Compact and classically preprocessed data-loading quantum circuit as a quantum random access memory

Bojia Duan [*]
School of Computer and Electronic Information/School of Artificial Intelligence, Nanjing Normal University, Nanjing 210023, China

Chang-Yu Hsieh [†]
Innovation Institute for Artificial Intelligence in Medicine of Zhejiang University, College of Pharmaceutical Sciences, Zhejiang University, Hangzhou 310058, China

(Received 6 August 2023; revised 27 May 2024; accepted 25 June 2024; published 22 July 2024)

Quantum random access memory (QRAM) is an essential ingredient for many quantum algorithms, such as quantum machine learning models. The quantum resources (qubit counts and circuit depths) required to load classical data into a circuit severely prohibit many reasonably sophisticated demonstrations in the near term. To this end, we propose a method to construct a compact QRAM circuit, leveraging classical preprocessing to design as shallow a circuit with as few auxiliary qubits as possible. The underlying idea is to map data-encoded quantum states into an orthogonal set of quantum states (characterized by a family of parametrized circuits); then one can straightforwardly entangle the data qubits with the address qubits using CNOT gates to realize a data-loading QRAM circuit. The nonunitary transformation to convert the orthogonalized quantum states back to the original data-encoded quantum states can be easily achieved with the help of a small set of auxiliary qubits and postselections. Numerical simulations on handwritten digits and Iris data sets are performed to assess the effectiveness of the proposed method, and we also highlight how noise influences quantum circuit performance across diverse configurations using random data sets. The proposed method for constructing a QRAM circuit can potentially facilitate the development and testing of many quantum machine learning methods in the near term.

DOI: 10.1103/PhysRevA.110.012616

I. INTRODUCTION

Quantum machine learning (QML), as a subfield of quantum computing, has garnered significant attention from researchers worldwide. Numerous proposals of QML algorithms effectively demonstrate the potential for achieving quantum advantages [1]. These promising breakthroughs are predominantly grounded in the transition from the classical computing basis to the quantum Hilbert space, enabling the exploitation of the intrinsic power inherent in quantum-mechanical principles, such as superposition and entanglement for information processing. This offers promising prospects to enhance machine learning, ranging from reduced computational complexity to improved generalization performance.

Efficiently loading a large amount of classical data onto a quantum computer poses a central challenge for ensuring the optimal performance of certain quantum algorithms, such as QML. In response, the concept of quantum random access memory (QRAM), which served as the counterpart to classical random access memory, was introduced to streamline the storage and retrieval of data in the form of quantum superposition [2]. QRAM allows for querying multiple memory addresses in superposition $\sum_i |i\rangle$, and returns the desired data $\sum_i |\mathbf{x}_i\rangle$ in an entangled and superpositioned form:

$$\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle|0\rangle^{\otimes n}\xrightarrow{\text{QRAM}}\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle|\mathbf{x}_i\rangle, \tag{1}$$

where $M=2^m$ and $N=2^n$ represent the number and dimensionality of the normalized data, respectively, and $|\mathbf{x}_i\rangle$ is an n-qubit quantum state stored in the ith memory cell. The potential of efficiently implementing QRAM could result in an exponential speedup for a variety of quantum algorithms [3–5]. Since then, many proposals have been put forward to implement the QRAM either via a quantum circuit model or physical implementations as well as its applications [6–19]. For a more comprehensive understanding of the various structures of QRAM, please refer to the detailed review provided in Appendix. A.

In this paper, we propose a framework to implement a compact QRAM circuit. The proposed state-preparation method is composed of three conceptual steps. Initially, a set of M classical data $\mathbf{x}_i$ undergoes preprocessing to be converted to a set of orthogonalized data $\tilde{\mathbf{x}}_i$. Subsequently, the proposed quantum circuit U_1 is employed to load the orthogonalized data $\tilde{\mathbf{x}}_i$ into a quantum superposition state $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\tilde{\mathbf{x}}_i\rangle$ with indices i. By doing so, the QRAM enables the representation of multiple orthogonalized classical data items simultaneously in the quantum domain. Finally, to recover the desired target state $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\mathbf{x}_i\rangle$, the quantum circuit U_2 (along with

[*]Contact author: duanbojia@njnu.edu.cn
[†]Contact author: kimhsieh@zju.edu.cn

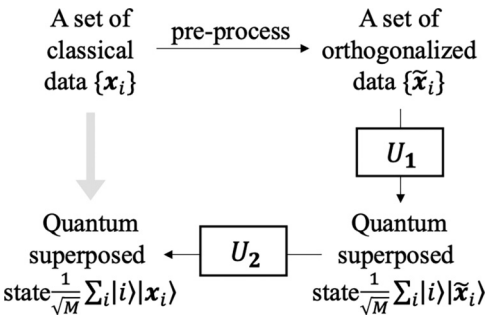

FIG. 1. The schematic diagram illustrates the entire process of the proposed QRAM. Initially, the M classical data $\{\mathbf{x}_i\}$ are preprocessed to become a set of orthogonalized data $\{\tilde{\mathbf{x}}_i\}$. Subsequently, the quantum circuit U_1 is employed to load the data set $\{\tilde{\mathbf{x}}_i\}$ into a quantum superposition state $\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle|\tilde{\mathbf{x}}_i\rangle$, and then the quantum circuit U_2 transforms the quantum state $\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle|\tilde{\mathbf{x}}_i\rangle$ to the target quantum state $\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle|\mathbf{x}_i\rangle$.

auxiliary qubits) is introduced to transform the orthogonalized data $|\tilde{\mathbf{x}}_i\rangle$ back to the original data $|\mathbf{x}_i\rangle$ while preserving the appropriate entanglement with the address qubits. Overall, a version of our proposed method requires just $O(poly(n, m))$ quantum resources.

The rest of the paper is organized as follows: Section II briefly introduces the proposed method for constructing a compact QRAM circuit with classical preprocessing, followed by the analysis of the QRAM circuit. Section III illustrates numerical simulations using both the handwritten digits data set and the Iris data set from sklearn. It also includes an experimental analysis of how noise affects quantum circuit performance across different configurations using random data sets. Section IV contains the relative conclusions and the possibility of future improvements. The typical research on QRAM is discussed in Appendix A.

II. METHOD

Our proposed method comprises three stages: data orthogonalization, a QRAM circuit for entangling the orthogonalized data with the address qubits, and a QRAM circuit for the nonunitary transformation for the reverse operation (nonorthogonalization) in the qubits' subspace. The schematic diagram for the entire workflow is summarized in Fig. 1. The preprocess depicted in the figure is presented in Sec. II A, and U_1 and U_2 are thoroughly discussed in Secs. II B and II C, respectively.

A. Data orthogonalization

Consider a set of normalized classical data $\{\mathbf{x}_i\} \in \mathbb{R}^N$, where $i = 0, 1, 2, \ldots, M-1$ represents the data index, and $N = 2^n$ is the dimension of $\mathbf{x}_i$. Assume that $\{\mathbf{x}_i\}$ are linearly independent, but any pair of data, such as $\mathbf{x}_i$ and $\mathbf{x}_j$, are probably not orthogonal to each other. For this data set, one may define an overlap matrix $S \in \mathbb{R}^{M\times M}$, where the element S_{ij} of the matrix S represents the inner product between the classical data vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, denoted as $S_{ij} := \langle \mathbf{x}_i|\mathbf{x}_j\rangle$ when using ket and bra notation. A new orthogonal data set $\{\tilde{\mathbf{x}}_i\}$ can be derived as follows:

$$\tilde{\mathbf{x}}_i = \sum_{j=0}^{M-1}(S^{-1/2})_{ij}\mathbf{x}_j, \tag{2}$$

where $S^{-1/2}$ is the inverse of $S^{1/2}$.

One can easily verify that these constructed vectors $\tilde{\mathbf{x}}_i$ are orthonormal:

$$\begin{aligned}\langle\tilde{\mathbf{x}}_i|\tilde{\mathbf{x}}_j\rangle &= \sum_k\sum_l \langle\mathbf{x}_k|(S^{-1/2})_{ik}(S^{-1/2})_{jl}|\mathbf{x}_l\rangle \\ &= \sum_k\sum_l (S^{-1/2})_{ik}\langle\mathbf{x}_k|\mathbf{x}_l\rangle(S^{-1/2})_{jl} \\ &= \sum_k\sum_l (S^{-1/2})_{ik}S_{kl}(S^{-1/2})_{lj} \\ &= (S^{-1/2}SS^{-1/2})_{ij} = \delta_{ij},\end{aligned} \tag{3}$$

where we use the fact that the $S^{-1/2}$ matrix is symmetric such that $S^{-1/2}_{ij} = S^{-1/2}_{ji}$.

It is important to note that in order to obtain the orthogonalized data, the matrix S should be well-conditioned. This condition is often satisfied when $m \leqslant n$ in the small data regime. But, under the scenario $m > n$, maintaining a well-conditioned S becomes more challenging. To address this issue, we proposed a straightforward "data grouping method," wherein we aggregate each set of 2^{m-n} data into small groups, treating each one as distinct new "grouped-data" points. In this sense, we have interchanged the roles of m and n. Consequently, the number of "grouped-address" qubits and "grouped-data" qubits are equal to n and m, respectively. A more detailed description of the data grouping method is provided in Appendix B. It is crucial to emphasize that this grouping method significantly improves the likelihood of obtaining a well-conditioned matrix S. Specifically, matrix S is deemed ill-conditioned when there exist at least two data points that exhibit high similarity. The transformation of the original data points into grouped data substantially reduces the similarity among them. In cases in which a single grouping approach fails to achieve the desired effect, we have the option to adjust the allocation of grouped data during preprocessing to further diminish the similarity between them. This straightforward method proves to be an effective strategy in facilitating the attainment of a well-conditioned matrix S in our approach.

It is worth noting that, for the sake of simplicity, we will focus on the scenario where $m \leqslant n$ in the subsequent discussions.

B. QRAM circuit for orthogonalized data

Let $\{|0\rangle, \ldots, |k\rangle, \ldots, |2^n-1\rangle\}$ denote a set of n-qubit computational bases (for instance, $|3\rangle = |0011\rangle$ when $n = 4$). Let $|\mathbf{x}_i\rangle := \sum_{k=0}^{N-1}(\mathbf{x}_i)_k|k\rangle$ represent an n-qubit amplitude encoding quantum state corresponding to the classical data $\mathbf{x}_i$. For clarity, all computational basis states denoted by $|\cdot\rangle$ utilize binary encoding, while all vectors represented by $|\cdot\rangle$ employ amplitude encoding. Define a $2^n \times 2^n$ unitary transformation $U_{\tilde{\mathbf{x}}}$ that converts the basis $|i\rangle$ to the quantum state $|\tilde{\mathbf{x}}_i\rangle$, i.e., $U_{\tilde{\mathbf{x}}}|i\rangle = |\tilde{\mathbf{x}}_i\rangle$. Since the states $|\tilde{\mathbf{x}}_i\rangle$ are mutually orthonormal,

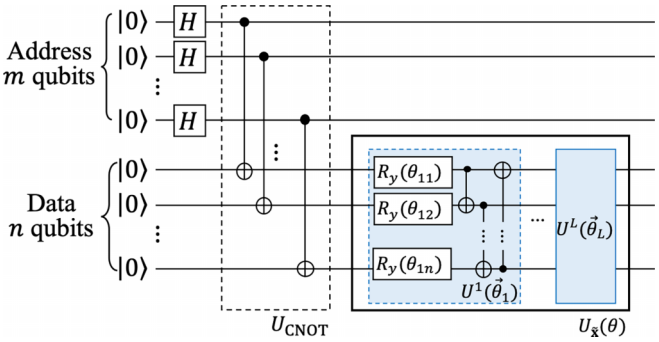

FIG. 2. An illustration of U_1 with the parametrized quantum circuit $U_1(\boldsymbol{\theta}) = [I^a \otimes U^d_{\tilde{\mathbf{x}}}(\boldsymbol{\theta})](U^{ad}_{\text{CNOT}})(H^a \otimes I^d)$. The left Hadamard and U_{CNOT} gates convert $|0\rangle^a|0\rangle^d$ to $\frac{1}{\sqrt{M}}\sum_i |i\rangle^a|i\rangle^d$. The right solid square represents a PQC with multiple layers of a hardware-efficient ansatz (HEA) [21], specifically designed for the conversion from $\frac{1}{\sqrt{M}}\sum_i |i\rangle^a|i\rangle^d$ to $\frac{1}{\sqrt{M}}\sum_i |i\rangle^a|\tilde{\mathbf{x}}_i\rangle^d$. Measurement is performed to estimate the probability distribution, which is then used to minimize the loss function, such as the mean-squared error (MSE) loss, in order to obtain the optimal parameters.

the following relationship holds:

$$U_{\tilde{\mathbf{x}}} = \sum_i |\tilde{\mathbf{x}}_i\rangle\langle i|. \tag{4}$$

As previously assumed, when $m \leqslant n$, one needs to construct an $N \times N$ unitary matrix in which the first $N \times M$ matrix should correspond to $U_{\tilde{\mathbf{x}}}$ and the remaining $N \times (N - M)$ matrix can be defined arbitrarily, as long as the overall transformation keeps its unitarity. One possible approach is to employ the Gram-Schmidt method to derive additional orthogonal states. Consequently, Eq. (4) can be unfolded in detail as follows:

$$U_{\tilde{\mathbf{x}}} = \sum_{i=0}^{M-1} |\tilde{\mathbf{x}}_i\rangle\langle i| + \sum_{i=N}^{M-N} |\tilde{\mathbf{u}}_i\rangle\langle i|. \tag{5}$$

The unitary operation $U_{\tilde{\mathbf{x}}}$, as presented in Eq. (4), is clearly defined. Therefore, it can be decomposed into elementary quantum gates using some quantum circuit compiling methods [20]. Another possible approach is to variationally learn a compact quantum circuit for $U_{\tilde{\mathbf{x}}}$ as a PQC. This approach leverages the quantum hardware to identify the desired circuit implementation of the desired transformation. More details are given in Sec. II B 1.

Once we have already obtained the quantum circuit for $U_{\tilde{\mathbf{x}}}$, then we may define the three-step unitary U_1, which transforms the initial state $|0\rangle$ to $|\Psi_1\rangle := \frac{1}{\sqrt{M}}\sum_i |i\rangle|\tilde{\mathbf{x}}_i\rangle$. First, the Hadamard gates are applied to the m address qubits. Subsequently, the U_{CNOT} gate is executed on both the m address qubits and the m data qubits. Finally, the $U_{\tilde{\mathbf{x}}}$ gate is solely applied to the n data qubits. A quantum circuit diagram that demonstrates the step-by-step execution of the circuit is shown in Fig. 2. The definition of U_1 is given by Eq. (6), where the superscripts a and d represent the address qubits and the data qubits, respectively:

$$U_1 = \left(I^a \otimes U^d_{\tilde{\mathbf{x}}}\right)\left(U^{ad}_{\text{CNOT}}\right)(H^a \otimes I^d). \tag{6}$$

It is easy to verify that after executing U_1, one then achieves the QRAM operation for loading the orthogonalized data:

$$\begin{aligned} U_1|0\rangle^{ad} &= \left(I^a \otimes U^d_{\tilde{\mathbf{x}}}\right)\left(U^{ad}_{\text{CNOT}}\right)(H^a \otimes I^d)|0\rangle^{ad} \\ &= \left(I^a \otimes U^d_{\tilde{\mathbf{x}}}\right)\left(U^{ad}_{\text{CNOT}}\right)\left(\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle^a \otimes |0\rangle^d\right) \\ &= \left(I^a \otimes U^d_{\tilde{\mathbf{x}}}\right)\left(\frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle^a \otimes |i\rangle^d\right) \\ &= \frac{1}{\sqrt{M}}\sum_{i=0}^{M-1}|i\rangle^a|\tilde{\mathbf{x}}_i\rangle^d = |\Psi_1\rangle. \end{aligned} \tag{7}$$

PQCs for $U_{\tilde{\mathbf{x}}}$

For a concrete illustration, we consider using the variational algorithm to design a quantum circuit for the unitary operation $U_{\tilde{\mathbf{x}}}(\boldsymbol{\theta})$ by optimizing the tuneable parameters $\boldsymbol{\theta}$. Once the optimal parameters are learned to minimize the loss function, as given in Eq. (8), the PQC $U_{\tilde{\mathbf{x}}}(\boldsymbol{\theta})$ is then a candidate for $U_{\tilde{\mathbf{x}}}$,

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L} = \arg\min_{\boldsymbol{\theta}} \sum_i (1 - |\langle i, \tilde{\mathbf{x}}_i|U(\boldsymbol{\theta})|0\rangle^{ad}|^2). \tag{8}$$

Figure 2 offers one possible circuit structure for $U_1(\boldsymbol{\theta})$. Recall that the left Hadamard and U_{CNOT} gates convert the initial state $|0\rangle^a|0\rangle^d$ to an entangled state $\frac{1}{\sqrt{M}}\sum_i |i\rangle^a|i\rangle^d$. Then the left parametrized quantum circuit $U_{\tilde{\mathbf{x}}}(\boldsymbol{\theta})$ generates the predicted probability amplitude that approximates $|\tilde{\mathbf{x}}_i\rangle$ for each index $|i\rangle$, thereby accomplishing the "data-loading" process. By comparing the predicted probability distribution with the orthogonalized data, typically by the evaluation of the mean-squared error (MSE) loss, the circuit can be optimized to derive the optimal parameters.

It is important to emphasize that the width of the PQCs is exclusively determined by the number of "data-qubits." Additionally, as the number of data records grows, the depth of the PQCs is also expected to increase. Although it is difficult to know precisely how deep a circuit has to extend, we assume a polynomial scaling as commonly assumed for variational quantum algorithms.

C. QRAM circuit for the reverse transformation

The objective of the QRAM circuit is to prepare the specific target state given in Eq. (1). Therefore, it is necessary to introduce a quantum circuit, denoted as U_2, to convert the orthogonalized superposed state $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\tilde{\mathbf{x}}_i\rangle$ back to the target state $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\mathbf{x}_i\rangle$. The entire framework, including the execution of U_1 and U_2, is depicted in Fig. 3.

Now, we present a comprehensive exposition on the design of U_2. More specifically, according to Eq. (2), one can easily obtain

$$\mathbf{x}_i = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1}(S^{1/2})_{ij}(S^{-1/2})_{ji}\mathbf{x}_i = \sum_{j=0}^{M-1}(S^{1/2})_{ij}\tilde{\mathbf{x}}_j. \tag{9}$$

Let us denote $S^{1/2}$ as A in this context, where A_{ij} represents the (i, j)th element of A, and I^d represents the identity matrix

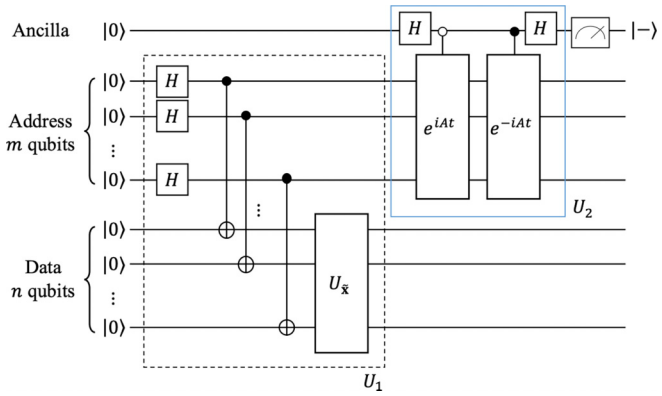

FIG. 3. The schematic diagram of the quantum circuit for our proposed QRAM. The left black dotted square represents the execution of U_1 in Fig. 2, and the right blue solid square represents the execution of U_2.

which is applied to the data qubits. As $|\Psi_1\rangle = \frac{1}{\sqrt{M}}\sum_i |i\rangle|\tilde{\mathbf{x}}_i\rangle$, we have

$$
\begin{aligned}
&(A^a \otimes I^d)|\Psi_1\rangle \\
&= \frac{1}{\sqrt{M}}(A^a \otimes I^d)\sum_i |i\rangle^a|\tilde{\mathbf{x}}_i\rangle^d \\
&= \frac{1}{\sqrt{M}}\begin{bmatrix} A_{00}I & \cdots & A_{0,M-1}I \\ \vdots & \ddots & \vdots \\ A_{M-1,0}I & \cdots & A_{M-1,M-1}I \end{bmatrix}\begin{bmatrix} \tilde{\mathbf{x}}_0 \\ \vdots \\ \tilde{\mathbf{x}}_{M-1} \end{bmatrix} \\
&= \frac{1}{\sqrt{M}}\begin{bmatrix} \sum_{j=0}^{M-1} A_{0j}\tilde{\mathbf{x}}_j \\ \vdots \\ \sum_{j=0}^{M-1} A_{M-1,j}\tilde{\mathbf{x}}_j \end{bmatrix} = \frac{1}{\sqrt{M}}\begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_{M-1} \end{bmatrix} \\
&= \frac{1}{\sqrt{M}}\sum_i |i\rangle^a|\mathbf{x}_i\rangle^d := |\Psi_T\rangle, \qquad (10)
\end{aligned}
$$

where $|\Psi_T\rangle$ denotes the target quantum state, which is stored within the desired QRAM. Consequently, after applying A on the address qubits, we can convert $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\tilde{\mathbf{x}}_i\rangle$ to $\frac{1}{\sqrt{M}}\sum_i |i\rangle|\mathbf{x}_i\rangle$.

More specifically, to apply the nonunitary transformation A on the system, we introduce an ancilla qubit on the top of the circuit as shown in Fig. 3. Denote $|+\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle)$ and $|-\rangle = 1/\sqrt{2}(|0\rangle - |1\rangle)$. Then the Hadamard gate H can convert $|0\rangle$ and $|1\rangle$ to $|+\rangle$ and $|-\rangle$, respectively, and vice versa. Given that A is a Hermitian operator, the matrices e^{-iAt} and e^{iAt} are both unitary. Now we introduce a controlled unitary $U_2 \in \mathbb{R}^{(M+1)\times(M+1)}$ satisfying

$$
\begin{aligned}
&(U_2 \otimes I^d)|0\rangle^c|\Psi_1\rangle \\
&= \frac{1}{\sqrt{2}}H^c(|0\rangle^c(e^{iAt} \otimes I^d)|\Psi_1\rangle + |1\rangle^c(e^{-iAt} \otimes I^d)|\Psi_1\rangle) \\
&= \frac{1}{\sqrt{2}}|+\rangle(e^{iAt} \otimes I^d)|\Psi_1\rangle + \frac{1}{\sqrt{2}}|-\rangle(e^{-iAt} \otimes I^d)|\Psi_1\rangle \\
&= |0\rangle(\cos(At) \otimes I^d)|\Psi_1\rangle + |1\rangle i(\sin(At) \otimes I^d)|\Psi_1\rangle. \qquad (11)
\end{aligned}
$$

After projecting the ancilla qubit onto the state $|1\rangle$, we get the following $|\Psi_2\rangle$:

$$
|\Psi_2\rangle = \frac{1}{\sqrt{P}}(\sin(At) \otimes I^d)|\Psi_1\rangle, \qquad (12)
$$

where $P = \mathrm{Tr}([\sin^\dagger(At)\sin(At) \otimes I^d]|\Psi_1\rangle\langle\Psi_1|) = \frac{1}{M}\mathrm{Tr}(\sin^\dagger(At)\sin(At)) = \frac{1}{M}||\sin(At)||_F^2$, with the fact that $|\Psi_1\rangle\langle\Psi_1| = \frac{1}{M}\sum_{ij}|i,\tilde{\mathbf{x}}_i\rangle\langle j,\tilde{\mathbf{x}}_j| = \frac{1}{M}I_M$ (I_M denotes the $M \times M$ identity matrix). And it is worth noting that $P = \frac{1}{M}||\sin(At)||_F^2$ is exactly the success probability of projecting the ancilla qubit onto the state $|1\rangle$, where $||\cdot||_F$ presents the Frobenius norm of the matrix.

In the case in which $t||A||$ is sufficiently small, the following approximation holds, $|\Psi_2\rangle \approx (A \otimes I)|\Psi_1\rangle = |\Psi_T\rangle$. Upon projecting the ancilla qubit into $|1\rangle$, the desired QRAM state is successfully prepared.

Once again, we can either employ parametrized quantum circuits or quantum circuit compiling methods to effectively construct the quantum circuit required for executing U_2 [20,22].

D. Analysis

In this section, we shall perform a comprehensive analysis of the proposed method, including complexity, error, and the success probability. For the sake of discussing more extensive scenarios, we will employ the notations $k = \max(m, n)$ and $p = \min(m, n)$ in our further discussions.

Complexity analysis

The consumption of the quantum resources encompasses qubits and quantum gates necessary for executing the quantum circuit. As previously discussed, our proposed framework requires one ancilla qubit. Consequently, the overall memory complexity is $O(1)$.

Regarding the gate complexity, the U_1 stage requires p Hadamard gates, p CNOT gates and the PQCs for $U_{\tilde{\mathbf{x}}_i}$. If the number of layers in the PQC is denoted as L, then $O(kL)$ single rotations and CNOT gates are required when using the hardware-efficient ansatz (HEA) in Fig. 2. Empirically, L is observed as a polynomial function of k within the framework of variational quantum circuits. This choice stems from the tolerance associated with the precision of generating quantum states. In situations in which precision can be flexibly adjusted, it becomes possible to strike a balance between efficiency and accuracy by reducing the circuit depth L. This situation may be more suitable for certain QML applications, where only retaining the main features of the input data is necessary. In this sense, the number of quantum gates for executing the U_1 stage is $O(poly(k))$. Considering the U_2 stage, to efficiently simulate the unitary e^{iAt} using the method in Ref. [22], $O(poly(p))$ elementary quantum gates are needed. It is worth noting that the controlled e^{iAt} operation and the execution of $U_{\tilde{\mathbf{x}}_i}$ can be performed in parallel. As a result, the overall gate complexity is $O(poly(k))$, primarily governed by the larger number of address qubits and data qubits.

While rigorously determining the computational complexity of training the PQCs is not a straightforward task, we know the complexity is given by $O(N_t N_m)$, where N_t denotes the

number of iterative training steps required to obtain a fully minimized loss function, and N_m represents the number of measurements performed at each step.

Error and success probability analysis

For the sake of simplicity, we have omitted the obvious identity matrix I, which should be applied on the data qubits whenever it can be easily understood in the following discussion. Referring to Eqs. (10) and (12), our goal is to get the target quantum state $|\Psi_T\rangle = A|\Psi_1\rangle$, but our quantum circuit actually outputs the quantum state $|\Psi_2\rangle = \frac{1}{\sqrt{P}}\sin(At)|\Psi_1\rangle$.

Now let us define an error ϵ to bound the infidelity $1 - |\langle\Psi_T|\Psi_2\rangle|^2$:

$$\begin{aligned}\epsilon &\geqslant 1 - \frac{|\langle\Psi_1|A^\dagger \sin(At)|\Psi_1\rangle|^2}{P} \\ &= 1 - \frac{[\mathrm{Tr}(A^\dagger \sin(At)|\Psi_1\rangle\langle\Psi_1|)]^2}{\mathrm{Tr}(\sin^2(At))/M} \\ &= 1 - \frac{[\mathrm{Tr}(A^\dagger \sin(At))]^2}{M\mathrm{Tr}(\sin^2(At))} \\ &= 1 - \frac{\left(\sum_{i=1}^M \sigma_i \sin(\sigma_i t)\right)^2}{M\sum_{i=1}^M \sin^2(\sigma_i t)}, \end{aligned} \tag{13}$$

where σ_i is the ith singular value of the matrix A. In short, after specifying a bound ϵ on the state infidelity, we can select t to make sure this bound is satisfied. Once t is set, the success probability for projecting the ancilla qubit on $|1\rangle$ is straightforwardly given by $P = \frac{1}{M}||\sin(At)||_F^2$.

Given Eq. (13), it is clear that the condition number of the matrix $A = \sqrt{S}$ is lower-bounded by 1. This well-conditioned case happens when $\sigma_i = 1$, and it can happen when the data-encoded states are all orthogonal to begin with. For this ideal case, $\epsilon = 0$ always holds regardless of the number of data points M and the choice for the hyperparameter t. The success probability P can be easily optimized to a high value, too. From an alternate perspective, in the scenario where all data points exhibit orthogonality, the desired QRAM consequently obviates the necessity for the U_2 stage, directly resulting in a success probability of 1. On the other hand, for an ill-conditioned A (corresponding to many $\sigma_i \approx 0$ when many basis vectors are linearly dependent), then the error ϵ is very difficult to suppress in the big-data limit when $M \gg 1$. This is because $\epsilon \geqslant 1 - \frac{\mathrm{Tr}(A^2)\mathrm{Tr}(\sin^2(At))}{M\mathrm{Tr}(\sin^2(At))} = 1 - \frac{||A||^2}{M}$, and the second term on the right side scales with $||A||^2/M$ factor. In addition, the success probability P will scale with $1/M \ll 1$ in the ill-conditioned limit.

III. NUMERICAL SIMULATIONS

This section focuses on the numerical simulations to assess the effectiveness of our proposed method, which are performed with PENNYLANE [23]. Here we utilize two data sets, namely the handwritten digits data set and the Iris data set, obtained from sklearn [24]. The handwritten digits data set is selected to represent the case in which $m \leqslant n$, while the Iris data set is chosen to represent the case in which $m > n$. These specific cases align with the theoretical analysis discussed in Sec. II. Moreover, we incorporate an experimental investigation into the impact of noise on quantum circuit performance across diverse configurations using random data sets.

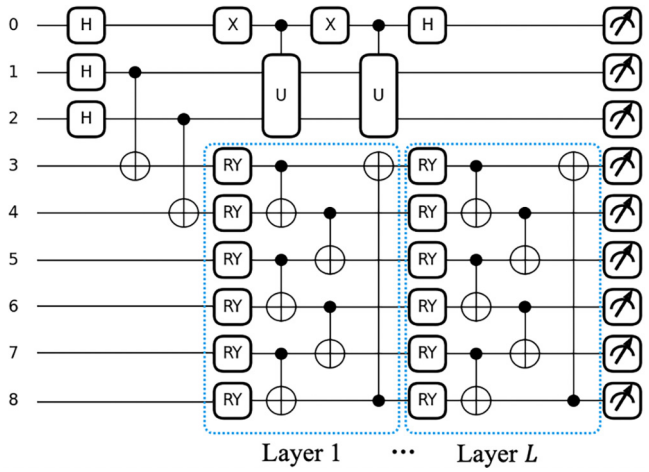

FIG. 4. The simplified quantum circuit for loading four handwritten digits data points. The circuit consists of various qubits with designated roles: qubit 0 serves as the ancilla qubit, qubits 1–2 function as the address qubits, and qubits 3–8 represent the data qubits. Notably, the PQCs with L layers, enclosed within blue dotted squares, solely operate on the data qubits. To enhance clarity, intermediate layers of the PQCs have been omitted in the illustration. Additionally, the controlled unitary U_2 exclusively acts on the ancilla and address qubits, as depicted in the circuit layout.

A. Handwritten digits data set

Each data point in the handwritten digits data set is represented as an 8×8 image of a digit, which is then transformed into an $N = 64$-dimensional vector. Consequently, loading a single data point requires $n = \log_2 N = 6$ qubits. In the following example, we choose $M = 4$ data points, implying that $m = \log_2 M = 2$, thereby satisfying the case in which $m \leqslant n$.

The quantum circuit that exemplifies the process of our proposed QRAM, as depicted in Fig. 3, is specifically illustrated in Fig. 4. As depicted in the circuit layout, it is evident that the PQCs solely operate on the data qubits (qubits 3–8), where the U_2 solely acts on the ancilla and address qubits (qubits 0–2). This clear separation of operations on qubits ensures an efficient parallel processing of the quantum circuit.

Now, we look at how well the proposed QRAM circuit performs in loading four handwritten digits data points. Figure 5 provides a clear comparison between the original input data and the output quantum state of the QRAM circuit, as displayed in the first and fourth columns, respectively. Additionally, the second column represents the orthogonalized data, which are computed by the operator $S^{-1/2}$, while the third column represents the output quantum state after executing the operator U_1 in Sec. II B. In this example, the PQC comprises $L = 36$ layers, and the final fidelity is at least 99%. Here, we choose the hyperparameter $t = 0.68$ for executing the unitary $e^{i\sqrt{S}t}$ and the success probability of measuring the ancilla to be the desired state with probability 32.51%.

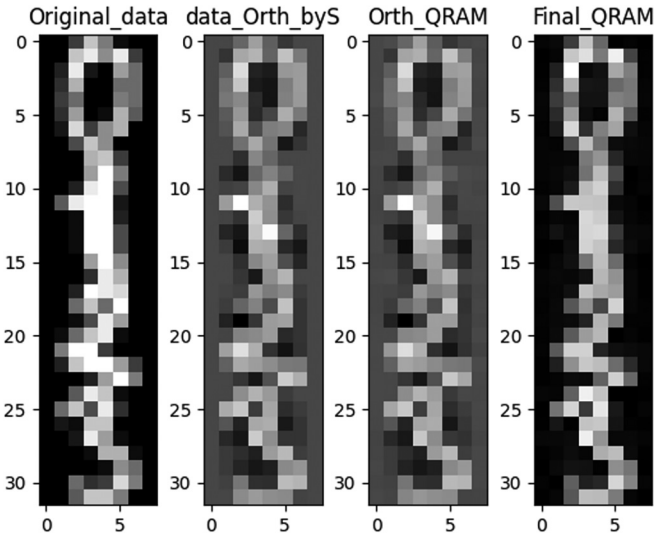

FIG. 5. An illustration of the handwritten digits obtained from our proposed QRAM. Each column represents a specific stage of the data-processing pipeline. The first column represents the original four handwritten digits, providing a reference for comparison. The second column represents the orthogonalized data, obtained through classical computation using the operator $S^{-1/2}$. The third and fourth columns represent the output quantum states of the QRAM circuit. Specifically, the third column represents the quantum state after performing the U_1 stage, and the fourth column represents the quantum state after the whole QRAM circuit. It is worth noting that the QRAM circuit achieves a fidelity of at least 99% when utilizing $L = 36$ layers of PQCs. Moveover, the success probability of obtaining the desired state is measured to be 32.51% with a parameter value of $t = 0.68$.

B. Iris data set

The dimensionality of each data point of the Iris data set is $N = 4$, resulting in $n = \log_2 N = 2$ qubits. In the following example, we choose $M = 16$ data points, implying that $m = \log_2 M = 4$, thereby satisfying the case in which $m > n$.

Figure 6 illustrates the quantum circuit designed for loading 16 Iris data points. In this approach, we employ the data grouping method to achieve data orthogonalization. The key step involves swapping the values of m and n. This transformation converts each set of $M/N = 4$ original data points into a grouped-data (for a detailed explanation, see Appendix B). Consequently, we obtain $M_g = M/4 = 4$ grouped-data points, each with a dimension $N_g = N \times 4 = 16$, where the subscript g denotes "grouping." An illustration of this grouped-data is depicted in Fig. 7(b), where each row represents a "grouped-data." Subsequently, these grouped-data points are expanded into an MN-dimensional vector for further computations. Upon returning to Fig. 6, after the grouping process, the "grouped-address" qubits now consist of $m_g = \log_2 M_g = 2$ qubits, ranging from 1 to 2, while the "grouped-data" qubits consist of $n_g = \log_2 N_g = 4$ qubits, ranging from 3 to 6.

Now, we also look at how well the proposed QRAM circuit performs in loading 16 Iris data points. Figure 7 provides a clear comparison between the original input data and the output quantum state of the QRAM circuit. In this example, the PQC consists of $L = 12$ layers, and the final fidelity is at least

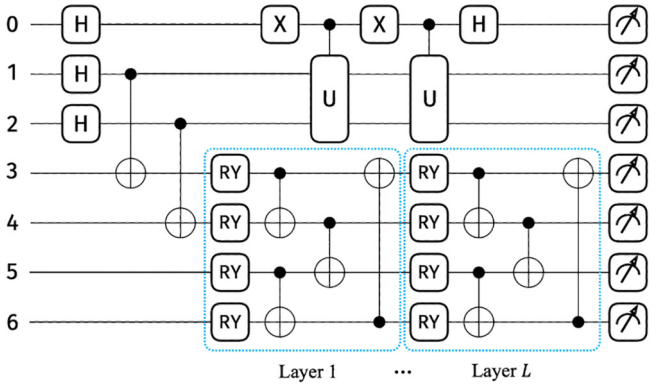

FIG. 6. The simplified quantum circuit for load 16 Iris data points. Qubit 0 serves as the ancilla qubit, qubits 1–2 function as the "grouped-address" qubits, and qubits 3–6 represent the "grouped-data" qubits. However, in fact, qubits 1–4 function as the actual address qubits, and qubits 5–6 represent the actual data qubits, which is different with Fig. 4. Notably, the PQCs with L layers, enclosed within blue dotted squares, operate on both the $m - n$ actual address qubits and the n actual data qubits. To enhance clarity, intermediate layers of the PQCs have been omitted in the illustration.

99.8%. For the purpose of executing the unitary operation $e^{i\sqrt{S}t}$, we select the hyperparameter $t = 0.8$, and the success probability of measuring the ancilla qubit in the desired state is determined to be 25.4%.

C. Numerical analysis

Now we present a comparative analysis of the experimental results derived from two distinct cases. An illustration of this comparative analysis is depicted in Fig. 8. The figure showcases the trends of success probability and fidelity concerning various values of t. In particular, Fig. 8(a) presents the experimental results obtained from the handwritten digits data set, corresponding to the case illustrated in Sec. III A. Additionally, Fig. 8(b) displays the results from the Iris data set, as discussed in Sec. III B. As demonstrated in Sec. II D, the success probability for projecting the ancilla qubit onto $|1\rangle$ is directly determined by $P = \frac{1}{M}||\sin(At)||_F^2$. Consequently, we can readily visualize the probability trends corresponding to the variable t using this equation. Additionally, fidelities are calculated straightforwardly by computing the inner product between the final quantum state output in the numerical simulations and the original classical data for different values of t. For both experimental cases, the variable t is varied across a range of 0–2, with 100 sampling points. Indeed, we are particularly interested in the region where t is relatively small and fidelity is comparatively high.

The success probability (depicted by the blue solid line) shows fluctuations with peaks and valleys. This pattern aligns with the mathematical properties of the square of the sine function. Conversely, predicting the fidelity trend solely from theoretical analysis presents a challenge. However, experimental observations from both figures indicate that the fidelity, depicted by the red dashed line, typically decreases as t increases, especially when t is relatively small.

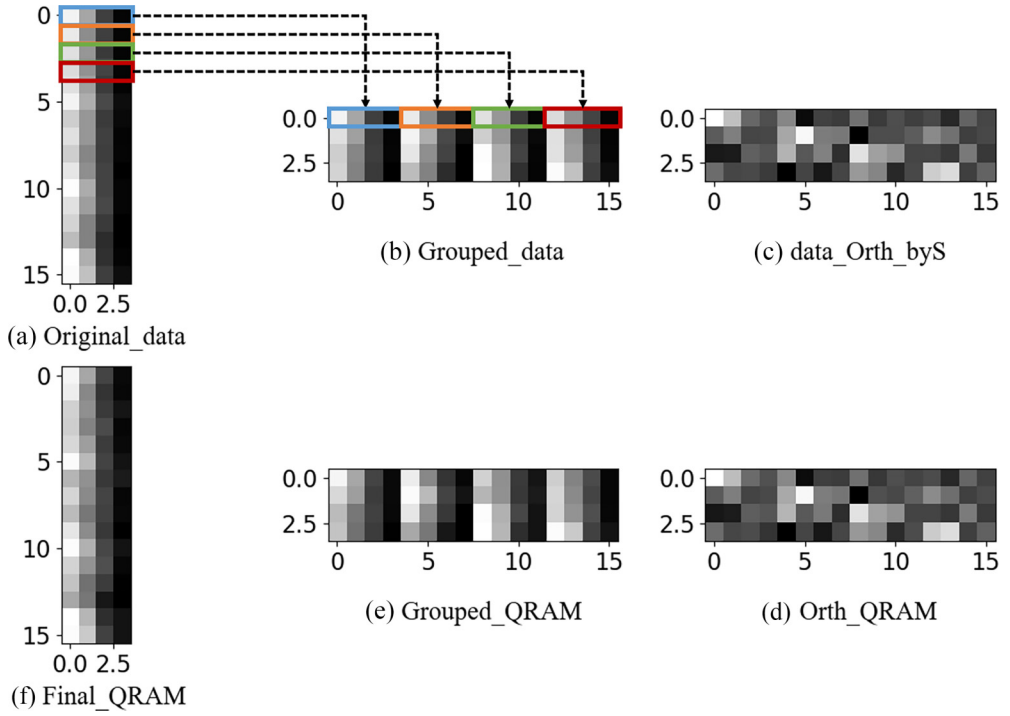

FIG. 7. An illustration of the Iris data points obtained from our proposed QRAM. The label of the subfigure represents a specific stage of the data-processing pipeline from (a) to (f): (a) The original 16 Iris data points; each has dimension 4 (in a row). (b) The 4 grouped Iris data points; each "grouped-data" has dimension 16 (in a row). It is essential to understand that, for example, the first row in (b) is derived by consolidating the data from the first four rows in (a). (c) The orthogonalized data, obtained through classical computation using the operator $S^{-1/2}$. (d) The quantum state after performing the U_1 stage. (e) The output of the QRAM storing the grouped Iris data points. (f) The final output of our proposed QRAM. Eventual outcome shows that the QRAM circuit achieves a fidelity of at least 99.8% when utilizing $L = 12$ layers of PQCs. Moveover, the success probability of obtaining the desired state is measured to be 25.4% with a parameter value of $t = 0.8$.

Recall that the primary goal of our experiments is to achieve higher success probability while maintaining high fidelity. Therefore, by considering the trends illustrated in Fig. 8, an optimal choice for the value of t can be determined. For example, in Fig. 8(a), when the fidelity exceeds 0.99, the probability increases with the increment of t, leading us to choose $t = 0.68$ as demonstrated in Sec. III A. Conversely, in Fig. 8(b), when the fidelity surpasses 0.99, the probability initially increases with t but then begins to decrease again. As a result, we opt for the value of $t = 0.8$, which corresponds to the first peak of the success probability, as illustrated in Sec. III B.

D. Noisy analysis

Our experimental design investigated the capability of a quantum circuit to represent an ensemble of random target states across various qubit counts. The noise experiments were conducted following the methodology outlined in Ref. [25]. We considered six cases: $n = 3$ and $m = 2$, $n = 4$ and $m = 2$, $n = 4$ and $m = 3$, $n = 5$ and $m = 2$, $n = 5$ and $m = 3$, and $n = 6$ and $m = 2$. Three different types of noise were explored: phase damping, amplitude damping, and depolarizing channels, with noise levels $\gamma \in \{0.001, 0.01, 0.05, 0.1\}$.

To introduce noise into the quantum circuit, we utilized PENNYLANE's noise gate, following the approach described in Ref. [25]. Specifically, noise channels were added after each two-qubit or multiqubit gate in a layer of the circuit. Additionally, we employed the reoptimization technique described in Ref. [25]. First, the circuit was optimized in a noiseless environment to determine the optimal parameters for the ideal case. These parameters were then used to evaluate the fidelity using the noisy circuit (non-reoptimized). Subsequently, starting from these noiseless optimum parameters, the circuit was reoptimized with the noise channels included (reoptimized).

The experimental results are presented in Fig. 9, where each row represents a specific problem size (i.e., identical n and m), and each column represents a specific noise type (i.e., phase damping, amplitude damping, and depolarizing channels). Solid lines represent the fidelity after reoptimization, while dashed lines represent the fidelity without reoptimization. Different colors of lines correspond to different gamma values.

Based on the experimental findings, we draw the following conclusions:

(i) Noiseless Conditions: increasing the number of layers results in fidelity improvements surpassing 0.99, with fidelity levels exceeding 0.99 for $n = 3, m = 2$ with 8 layers; $n = 4, m = 2$ with 11 layers; $n = 4, m = 3$ with 18 layers; $n = 5, m = 2$ with 18 layers; $n = 5, m = 3$ with 34 layers; and $n = 6, m = 2$ with 34 layers.

We can fit a polynomial function to model the relationship between the number of qubits (x) and the required number of layers (y) to achieve high fidelity. Using polynomial fitting, the relationship can be expressed as $y = poly(x)$. Current data

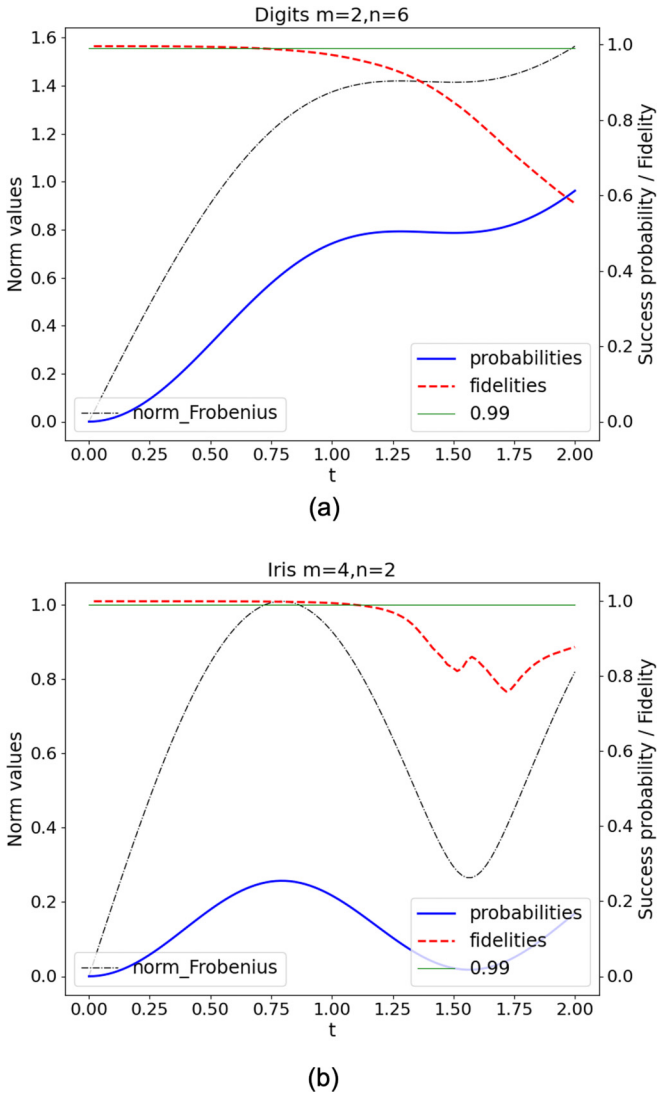

FIG. 8. The variation trends of success probability and fidelity with different values of t. Part (a) presents the experimental results on the handwritten digits data set, while (b) shows the results on the Iris data set. The thick red dashed line represents the trend of the fidelity, while the thick blue solid line showcases the variation of the success probability. For clarity, the thin black dashed line indicates the Frobenius norm value of $\sin(At)$, while the thin green solid line represents a reference line for fidelity at 0.99.

suggest that a cubic polynomial provides a reasonable fit. However, whether this model holds for higher qubit counts requires further experimentation. Nonetheless, the existing results indicate the potential for resource prediction in the first stage. In the second stage, the resources required for quantum gates are determined by the preparation of the unitary $e^{i\sqrt{S}t}$. The gate count and depth depend on the simulation technique used. Reference [26] proposed an effective method for Hamiltonian simulation by utilizing quantum signal processing (QSP). This strategy involves carefully reducing overhead costs at each step to run a complete QSP protocol on noisy quantum hardware. By implementing a polynomial approximation of the time evolution operator, this method provides a viable approach for efficiently preparing $e^{i\sqrt{S}t}$ even in the presence of noise.

(ii) Noise Impact on Fidelity and Layer Count Optimization: The introduction of noise prompts fidelity enhancement to a certain extent by necessitating a reduction in the number of layers. For example, in a noiseless scenario, for $n = 3, m = 2$, increasing the layer count from 8 to 11 may yield a slight fidelity increase. However, in the presence of noise (e.g., with $\gamma = 0.001$ for depolarizing channels), it is observed that with eight layers, the fidelity peaks. Subsequently, increasing the layer count results in fidelity degradation. Similar phenomena are also observable in other problem sizes, especially pronounced in the case of depolarizing channels.

(iii) Reoptimization Impact on Fidelity with Noise: Maintaining a constant layer count, reoptimization exerts minimal influence on fidelity enhancement, as evidenced by instances in which the solid line surpasses the dashed line. From the conclusion above in (ii), it is evident that reducing the layer count during reoptimization has a more pronounced effect on fidelity improvement.

(iv) Fidelity Decline with Qubit and Layer Expansion: At equivalent noise levels (with equal γ), fidelity experiences a rapid decline with increasing qubit and layer counts. This trend is observable across different problem sizes depicted by various subgraphs, which reflect varying n and m.

(v) Fidelity Variation Across Noise Types and Noise Levels: Different types of noise exert distinct impacts on fidelity, with phase damping demonstrating the least effect and depolarizing noise exhibiting the strongest influence. Specifically, despite fidelity decreasing with all three types of noise, fidelity is highest under phase damping, followed by amplitude damping, and lowest under depolarizing noise, when considering the same noise level (with equal γ). This trend is evident when comparing lines of the same color (e.g., blue) across subgraphs representing identical problem sizes (n and m).

Moreover, fidelity remains relatively high at lower qubit counts when γ is set to 0.001. However, fidelity significantly degrades with higher γ values, such as 0.01, 0.05, and 0.1.

(vi) Comparative Analysis of Configurations: In configurations with the same total qubit count (i.e., $m + n$), such as $n = 5, m = 3$ and $n = 6, m = 2$, $n = 4, m = 3$ and $n = 5, m = 2$, fidelity exhibits some variability at lower layer counts. Specifically, configurations like $n = 6, m = 2$ and $n = 5, m = 2$ demonstrate higher fidelity compared to $n = 5, m = 3$ and $n = 4, m = 3$ when the layer count is low. However, as the layer count increases, the performance of these configurations converges, resulting in similar fidelity outcomes. This phenomenon is observable by comparing subgraphs in the same column (i.e., the same noise type) for configurations with the same total qubit count.

In conclusion, our study highlights how noise influences quantum circuit performance across diverse configurations. This analysis of fidelity trends offers insights into optimizing circuits for practical applications, contributing to ongoing efforts to enhance quantum computing technologies.

IV. CONCLUSION

In this work, we present a QRAM framework that approximately loads a set of classical data into a compact quantum

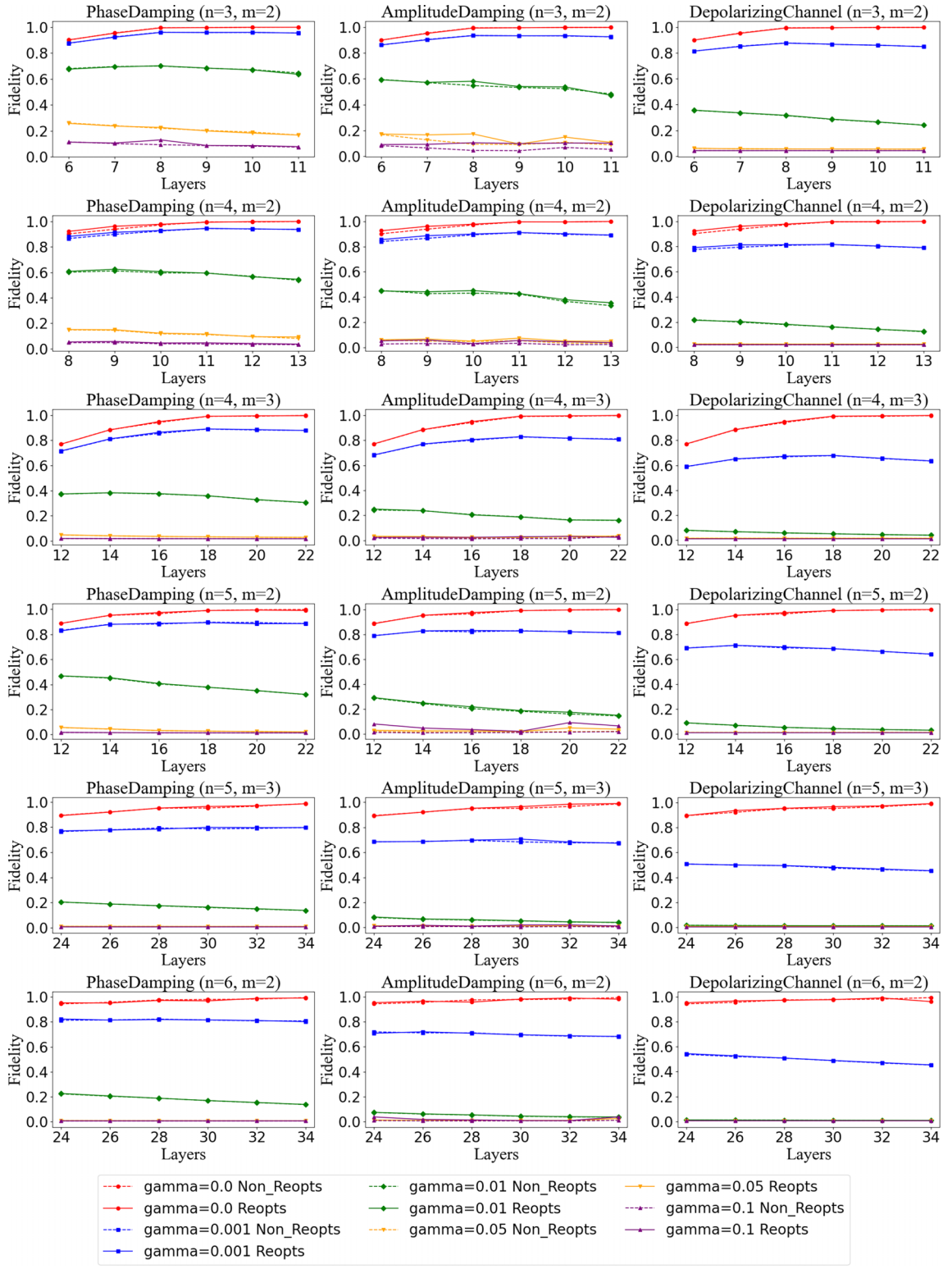

FIG. 9. Fidelity for different noise types, noise levels, and layer counts. Each column represents the fidelity for a specific noise type: the first column represents fidelity under phase damping noise, the second column represents fidelity under amplitude damping noise, and the third column represents fidelity under depolarizing channel noise. Each row represents a different problem size. For example, the first row represents the case in which the data qubit number is $n = 3$ and the address qubit number is $m = 2$. Other rows follow similarly. The lines with different colors represent different noise levels: red for $\gamma = 0$, blue for $\gamma = 0.001$, green for $\gamma = 0.01$, yellow for $\gamma = 0.05$, and purple for $\gamma = 0.1$. The solid lines represent results with reoptimization, while the dashed lines represent results without reoptimization.

circuit. The QRAM process entails a classical preprocessing step to orthogonalize the classical data, followed by loading these orthogonalized data using a parametrized quantum circuit. Subsequently, a set of controlled unitaries is employed to transform the quantum state into the desired target state, effectively loading the original data with their indices in the quantum superposition. With the support of classical computing, the proposed QRAM framework gives a potentially viable quantum-classical approach in the near-term NISQ era.

To demonstrate that the proposed QRAM circuit is practical for approximately loading multiple classical data, we illustrate numerical simulations on the handwritten digits and Iris data set. As clearly shown in Sec. III, a $poly(m, n)$-depth quantum circuit can efficiently approximate the loading of multiple data into a quantum superposition using amplitude encoding.

We remind the reader that the overall framework for loading the orthogonalized data into a quantum superposition and reverting the orthogonalized data to the original ones is not restricted to our proposed approach discussed in the main text, as there are alternative ways to construct U_1 and U_2. For instance, one can design U_1 without using PQCs and variational optimizations, since we can explicitly write down the unitary matrix for classical data (there is no issue of "exponentially" large Hilbert space as in simulating a genuine quantum system). As for U_2, one may consider other techniques to generate a linear combination of unitaries to realize the nonunitary transformation. As exposed in our analysis and numerical demonstrations, there is no reason that the success probability of the state preparation will diminish with the sizes of data. Additionally, a range of quantum circuit compressing algorithms may also help us to further reduce the depth of the QRAM circuit [27–31].

Our experiments highlight the significant impact of noise on quantum circuit performance. While increasing the layer count improves fidelity under noiseless conditions, reducing the layer count during reoptimization has a more pronounced effect on fidelity improvement under noisy conditions. Currently, as the circuit depth increases, fidelity decreases rapidly with higher noise levels, indicating a low tolerance to noise. However, with advancements in quantum hardware and algorithm optimizations, we believe there is potential to further advance the practical applications of these algorithms.

Our method defies the previously established mainstream theoretical conclusion on the subject of loading quantum data as we heavily leverage classical preprocessing to significantly reduce the reliance on quantum resources to the bare minimum. We believe this work can efficiently enable the loading of many nontrivial data sets into quantum circuits and facilitate the development of quantum algorithms, such as quantum machine learning.

ACKNOWLEDGMENTS

This work was supported by the Funding of National Natural Science Foundation of China (Grant No. 62101270), and Natural Science Foundation of the Jiangsu Higher Education Institutions, China (Grant No. 21KJB520010).

APPENDIX A: RELATED WORKS

Various QRAM structures have been proposed, and we summarize them as follows.

1. Bucket-brigade QRAM

The "bucket-brigade" architecture uses a binary tree structure to encode M N-dimensional vectors into M leaf nodes (memory cells), where a three-level system "qutrit" is introduced to facilitate the routing of accessing leaves, yielding only $O(m)$ quantum switch activations [2]. Several proposals for experimental implementations of the bucket-brigade QRAM using photonic or acoustic systems have been put forth [7–9]. The follow-up papers [10,11] also proposed a circuit-based implementation of the bucket-brigade QRAM, and they presented a number of different circuit families that perform the task of a QRAM. They also discussed whether all the components are required to be error-corrected for maintaining a high-fidelity operation. Results showed that the fault-tolerant components might not be necessary for algorithms that require only polynomial queries to the memory (such as the quantum linear system algorithm [32,33]), but might be necessary for superpolynomial query algorithms (such as Grover's search [34], which is widely used as a subroutine in various quantum algorithms) [10]. Reference [12] proved that a noise-resilient QRAM can be implemented with realistically noisy devices. Specifically, the bucket-brigade architecture can be used to perform high-fidelity queries of large memories without the need for quantum error correction, provided physical error rates are sufficiently low.

In addition, a quantum walk is introduced to implement the bucket-brigade QRAM [13,14]. Reference [13] provides a bucket-brigade QRAM scheme based on a quantum walk, which requires only $O(m)$ steps to access and retrieve $O(M)$ data in the form of quantum superposition states. In this scheme, the bucket is represented as a quantum walker distinguished by $left$ and $right$ chirality, and quantum motion is executed on a full binary tree to transport the bucket to the specified memory cells. However, it still requires all $O(Mm)$ qutrits to be installed for the routing scheme. Reference [14] physically implement a bucket-brigade QRAM using a multiparticle continuous-time quantum walk with two internal states. Data with address information are dual-rail encoded into quantum walkers. This bucket-brigade QRAM processes M n-qubit data in a quantum circuit of depth $O(m\log(m+n))$ and requires $O(m+n)$ qubits and $O(M(m+n))$ quantum gate resources, which is more efficient than the conventional bucket-brigade QRAM, which requires $O(M+n)$ qubits and $O(m^2+mn)$ steps.

Still, it was questioned whether the bucket-brigade QRAM provides a genuine quantum advantage if we take into account all the required resources [35].

2. Circuit-based QRAM

In numerous quantum algorithms, QRAM is commonly utilized as the first step to load classical data, which are then fed into the subsequent circuit. Therefore, it is desirable to directly implement a QRAM in a quantum circuit model.

Möttönen *et al.* introduced a method to efficiently implement an amplitude encoding using uniformly controlled rotations, which costs $2^{m+n+2} - 4(m+n) - 4$ CNOT gates and $2^{m+n+2} - 5$ one-qubit elementary rotations to reconstruct the $(m+n)$-qubit QRAM-like model [6].

In recent years, Ref. [15] presented a circuit-based QRAM termed a flip-flop QRAM (FF-QRAM) to construct a quantum database of classical information. The FF-QRAM stores the data one by one, such that the complexity of constructing data is in $O(n)$ qubits and $O(Mn)$ steps, where M represents the number of classical data. Each datum is executed in three stages: flip, register, and flop. The flip stage is to set all the bits pertaining to a particular address and data point to 1 by a classically controlled Pauli-X gate, and the flop is the inverse of the flip. The midstage register is applied to store the angle of the data by the n-qubit controlled rotation $C^nR(\boldsymbol{\theta})$. Reference [16] proposed a strategy to load continuous data without postselection in $O(Mn)$ time. This method is inspired by the probabilistic quantum memory and FF-QRAM that completes the generalized amplitude encoding.

A range of circuit-based data-loading approaches can also be extended to implement QRAM [36–39]. The quantum resource consumption, whether in terms of qubits or quantum gates, is at least $O(MN)$ for data-loading of $m+n$ qubits.

3. PQC-based QRAM

With the growing emphasis on the application of parametrized quantum circuits (PQCs) in quantum machine learning during the noisy intermediate-scale quantum (NISQ) era [40,41], the recent emergence of research has focused on investigating the implementation of an approximate circuit-based QRAM [17–19]. Reference [17] introduces a novel approach for encoding classical data into quantum states approximately using a trainable entangling quantum generative adversarial network (EQ-GAN) circuit. The EQ-GAN employs entangling operations between the generator output and true quantum data to converge to the Nash equilibrium. However, the EQ-GAN solely generates synthetic data, rather than processing the addressable data. Similar to EQ-GAN QRAM, Ref. [18] proposed a trainable PQC-based QRAM. It takes address lines as input and gives out the corresponding data in these address lines as the output. Although it has been noted that a faster convergence can be achieved by loading images from QRAM and transmitting them to a QNN, the proposed architecture is limited to sequential access of data and does not scale efficiently for large data sets. In Ref. [19], we introduce a novel Hamiltonian-based data-loading (HDL) protocol that achieves high-fidelity data loading using only a shallow PQC without ancilla qubits. The HDL first identifies a Hamiltonian $\hat{H}$ whose unique ground state $|\psi\rangle$ represents the normalized data in the form of amplitude encoding. Then the target state is reconstructed with a PQC by utilizing methods like a variational quantum eigensolver to minimize energy. We explore the possibility of constructing a circuit-based QRAM to load multiple data with the HDL protocol.

It is noteworthy that the circuit-based QRAM or PQC-based QRAM structure resembles the implementation of data access through quantum circuits instead of relying on routing to physical storage devices. As a consequence, in quantum algorithms, this type of QRAM serves more as a subroutine that necessitates reexecution upon each utilization. Furthermore, compared with the circuit-based QRAM, which requires exponential resource consumption, the PQC-based methods transfer a huge portion of the computational costs to classical calculations, which is expected to reduce quantum resources to a polynomial level. Once we have learned the shallow circuit, we can repeatedly load classical data into various quantum algorithm tasks.

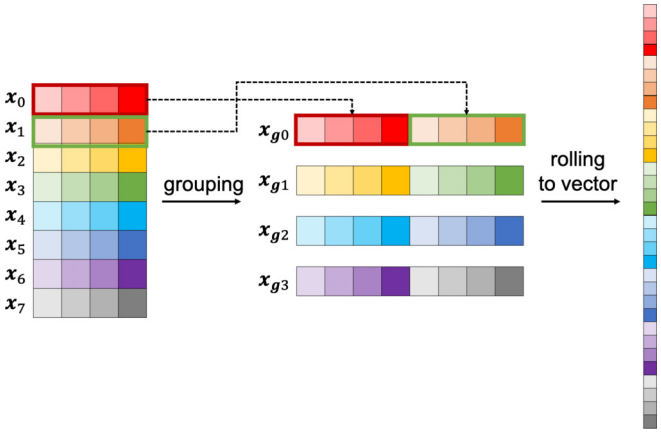

FIG. 10. An illustration of the data grouping method. Here, $\mathbf{x}_i$ represents the original data, where i ranges from 0 to $M-1$. The grouping operation combines M/N data points into one grouped-data, such as grouping $\mathbf{x}_0$ and $\mathbf{x}_1$ into a grouped-data $\mathbf{x}_{g0}$. Following this operation, the newly grouped-data points are rolled into a vector, which will be encoded into the quantum state in later stages, as discussed in Sec. II. Notably, the sequence of rolling, whether before or after grouping, does not impact the final quantum state generated.

APPENDIX B: DATA GROUPING METHOD

In this Appendix, we delve deeper into the data grouping method. In the case when $m > n$, we can simply swap the roles of m and n, as explained in Sec. II A. To enhance clarity, we provide an illustrative example of the grouping method in Fig. 10, where $M = 8$ and $N = 4$. The grouping operation combines $M/N = 2$ data points into one grouped-data, resulting in $M_g = 4$ grouped-data points and $N_g = 8$ dimension for each grouped-data. Specifically, in this context, $\lceil \log_2 M_g \rceil = 2$ represents the number of "grouped-address" qubits and $\lceil \log_2 N_g \rceil = 3$ represents the number of "grouped-data" qubits. The grouped-data set is then rolled into an $M \times N = M_g \times N_g = 32$-dimensional vector, which will be processed in subsequent stages, as discussed in Sec. II.

As depicted in Fig. 10, the order of rolling, whether before or after grouping, does not impact the final quantum state generated. The grouping method affects only the intermediate calculation process: constructing the matrix S based on the newly grouped-data and the orthogonalized grouped-data derived from S. Therefore, upon completing the entire process, we obtain optimized quantum circuits

capable of generating the desired quantum state that stores the original data along with their corresponding address indices.

Certainly, our proposed method is not confined to the strategy of swapping m and n. We have the flexibility to adjust the number of data points within each group as a factor of M, denoted as c. In this scenario, we have $M_g = M/c$ and $N_g = N \times c$. Importantly, this strategy only alters the circuits of U_1 and U_2, but it does not affect the final quantum state generated in the end.

[1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature (London) **549**, 195 (2017).

[2] V. Giovannetti, S. Lloyd, and L. Maccone, Quantum random access memory, Phys. Rev. Lett. **100**, 160501 (2008).

[3] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big feature and big data classification, Phys. Rev. Lett. **113**, 130503 (2014).

[4] B. Duan, J. Yuan, Y. Liu, and D. Li, Quantum algorithm for support matrix machines, Phys. Rev. A **96**, 032301 (2017).

[5] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. **10**, 631 (2014).

[6] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Transformation of quantum states using uniformly controlled rotations, Quantum Inf. Comput. **5**, 467 (2005).

[7] V. Giovannetti, S. Lloyd, and L. Maccone, Architectures for a quantum random access memory, Phys. Rev. A **78**, 052310 (2008).

[8] F. Y. Hong, Y. Xiang, Z. Y. Zhu, L. Z. Jiang, and L. N. Wu, Robust quantum random access memory, Phys. Rev. A **86**, 010306(R) (2012).

[9] C. T. Hann, C.-L. Zou, Y. Zhang, Y. Chu, R. J. Schoelkopf, S. M. Girvin, and L. Jiang, Hardware-efficient quantum random access memory with hybrid quantum acoustic systems, Phys. Rev. Lett. **123**, 250501 (2019).

[10] S. Arunachalam, V. Gheorghiu, T. Jochym-O'Connor, M. Mosca, and P. V. Srinivasan, On the robustness of bucket brigade quantum ram, New J. Phys. **17**, 123010 (2015).

[11] O. Di Matteo, V. Gheorghiu, and M. Mosca, Fault-tolerant resource estimation of quantum random-access memories, IEEE Trans. Quantum Eng. **1**, 4500213 (2020).

[12] C. T. Hann, G. Lee, S. M. Girvin, and L. Jiang, Resilience of quantum random access memory to generic noise, PRX Quantum **2**, 020311 (2021).

[13] R. Asaka, K. Sakai, and R. Yahagi, Quantum random access memory via quantum walk, Quantum Sci. Technol. **6**, 035004 (2021).

[14] R. Asaka, K. Sakai, and R. Yahagi, Two-level quantum walkers on directed graphs. II. Application to quantum random access memory, Phys. Rev. A **107**, 022416 (2023).

[15] D. K. Park, F. Petruccione, and J.-K. K. Rhee, Circuit-based quantum random access memory for classical data, Sci. Rep. **9**, 3949 (2019).

[16] T. M. De Veras, I. C. De Araujo, D. K. Park, A. J. Da Silva, Circuit-based quantum random access memory for classical data with continuous amplitudes, IEEE Trans. Comput. **70**, 2125 (2020).

[17] M. Y. Niu, A. Zlokapa, M. Broughton, S. Boixo, M. Mohseni, V. Smelyanskyi, and H. Neven, Entangling quantum generative adversarial networks, Phys. Rev. Lett. **128**, 220505 (2022).

[18] K. Phalak, J. Li, and S. Ghosh, Trainable PQC-based QRAM for quantum storage, IEEE Access **11**, 51892 (2023).

[19] B. Duan and C.-Y. Hsieh, Hamiltonian-based data loading with shallow quantum circuits, Phys. Rev. A **106**, 052422 (2022).

[20] A. M. Krol, A. Sarkar, I. Ashraf, Z. Al-Ars, and K. Bertels, Efficient decomposition of unitary matrices in quantum circuit compilers, Appl. Sci. **12**, 759 (2022).

[21] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature (London) **549**, 242 (2017).

[22] K. Li, Towards a NISQ algorithm to simulate Hermitian matrix exponentiation, arXiv:2105.13610.

[23] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. A. Narayanan, A. Asadi *et al.*, Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, Scikit-learn: Machine learning in python, J. Mach. Learn. Res. **12**, 2825 (2011).

[25] E. Fontana, N. Fitzpatrick, D. M. Ramo, R. Duncan, and I. Rungger, Evaluating the noise resilience of variational quantum algorithms, Phys. Rev. A **104**, 022403 (2021).

[26] Y. Kikuchi, C. McKeever, L. Coopmans, M. Lubasch, and M. Benedetti, Realization of quantum signal processing on a noisy quantum computer, npj Quantum Inf. **9**, 93 (2023).

[27] E. Kökcü, D. Camps, L. Bassman Oftelie, J. K. Freericks, W. A. de Jong, R. Van Beeumen, and A. F. Kemper, Algebraic compression of quantum circuits for hamiltonian evolution, Phys. Rev. A **105**, 032420 (2022).

[28] R. Dilip, Y.-J. Liu, A. Smith, and F. Pollmann, Data compression for quantum machine learning, Phys. Rev. Res. **4**, 043007 (2022).

[29] S. Gulania, Z. He, B. Peng, N. Govind, and Y. Alexeev, Quybean algebraic compiler for quantum circuit compression, in *2022 IEEE/ACM 7th Symposium on Edge Computing (SEC)* (IEEE, Piscataway, NJ, 2022), pp. 406–410.

[30] P. Rakyta and Z. Zimborás, Efficient quantum gate decomposition via adaptive circuit compression, arXiv:2203.04426.

[31] M. Ostaszewski, E. Grant, and M. Benedetti, Structure optimization for parameterized quantum circuits, Quantum **5**, 391 (2021).

[32] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).

[33] B. Duan, J. Yuan, C.-H. Yu, J. Huang, and C.-Y. Hsieh, A survey on hhl algorithm: From theory to application in quantum machine learning, Phys. Lett. A **384**, 126595 (2020).

[34] L. K. Grover, Quantum mechanics helps in searching for a needle in a haystack, Phys. Rev. Lett. **79**, 325 (1997).

[35] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, Quantum machine learning: A classical perspective, Proc. R. Soc. A **474**, 20170551 (2018).

[36] X. Sun, G. Tian, S. Yang, P. Yuan, and S. Zhang, Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis, IEEE Trans. Comput. Aided Des. Integr. Circ. Syst. **42**, 3301 (2023).

[37] S. Johri, S. Debnath, A. Mocherla, A. Singk, A. Prakash, J. Kim, and I. Kerenidis, Nearest centroid classification on a trapped ion quantum computer, npj Quantum Inf. **7**, 122 (2021).

[38] X.-M. Zhang, T. Li, and X. Yuan, Quantum state preparation with optimal circuit depth: Implementations and applications, Phys. Rev. Lett. **129**, 230504 (2022).

[39] X.-M. Zhang, M.-H. Yung, and X. Yuan, Low-depth quantum state preparation, Phys. Rev. Res. **3**, 043200 (2021).

[40] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, Quantum Sci. Technol. **4**, 043001 (2019).

[41] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).