we restrict ourselves to n-qubit systems. We begin with $n=2$ case. A pure two-qubit quantum state $|\Phi\rangle$ is called separable or product state if and only if it can be written as a tensor product of two pure states corresponding to individual subsystems,

$$|\Phi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle. \tag{48}$$

If the state $|\Phi\rangle$ is not of the form (48) then its an entangled state. A pure n-qubit quantum state $|\Psi\rangle$ is separable only if it can be written as the tensor product of n quantum states as

$$|\Psi\rangle = |\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle. \tag{49}$$

Such states are also called n-separable states [31]. Equivalently, a pure state $|\Psi\rangle$ is an n-separable state if it is separable across all the possible bipartitions of the n qubits. If this condition is violated then the state is no longer n-separable. Some states can be entangled in certain bipartitions and separable in others. Some states are entangled across all bipartitions. This motivates a classification of n-partite quantum states on the bases of entanglement. For two-qubit systems, there are only two classes – separable and entangled states.

For three-qubit systems, we have more. Let A, B, and C represent the three-qubits, and let us use A-B to denote that subsystems A and B are separable and AB to denote that subsystems A and B are entangled. Then the entanglement classes can be written as {A-B-C, AB-C, A-BC, AC-B, ABC}. Note that we do not distinguish between W states and GHZ states defined in [22] and keep them in the same class ABC.

The same classification of the entanglement can be extended to n number of qubits. The question that is relevant to us is the following: given an n-qubit arbitrary state, is there a way to label it according to its entanglement class. We show that classical kNN algorithm can classify these states for $n=2,3$ with high accuracy. Furthermore, the same accuracy can be achieved by our QkNN algorithm without the classical description of the given quantum state, establishing the advantage of QkNN over classical kNN algorithm, with the only requirement being that the circuits for state preparation must be provided.

Note that these train circuits can be easily prepared in this setting. For example, train states which have two qubits maximally entangled can be easily prepared by applying a Hadamard gate and a CNOT gate on the two qubits, and then applying random one qubit gates on each qubit, such as rotation over randomly chosen angles. Computing the explicit description of such states would be difficult, but QkNN requires only these circuits and not the description of the states.

2. *Simulation results*

We present the results of simulation of estimating the entanglement class of a test state $|\psi\rangle$ using classical kNN in Table I. We use this numerical experiment to demonstrate that quantum states in Hilbert space has a nearest-neighbor structure when it comes to entanglement. That is, the closer the states are, the similar their entanglement is.

For two-qubit states, we demonstrate this for both separable (entanglement entropy $=0$) vs entangled (entanglement entropy $\neq 0$) and separable vs maximally entangled (entanglement entropy $=1$). We also demonstrate it for 3 qubit states with a classification among the 5 different classes mentioned above (VI A 1).

No. of Qubits	No. of classes	Entanglement classes	Accuracy
2	2	Separable, Entangled	99%
2	2	Separable, Maximally entangled	100%
3	5	1-2-3, 12-3, 1-23, 13-2, 123	89%

TABLE I. Entanglement classification using classical kNN classifier. Cardinality of the set of train states is $M =$ (number of classes)$\times$(class size). In each case, the total number of train states used for each class is 10^5

B. Quantum state discrimination

Another application that we propose for QkNN is a problem analogous to quantum state discrimination [32]. The problem of quantum state discrimination is originally formulated for arbitrary (mixed) states and in terms of measurement. Consider a collection of states $\mathcal{S} = \{\rho_0, \ldots, \rho_{M-1}\}$ with associated probabilities $p = (p_0, \ldots, p_{M-1})$. A state is drawn from S according to p and is prepared. The aim is to find a measurement which maximises the probability of correctly identifying the prepared state.

We are concerned with an analogous problem. Suppose we are given a circuit to produce an unknown (pure) state $|\psi\rangle$. We are guaranteed that $|\psi\rangle$ is one of the M known (pure) states $\{|\phi_0\rangle, \ldots, |\phi_{M-1}\rangle\}$. We also assume that we are given circuits to prepare the known states. The task is to correctly deduce the identity of $|\psi\rangle$.

In such a scenario, one can run the QkNN algorithm with $k = 1$ to obtain the j such that $|\psi\rangle = |\phi_j\rangle$ with $O(\sqrt{M})$ oracle calls, as the largest value fidelity can take is 1 and $\mathrm{F}(u, v) = 1$ if and only if $u = v$.

VII. CONCLUSION

In this paper, we have presented a novel QkNN algorithm, which is a quantum analog of classical kNN algorithm. We use Swap test and generalizations of quantum analog to digital conversion algorithms to construct an oracle which enables us to reduce the problem of quantum kNN to an instance of quantum k-maxima finding algorithm. We assume that state preparations circuits are provided and the algorithm uses fidelity as a similarity measure which is widely used in problems where the data is inherently quantum. The algorithm requires $O(\sqrt{kM})$ calls to these circuits to obtain the identities of the k nearest neighbors of the test out of M train states. Since the metric Bures distance is a monotonous function of fidelity, kNN done using fidelity is in agreement with kNN carried out using Bures distance. We also present a variant of the algorithm which uses dot product as distance measure.

An advantage of the proposed algorithm is its ability to classify quantum states without their explicit classical description in some basis. Instead, we require circuits capable of preparing these states. Furthermore, while dealing with quantum data, the algorithm is able to classify without the requirement of quantum state tomography which is essential if one were to use any other, quantum or classical, kNN algorithm which requires classical description of states. Along with the calls to the staet preparation circuits being $O(\sqrt{kM})$, all the other parts of the circuit are efficiently preparable. The number of qubits required are also poly-logarithmic in the dimension of the states involved and the number of train states.

We then show the effectiveness of kNN method in identifying the type of entanglement in quantum states. For the problem of entanglement classification, preparing train states of different types of entanglement is much easier when we're working with their circuits rather than working with their classical descriptions. Furthermore, we discuss the applicability of the algorithm in an analogous version of quantum state discrimination.

A particular future direction regarding the applications of QkNN is to study its capability in an analogous version of quantum gate discrimination [33]. This is due to the ability of QkNN to work with quantum circuits which could be efficient representations of unitary matrices. The problem of identifying a test circuit among a finite set of train circuits can potentially be addressed using a column-wise QkNN.

VIII. ACKNOWLEDGEMENTS

We would like to thank Yuan Feng, Sanjiang Li, and Christopher Ferrie for fruitful discussions. The quantum circuits were generated using the Quantikz package [34] and Matcha [35].

[1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).
[2] P. Wittek, *Quantum machine learning: what quantum computing means to data mining* (Academic Press, 2014).
[3] M. Schuld, I. Sinayskiy, and F. Petruccione, An introduction to quantum machine learning, Contemporary Physics **56**, 172–185 (2014).
[4] S. Arunachalam and R. de Wolf, A survey of quantum learning theory (2017), arXiv:1701.06806 [quant-ph].
[5] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Physical review letters **103**, 150502 (2009).

[6] N. Wiebe, D. Braun, and S. Lloyd, Quantum algorithm for data fitting, Physical review letters **109**, 050505 (2012).

[7] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, Nature Physics **13**, 431 (2017).

[8] B. Wang, Learning to detect entanglement (2017), arXiv:1709.03617 [quant-ph].

[9] S. Lu, S. Huang, K. Li, J. Li, J. Chen, D. lu, Z. Ji, Y. Shen, D. Zhou, and B. Zeng, Separability-entanglement classifier via machine learning, Physical Review A **98** (2017).

[10] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Reviews of Modern Physics **91**, 045002 (2019).

[11] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nature Physics **10**, 631 (2014).

[12] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, Physical Review Letters **113**, 10.1103/physrevlett.113.130503 (2014).

[13] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning, arXiv preprint arXiv:1307.0411 (2013).

[14] T. Cover and P. Hart, Nearest neighbor pattern classification, IEEE transactions on information theory **13**, 21 (1967).

[15] C. Dürr and P. Hoyer, A quantum algorithm for finding the minimum, CoRR **quant-ph/9607014** (1996).

[16] N. Wiebe, A. Kapoor, and K. M. Svore, Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning, Quantum Info. Comput. **15**, 316–356 (2015).

[17] 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, International Journal of Theoretical Physics **56**, 3496 (2017).

[18] H. Chen, Y. Gao, and J. Zhang, Quantum k-nearest neighbor algorithm, Dongnan Daxue Xuebao **45**, 647 (2015).

[19] Y. Dang, N. Jiang, H. Hu, Z. Ji, and W. Zhang, Image classification based on quantum k-nearest-neighbor algorithm, Quantum Information Processing **17**, 1 (2018).

[20] M. Schuld, I. Sinayskiy, and F. Petruccione, Quantum computing for pattern classification, in *PRICAI 2014: Trends in Artificial Intelligence*, edited by D.-N. Pham and S.-B. Park (Springer International Publishing, Cham, 2014) pp. 208–220.

[21] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, Quantum fingerprinting, Phys. Rev. Lett. **87**, 167902 (2001).

[22] W. Dür, G. Vidal, and J. I. Cirac, Three qubits can be entangled in two inequivalent ways, Physical Review A **62**, 10.1103/physreva.62.062314 (2000).

[23] K. Mitarai, M. Kitagawa, and K. Fujii, Quantum analog-digital conversion, Phys. Rev. A **99**, 012301 (2019).

[24] S. Aaronson, The learnability of quantum states, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **463**, 3089–3114 (2007).

[25] Y. Liao and V. R. Vemuri, Use of k-nearest neighbor classifier for intrusion detection, Computers & security **21**, 439 (2002).

[26] I. Mani and I. Zhang, knn approach to unbalanced data distributions: a case study involving information extraction, in *Proceedings of workshop on learning from imbalanced datasets*, Vol. 126 (2003).

[27] R. J. Samworth *et al.*, Optimal weighted nearest neighbor classifiers, The Annals of Statistics **40**, 2733 (2012).

[28] C. Dürr, M. Heiligman, P. Hoyer, and M. Mhalla, Quantum query complexity of some graph problems, SIAM Journal on Computing **35**, 1310–1328 (2006).

[29] K. Miyamoto, M. Iwamura, and K. Kise, A quantum algorithm for finding k-minima (2019), arXiv:1907.03315 [quant-ph].

[30] M. A. Nielsen and I. Chuang, Quantum computation and quantum information (2002).

[31] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, Quantum entanglement, Reviews of Modern Physics **81**, 865–942 (2009).

[32] J. Bae and L.-C. Kwek, Quantum state discrimination and its applications, Journal of Physics A: Mathematical and Theoretical **48**, 083001 (2015).

[33] G. Chiribella, G. M. D'Ariano, and M. Roetteler, Identification of a reversible quantum gate: assessing the resources, New Journal of Physics **15**, 103019 (2013).

[34] A. Kay, Tutorial on the quantikz package, arXiv preprint arXiv:1809.03842 (2018).

[35] Mathcha, Mathcha - online mathematics editor, year = 2021, url = http://www.mathcha.io,.

[36] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, Quantum algorithms revisited, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **454**, 339–354 (1998).

[37] L. Ruiz Pérez and J. C. Garcia-Escartin, Quantum arithmetic with the quantum fourier transform, Quantum Information Processing **16** (2017).

Appendix A: Quantum Analog-Digital Conversion algorithms

Mitarai et al. describes a set of algorithms [23] to carry out analog-digital conversions within a quantum circuit.

Definition 1. *abs-QADC [23] Let $\widetilde{r_j}$ denote the m-bit string $\widetilde{r_j}^1, \widetilde{r_j}^2, \ldots, \widetilde{r_j}^m$ that best approximates $|c_j|$ by $\sum_{k=1}^{m} \widetilde{r_j}^k 2^{-k}$. An m-bit abs-QADC operation transforms analog-encoded state $\sum_{j=1}^{N} c_j |j\rangle |0\rangle^{\otimes m}$ to $\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle |\widetilde{r_j}\rangle$*

Definition 2. *real-QADC [23] Let $\widetilde{x_j}$ denote the m-bit string $\widetilde{x_j}^1, \widetilde{x_j}^2, \ldots, \widetilde{x_j}^m$ that best approximates the real part of c_j by $\sum_{k=1}^{m} \widetilde{x_j}^k 2^{-k}$. An m-bit real-QADC operation transforms analog-encoded state $\sum_{j=1}^{N} c_j |j\rangle |0\rangle^{\otimes m}$ to $\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle |\widetilde{x_j}\rangle$*

Theorem 1. *abs-QADC [23] There exists an m-bit abs-QADC algorithm that runs using $O(1/\epsilon)$ controlled-U_A gates and $O((log^2 N)/\epsilon)$ single and two qubit gates with output state fidelity $(1 - O(poly(\epsilon)))$, where $\epsilon = 2^{-m}$*

Theorem 2. *real-QADC [23] There exists an m-bit real-QADC algorithm that runs using $O(1/\epsilon)$ controlled-U gates and $O((log^2 N)/\epsilon)$ single and two qubit gates with output state fidelity $(1 - O(poly(\epsilon)))$, where $\epsilon = 2^{-m}$*

Appendix B: Quantum Phase Estimation

Quantum phase estimation is a quantum procedure that can be used to estimate the phase of the eigenvalue of a given eigenvector of a unitary operator. It relies on the quantum fourier transform and is the engine behind some of the most popular quantum algorithms such as the Shor's algorithm for factoring,

Theorem 3. *[36] Let U be a unitary operator acting on M-qubit Hilbert space with eigenstates $\{|\psi_j\rangle\}_{j=1}^{2^M}$ and corresponding eigenvalues $\{e^{2\pi i\phi_j}\}_{j=1}^{2^M}$ where $\phi_j \in [0, 1)$. Let $\epsilon = 2^{-m}$ for some positive integer m. There exists a quantum algorithm, which consists of $O(1/\epsilon)$ controlled-U calls and $O(log^2(1/\epsilon))$ single and two-qubit gates, that performs transformation $\sum_{j=1}^{2^M} a_j|\psi_j\rangle|0\rangle^{\otimes m} \rightarrow |\psi_{PE}\rangle = \sum_{j=1}^{2^M} a_j|\psi_j\rangle|\tilde{\phi}_J\rangle$ where $|\tilde{\phi}_J\rangle$ denotes a bitstring $\tilde{\phi_J}^{(1)}\tilde{\phi_J}^{(2)}\ldots\tilde{\phi_J}^{(m)}$ such that $\left|\sum_{k=1}^{m}\tilde{\phi_J}^{(k)}2^{-k} - \phi_J\right| \leq \epsilon$ for all j with state fidelity at least $1 - poly(\epsilon)$.*

Appendix C: Quantum arithmetics

Within a quantunm circuit, one can always apply simple arithmetic functions such as additions, multiplication, trginometric functions, exponentiation, etc. This is explained using the following theorems.

Theorem 4. *[37] Let $\mathbf{a}, \mathbf{b}$ be m-bit strings. There exists a quantum algorithm that performs transformation $|\mathbf{a}\rangle|\mathbf{b}\rangle \rightarrow |\mathbf{a}\rangle|\mathbf{a}+\mathbf{b}\rangle$ with O(poly(m)) single and two qubit gates.*

Using this quantum adder, one can construct a circuit capable of carrying out any basic function in a similar manner in a quantum circuit.

Theorem 5. *[23] Some basic functions such as inverse, trigonometric functions, square root, and inverse trigonometric functions can be calculated to accuracy ϵ, that is, we can perform a transformation $|a\rangle|0\rangle \rightarrow |a\rangle|\mathbf{f}(\tilde{\mathbf{a}})\rangle$ such that $|\mathbf{f}(\mathbf{a}) - \mathbf{f}(\tilde{\mathbf{a}})| \leq \epsilon$ where f is the required function using $O(poly(\log(1/\epsilon)))$ quantum arithmetics.*

Appendix D: Hadamard Test

The Hadamard test is a quantum circuit which can be used to compute the real part or the imaginary part of the inner product between two quantum states $|u\rangle, |v\rangle \in \mathbb{C}^n$. Since, we are more interested in the real part of the inner product, we will be using and explaining that particular version of the Hadamard test.

Let $U|0\rangle = |u\rangle$ and $V|0\rangle = |v\rangle$. The aim is to use U amd V in a controlled manner to construct the state

$$\frac{1}{2}\Big[|0\rangle\Big(|u\rangle + |v\rangle\Big) + |1\rangle\Big(|u\rangle - |v\rangle\Big)\Big] \tag{D1}$$

The detailed circuit is shown in Figure 9. The real part of the inner product can be estimated from measuring the first qubit as

$$\Pr(0) = \frac{1}{2} + \frac{1}{2}\mathrm{Re}(\langle u|v\rangle), \tag{D2}$$

$$\Pr(1) = \frac{1}{2} - \frac{1}{2}\mathrm{Re}(\langle u|v\rangle). \tag{D3}$$

The quantity $\Pr(0) - \Pr(1)$ gives us the desired real part of the inner product.

Appendix E: Quantum k Nearest neighbors algorithm using dot product

A more general distance measure used in kNN problems is the dot product. Let $\mathcal{H}$ be the n-qubit Hilbert space and let $|v\rangle \in \mathcal{H}$ be the real valued test state whose label is to be determined. Let

$$\{|u_i\rangle : i \in \{0, \ldots, M-1\}\} \subset \mathcal{H} \tag{E1}$$

be a collection of M known real valued train states whose labels are known to us

Let $X_i \equiv \mathrm{X}(v, u_i) = \langle v|u_i\rangle$ be the dot product between the test state $|v\rangle$ and the i^{th} train state $|u_i\rangle$ and

$$X = [X_0, \ldots, X_{M-1}] \tag{E2}$$

be a table of length M containing the dot product values with the test state $|v\rangle$ and all the train states $\{|u_i\rangle\}$. The general approach adopted here is very similar to fidelity based QkNN. One can see that the problem of QkNN using dot product is an instance of the quantum k-maxima finding algorithm on the set X given in (E2). Similar to QkNN using fidelity, the problem boils down to being able to perform Grover search on the Boolean function

$$f_{y,A}(j) = \begin{cases} 1 & : X_j > X_y \text{ and } j \notin A \\ 0 & : \text{otherwise.} \end{cases} \tag{E3}$$

That is, we should be able to prepare the oracle

$$\mathcal{O}_{y,A}|j\rangle|0\rangle = \begin{cases} |j\rangle|1\rangle & : X_j > X_y \text{ and } j \notin A \\ |j\rangle|0\rangle & : \text{otherwise.} \end{cases} \tag{E4}$$

The assumptions here are that we are provided with *state preparation* oracles $\mathcal{V}, \mathcal{W}$ of the form

$$|0^n\rangle \xrightarrow{\mathcal{V}} |v\rangle, \tag{E5}$$

$$|j\rangle|0^n\rangle \xrightarrow{\mathcal{W}} |j\rangle|u_j\rangle, \tag{E6}$$

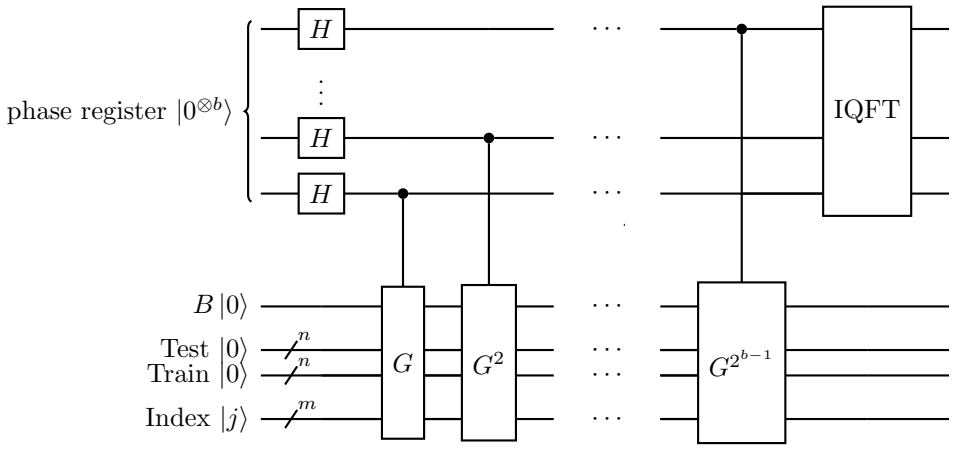

FIG. 8. Quantum phase estimation on an operator G. This is the innards of the 'PhaseEst on G' operator in Figure 7.

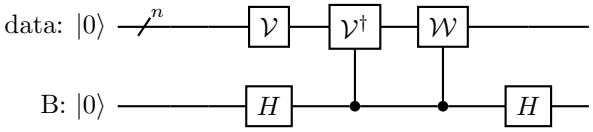

FIG. 9. Circuit for Hadamard test.

for $j \in \{0, \ldots, M-1\}$. The construction of the oracle is based on the real-QADC circuit from [23]. In that circuit, to compute the real values of the coordinates of the state, we apply the Hadamard test for real part of dot product in superposition with the state and standard basis vectors. In the quantum kNN setting, we apply Hadamard test in superposition with test state and the train states. We now go on to show the correctness of such a protocol and a method to build the required oracle $\mathcal{O}_y$ using it. The explicit construction of the oracle is as follows:

1. Initialise three registers named *index*, *data*, B of sizes $m, n, 1$ respectively, where $n = \log N$ and $m = \log M$.

$$|j\rangle_{\text{in}}|0^{\otimes n}\rangle_{\text{data}}|0\rangle_B \tag{E7}$$

2. Apply $\mathcal{V}$ on the data register

$$\xrightarrow{\mathcal{V}} |j\rangle_{\text{in}}|v\rangle_{\text{data}}|0\rangle_B \tag{E8}$$

3. Perform Hadamrd test to obtain the state

$$\begin{aligned}\xrightarrow{\text{Hadamard Test}} \frac{1}{2}|j\rangle_{\text{in}}\Big[&\Big(|v\rangle_{\text{data}} + |u_j\rangle_{\text{data}}\Big)|0\rangle_B + \\ &\Big(|v\rangle_{\text{data}} - |u_j\rangle_{\text{data}}\Big)|1\rangle_B\Big] \\ &= |j\rangle_{\text{in}}|\Psi_j\rangle_{\text{data},B}\end{aligned} \tag{E9}$$

where,

$$\begin{aligned}|\Psi_j\rangle_{\text{data},B} = &\Big(|v\rangle_{\text{data}} + |u_j\rangle_{\text{data}}\Big)|0\rangle_B + \\ &\Big(|v\rangle_{\text{data}} - |u_j\rangle_{\text{data}}\Big)|1\rangle_B\end{aligned} \tag{E10}$$

Define V to be the combined unitary transformations of steps (2) and (3). If one now measures the B register, one would see the probabilities as

$$\Pr(B=0) = \frac{1+X_j}{2}, \tag{E11}$$

$$\Pr(B=1) = \frac{1-X_j}{2}. \tag{E12}$$

The information regarding dot product is now encoded in the amplitudes. We must now convert it into a 'digital' format which can be further utilised.

4. Construct a gate

$$H = V_{\text{in,data},B}\; \mathcal{S}_{0_{\text{data},B}}\; V^\dagger_{\text{in,data},B}\; Z_B, \tag{E13}$$

where Z_B denotes the application of the Z gate on register B and $S_0 = \mathbb{1} - 2|0\rangle\langle 0|$. This H gate can be seen as the H gate in the real QADC algorithm [23], with the controlled CNOT gate replaced by controlled $\mathcal{W}$. The action of H on the current state can be written as controlled action of operators H_j (refer here (H)):

$$H|j\rangle_{\text{in}}|\Psi_j\rangle_{\text{data},B} = |j\rangle_{\text{in}}\Big(H_j|\Psi_j\rangle_{\text{data},B}\Big), \tag{E14}$$

where

$$H_j = (\mathbb{1} - 2|\Psi_j\rangle\langle\Psi_j|_{\text{data},B})Z_B \tag{E15}$$

5. $|\Psi_j\rangle_{\text{tr,tst},B}$ can be decomposed into two eigenstates of H_j, namely $|\Psi_{j+}\rangle$ and $|\Psi_{j-}\rangle$, corresponding to the eigenvalues $e^{\pm i2\pi\theta_j}$, respectively. Here,

$\sin(\pi\theta_j) = \sqrt{\frac{1}{2}(1+X_j)}$ and $\theta \in [1/4, 1/2)$ (refer Appendix I). The decomposition is given as

$$|\Psi_j\rangle = \frac{-i}{\sqrt{2}}(e^{i\pi\theta_j}|\Psi_{j+}\rangle - e^{-i\pi\theta_j}|\Psi_{j-}\rangle), \tag{E16}$$

6. The operator H has the dot product values $\{X_j\}$ stored in its eigenvalues. To get a b-bit binary representation of θ_j, we now run the phase estimation algorithm on H. To this end, we bring the *phase register* containing b qubits and run the phase estimation algorithm:

$$\xrightarrow{\text{PhaseEst.}} \frac{-i}{\sqrt{2}}|j\rangle_{\text{in}}\Big[e^{i\pi\theta_j}|\theta_j\rangle_{\text{ph}}|\Psi_{j+}\rangle_{\text{data}B} - e^{-i\pi\theta_j}|1-\theta_j\rangle_{\text{ph}}|\Psi_{j-}\rangle_{\text{data},B}\Big] \equiv |j\rangle_{\text{in}}|\Psi_{j,\text{AE}}\rangle_{\text{ph,data},B}. \tag{E17}$$

where we have defined the combined state of all registers except index register after estimation to be

$$|\Psi_{j,\text{AE}}\rangle_{\text{ph,data},B} = \frac{-i}{\sqrt{2}}\Big(e^{i\pi\theta_j}|\theta_j\rangle_{\text{ph}}|\Psi_{j+}\rangle_{\text{data},B} - e^{-i\pi\theta_j}|1-\theta_j\rangle_{\text{ph}}|\Psi_{j-}\rangle_{\text{data},B}\Big). \tag{E18}$$

Here, $|\theta_j\rangle_{\text{ph}}$ and $|1-\theta_j\rangle_{\text{ph}}$ are b-qubit states storing b-bit binary representation of θ_j and $1-\theta_j$ respectively.

7. Introducing a register, *dp*, compute $X_j = 2\sin^2(\pi\theta_j) - 1$ using quantum arithmetic from theorem (C). Note that $\sin(\pi\theta_j) = \sin(\pi(1-\theta_j))$, and X_j is uniquely recovered. Then our total state is

$$\xrightarrow{\text{quantum arithmetics}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|\Psi_{j,\text{AE}}\rangle_{\text{ph,data},B} \tag{E19}$$

8. Uncompute everything in registers ph,data and B to get

$$\xrightarrow{\text{uncompute ph, data, B}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}} \tag{E20}$$

Now, we have the successfully converted the dot product values from amplitudes to digital format.

9. Add more registers and apply $\mathcal{X}$, so that we have

$$\xrightarrow{\mathcal{X}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|y\rangle_{\text{in'}}|X_y\rangle_{\text{dp'}} \tag{E21}$$

10. Add an extra qubit Q_1 and apply the gate J defined in (11) on registers dp and dp$'$ to get the state

$$\xrightarrow{J} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|y\rangle_{\text{in'}}|X_y\rangle_{\text{dp'}}|g(j)\rangle_{Q_1} \tag{E22}$$

where

$$g(j) = \begin{cases} 1 & : X_j > X_y \\ 0 & : X_j \le X_y, \end{cases} \tag{E23}$$

11. Uncompute the registers in$'$ and dp$'$ to obtain the state

$$\xrightarrow{\text{uncompute in', dp'}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|g(j)\rangle_{Q_1} \tag{E24}$$

12. Add an extra qubit Q_2 and for every $i_l \in A$, apply the gate $D^{(i_l)}$ defined in (13) on registers index and Q_2 to get the state

$$\xrightarrow{\left(D^{(i_1)}\dots D^{(i_k)}\right)_{\text{in},Q_1}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|g(j)\rangle_{Q_1}|\chi_A(j)\rangle_{Q_2} \tag{E25}$$

where $\chi_A(j) = 1$ if $j \in A$ and 0 otherwise, is the indicator function of the set A.

13. Add an extra qubit Q_3. Then apply an X gate on Q_2 and a Toffoli gate with controls Q_1, Q_2 and target Q_3. This results in the state

$$\xrightarrow{X,\text{Toffoli}} |j\rangle_{\text{in}}|X_j\rangle_{\text{dp}}|g(j)\rangle_{Q_1}|\chi_A(j)\rangle_{Q_2}|f_{y,A}(j)\rangle_{Q_3} \tag{E26}$$

where $f_{y,A}$ is the Boolean function defined in (E).

14. Uncomputing every register except index and Q_3, we have

$$\xrightarrow{\text{uncompute}} |j\rangle_{\text{in}}|f_{y,A}(j)\rangle_{Q_3} \tag{E27}$$

$$\mathcal{O}_{y,A}|j\rangle|0\rangle = |j\rangle|f_{y,A}(j)\rangle. \tag{E28}$$

This completes the construction of the oracle (E). We may now use this oracle in the k-maxima finding algorithm to find the k nearest neighbors of a test state based on dot product.

Appendix F: Action of G as controlled G_js for the Fidelity based QkNN

Recall that

$$G = U_{\text{tr,tst},B}\mathcal{W}_{\text{in,tr}}S_{0_{\text{tr,tst,B}}}\mathcal{W}^\dagger_{\text{in,tr}}U^\dagger_{\text{tr,tst},B}Z_B, \quad G|k\rangle_{\text{in}}|\Psi_j\rangle_{\text{tr,tst},B} = |k\rangle_{\text{in}}\Big(G_k|\Psi_j\rangle_{\text{tr,tst},B}\Big). \tag{F1}$$

where

$$S_0 = \mathbb{1} - 2|0\rangle\langle 0|_{\text{tst, tr},B}, \quad S_k = \mathbb{1} - 2|0\rangle\langle 0|_{\text{tst, B}} \otimes |\phi_k\rangle\langle\phi_k| \tag{F2}$$

To show the equivalence between G and controlled G_k, it suffices to show the equivalence between $\mathcal{W}S_0\mathcal{W}^\dagger$ and S_k. Recall that we introduced $\mathcal{W}_{\text{ind, tr}}$ for the preparation of train states. The action of $\mathcal{W}$ is given by

$$\mathcal{W}_{\text{ind, tr}}|i\rangle_{\text{ind}}|0\rangle_{\text{tr}} = |i\rangle_{\text{ind}}|\phi_i\rangle_{\text{tr}}. \tag{F3}$$

We have

$$\mathbb{1}_{\text{in}} \otimes S_0 = \mathbb{1} - 2\sum_{k=0}^{M-1} |k,0,0,0\rangle\langle k,0,0,0|_{\text{in,tst,tr},B}, \quad \text{(F4)}$$

and therefore

$$\begin{aligned}
&\mathcal{W}S_0\mathcal{W}^\dagger \\
&= \mathcal{W}_{\text{ind,tr}} \left(\mathbb{1} - 2\sum_{k=0}^{M-1} |k,0,0,0\rangle\langle k,0,0,0|_{\text{tst, tr},B} \right) \mathcal{W}^\dagger_{\text{ind,tr}} \\
&= \mathbb{1} - 2\sum_{k=0}^{M-1} \mathcal{W}|k,0,0,0\rangle\langle k,0,0,0|\mathcal{W}^\dagger \\
&= \mathbb{1} - 2\sum_{k} |k,\phi_k,0,0\rangle\langle k,\phi_k,0,0| \\
&= \sum_k |k\rangle\langle k|_{\text{in}} \otimes \mathbb{1}_{\text{tst,tr},B} - 2\sum_k |k\rangle\langle k|_{\text{in}} \otimes |\phi_k\rangle\langle\phi_k| \otimes |0\rangle\langle 0|_{\text{tst},B}
\end{aligned} \quad \text{(F5)}$$

This is

$$\begin{aligned}
&\mathcal{W}S_0\mathcal{W}^\dagger \\
&= \sum_k |k\rangle\langle k| \otimes \left(\mathbb{1} - 2|0\rangle\langle 0|_{\text{tst, } B} \otimes |\phi_k\rangle\langle\phi_k|_{\text{tr}} \right) \\
&= \sum_k |k\rangle\langle k|_{\text{in}} \otimes (S_k)_{\text{tst, tr},B}
\end{aligned} \quad \text{(F6)}$$

as required.

Appendix G: Eigenvectors of G_j for the fidelity based QkNN

Recall that $G_j = US_jU^\dagger Z_B$ and $U|0\rangle_{\text{tst}}|\phi_{j_{\text{tr}}}\rangle|0\rangle_{\text{B}} = |\Psi_j\rangle$ This implies,

$$US_jU^\dagger = \mathbb{1} - 2|\Psi_j\rangle\langle\Psi_j| \quad \text{(G1)}$$

Let

$$|\Psi_{j0}\rangle = \frac{1}{2\alpha_j}\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} + |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|0\rangle_B, \quad \text{(G2)}$$

and

$$|\Psi_{j1}\rangle = \frac{1}{2\beta_j}\Big(|\psi\rangle_{\text{tr}}|\phi_j\rangle_{\text{tst}} - |\phi_j\rangle_{\text{tr}}|\psi\rangle_{\text{tst}}\Big)|1\rangle_B. \quad \text{(G3)}$$

where,

$$\alpha_j = \sqrt{\frac{1}{2}(1+F_j)}, \quad \text{(G4)}$$

and

$$\beta_j = \sqrt{\frac{1}{2}(1-F_j)}. \quad \text{(G5)}$$

So,

$$|\Psi_j\rangle = \alpha_j|\Psi_{j0}\rangle + \beta_j|\Psi_{j1}\rangle \quad \text{(G6)}$$

Consider the subspace $\mathcal{H} = \text{span}\Big(|\Psi_{j0}\rangle, |\Psi_{j1}\rangle\Big)$. Then

$$Z_B\Big|_{\mathcal{H}} = |\Psi_{j0}\rangle\langle\Psi_{j0}| - |\Psi_{j1}\rangle\langle\Psi_{j1}| \quad \text{(G7)}$$

$$\begin{aligned}
US_jU^\dagger\Big|_{\mathcal{H}} &= (1-2\alpha_j^2)|\Psi_{j0}\rangle\langle\Psi_{j0}| \\
&+ (1-2\beta_j^2)|\Psi_{j1}\rangle\langle\Psi_{j1}| \\
&- 2\alpha_k\beta_k(|\Psi_{j1}\rangle\langle\Psi_{j0}| + |\Psi_{j0}\rangle\langle\Psi_{j1}|)
\end{aligned} \quad \text{(G8)}$$

This implies that

$$\begin{aligned}
G_j\Big|_{\mathcal{H}} &= US_jU^\dagger Z_B\Big|_{\mathcal{H}} \\
&= (1-2\alpha_j^2)|\Psi_{j0}\rangle\langle\Psi_{j0}| \\
&- (1-2\beta_j^2)|\Psi_{j1}\rangle\langle\Psi_{j1}| \\
&- 2\alpha_k\beta_k(|\Psi_{j1}\rangle\langle\Psi_{j0}| - |\Psi_{j0}\rangle\langle\Psi_{j1}|)
\end{aligned} \quad \text{(G9)}$$

We can see that this $G_j\Big|_{\mathcal{H}}$ has the same structure as the analogous operator in [23]. By using the same proof of correctness of abs-QADC in [23] we can see that each $|\Psi_j\rangle_{\text{tr,tst},B}$ can be decomposed into two eigenstates of G_j. Let $\alpha_j = \sin(\pi\theta_j)$ for $\theta_j \in [\frac{1}{4}, \frac{1}{2})$. Substituting this in (G), we get

$$\begin{aligned}
G_j\Big|_{\mathcal{H}} &= \cos(2\pi\theta_j)|\Psi_{j0}\rangle\langle\Psi_{j0}| \\
&+ \cos(2\pi\theta_j)|\Psi_{j1}\rangle\langle\Psi_{j1}| \\
&- \sin(2\pi\theta_j)(|\Psi_{j1}\rangle\langle\Psi_{j0}| - |\Psi_{j0}\rangle\langle\Psi_{j1}|)
\end{aligned} \quad \text{(G10)}$$

Then, we can write $G_j\Big|_{\mathcal{H}}$ as

$$G_j\Big|_{\mathcal{H}} = \begin{bmatrix} \cos(2\pi\theta_j) & \sin(2\pi\theta_j) \\ -\sin(2\pi\theta_j) & \cos(2\pi\theta_j) \end{bmatrix} \quad \text{(G11)}$$

This implies that G_j has eigenvectors

$$|\Psi_{j\pm}\rangle = \frac{1}{\sqrt{2}}(|\Psi_{j0}\rangle \pm i|\Psi_{j1}\rangle). \quad \text{(G12)}$$

with eigenvalues $\Psi_\pm = e^{\pm i2\pi\theta_j}$ respectively. Now, we can decompose $|\Psi_j\rangle$ as

$$|\Psi_j\rangle = \frac{-i}{\sqrt{2}}(e^{i\pi\theta_j}|\Psi_{j+}\rangle - e^{-i\pi\theta_j}|\Psi_{j-}\rangle), \quad \text{(G13)}$$

as required.

Appendix H: Action of H as controlled H_js for the dot product based QkNN

Recall that

$$\begin{aligned} H &= V_{\text{in, data},B}\ \mathcal{S}_{0_{\text{data,B}}}\ \mathcal{V}^{\dagger}_{\text{in, data},B}\ Z_B, \\ H_j &= (\mathbb{1} - 2|\Psi_j\rangle_{\text{data,B}}\langle\Psi_j|_{\text{data,B}})Z_B \end{aligned} \tag{H1}$$

Then,

$$\begin{aligned} H =& V_{\text{in,data,B}} \\ & \Big(\mathbb{1}_{\text{in,data,B}} - \mathbb{1}_{\text{in}} \otimes 2|0\rangle_{\text{data,B}}\langle 0|_{\text{data,B}}\Big) \\ & V^{\dagger}_{\text{in,data,B}}\Big(\mathbb{1}_{\text{in,data}} \otimes Z_B\Big) \\ &= \Big(\mathbb{1}_{\text{in,data,B}} - \sum_{k=0}^{M-1} 2|k\rangle_{\text{in}}|\Psi\rangle_{\text{data,B}}\langle k|_{\text{in}}\langle\Psi|_{\text{data,B}}\Big) \\ & \Big(\mathbb{1}_{\text{in,data}} \otimes Z_B\Big) \\ &= \sum_{k=0}^{M-1} |k\rangle_{\text{in}}\langle k|_{\text{in}} \otimes \Big(\mathbb{1}_{\text{data,B}} - 2|\Psi_j\rangle_{data,B}\langle\Psi_j|_{data,B}\Big) \\ & \Big(\mathbb{1}_{\text{in,data}} \otimes Z_B\Big) \\ &= \sum_{k=0}^{M-1} |k\rangle_{\text{in}}\langle k|_{\text{in}} \otimes G_K \end{aligned} \tag{H2}$$

Appendix I: Eigenvectors of H_j for the dot product based QkNN

Recall that $H_j = \Big(\mathbb{1} - 2|\Psi_j\rangle\langle\Psi_j|\Big)Z_B$

Let

$$|\Psi_{j0}\rangle = \Big(|v\rangle_{\text{data}} + |u_j\rangle_{\text{data}}\Big)|0\rangle_B, \tag{I1}$$

and

$$|\Psi_{j1}\rangle = \Big(|v\rangle_{\text{data}} - |u_j\rangle_{\text{data}}\Big)|1\rangle_B. \tag{I2}$$

where,

$$\alpha_j = \sqrt{\frac{1}{2}(1 + X_j)}, \tag{I3}$$

and

$$\beta_j = \sqrt{\frac{1}{2}(1 - X_j)}. \tag{I4}$$

So,

$$|\Psi_j\rangle = \alpha_j|\Psi_{j0}\rangle + \beta_j|\Psi_{j1}\rangle \tag{I5}$$

Consider the subspace $\mathcal{H} = \text{span}\Big(|\Psi_{j0}\rangle, |\Psi_{j1}\rangle\Big)$. Then

$$Z_B\Big|_{\mathcal{H}} = |\Psi_{j0}\rangle\langle\Psi_{j0}| - |\Psi_{j1}\rangle\langle\Psi_{j1}| \tag{I6}$$

$$\begin{aligned} \Big(\mathbb{1} - 2|\Psi_j\rangle\langle\Psi_j|\Big)\Big|_{\mathcal{H}} =& (1 - 2\alpha_j^2)|\Psi_{j0}\rangle\langle\Psi_{j0}| \\ &+ (1 - 2\beta_j^2)|\Psi_{j1}\rangle\langle\Psi_{j1}| \\ &- 2\alpha_k\beta_k(|\Psi_{j1}\rangle\langle\Psi_{j0}| + |\Psi_{j0}\rangle\langle\Psi_{j1}|) \end{aligned} \tag{I7}$$

This implies that

$$\begin{aligned} H_j\Big|_{\mathcal{H}} =& (1 - 2\alpha_j^2)|\Psi_{j0}\rangle\langle\Psi_{j0}| \\ &- (1 - 2\beta_j^2)|\Psi_{j1}\rangle\langle\Psi_{j1}| \\ &- 2\alpha_k\beta_k(|\Psi_{j1}\rangle\langle\Psi_{j0}| - |\Psi_{j0}\rangle\langle\Psi_{j1}|) \end{aligned} \tag{I8}$$

The result immediately follows by comparing Equation (I) with (G).