algorithms

MDPI

Article

Optimizing Multidimensional Pooling for Variational Quantum Algorithms

Mingyoung Jeng, Alvir Nobel, Vinayak Jha, David Levy, Dylan Kneidel, Manu Chaudhary, Ishraq Islam, Evan Baumgartner, Eade Vanderhoof, Audrey Facer, Manish Singh, Abina Arshad and Esam El-Araby *

Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045, USA; mingyoungjeng@ku.edu (M.J.); islam.alvir@ku.edu (A.N.); vinayakjha@ku.edu (V.J.); david.levy@ku.edu (D.L.); dckneidel@ku.edu (D.K.); manu.chaudhary@ku.edu (M.C.); ishraq@ku.edu (I.I.); evan.baum@ku.edu (E.B.); eade.vanderhoof@ku.edu (E.V.); audfacer@ku.edu (A.F.); singhmanish5208@ku.edu (M.S.); abina.arshad@ku.edu (A.A.)

* Correspondence: esam@ku.edu

Abstract: Convolutional neural networks (CNNs) have proven to be a very efficient class of machine learning (ML) architectures for handling multidimensional data by maintaining data locality, especially in the field of computer vision. Data pooling, a major component of CNNs, plays a crucial role in extracting important features of the input data and downsampling its dimensionality. Multidimensional pooling, however, is not efficiently implemented in existing ML algorithms. In particular, quantum machine learning (QML) algorithms have a tendency to ignore data locality for higher dimensions by representing/flattening multidimensional data as simple one-dimensional data. In this work, we propose using the quantum Haar transform (QHT) and quantum partial measurement for performing generalized pooling operations on multidimensional data. We present the corresponding decoherence-optimized quantum circuits for the proposed techniques along with their theoretical circuit depth analysis. Our experimental work was conducted using multidimensional data, ranging from 1-D audio data to 2-D image data to 3-D hyperspectral data, to demonstrate the scalability of the proposed methods. In our experiments, we utilized both noisy and noise-free quantum simulations on a state-of-the-art quantum simulator from IBM Quantum. We also show the efficiency of our proposed techniques for multidimensional data by reporting the fidelity of results.

Keywords: quantum computing; convolutional neural networks; quantum machine learning; pooling layers

check for updates

Citation: Jeng, M.; Nobel, A.; Jha, V.; Levy, D.; Kneidel, D.; Chaudhary, M.; Islam, I.; Baumgartner, E.; Vanderhoof, E.; Facer, A.; et al. Optimizing Multidimensional Pooling for Variational Quantum Algorithms. *Algorithms* **2024**, *17*, 82. https://doi.org/10.3390/a17020082

Academic Editors: Bharat Rawal and Gopal Chaudhary

Received: 31 December 2023
Revised: 9 February 2024
Accepted: 13 February 2024
Published: 15 February 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

For performing machine learning (ML) tasks on multidimensional data, convolutional neural networks (CNNs) often outperform other techniques, such as multi-layer perceptrons (MLPs), with smaller model sizes, shorter training times, and higher accuracies [1,2]. One factor that contributes to the benefits of CNNs is the conservation of spatio–temporal data locality, allowing them to preserve only relevant data connections and remove extraneous ones [1,2]. CNNs are constructed using a sequence of convolution and pooling pairs followed by a fully connected layer [1]. In the convolution layer, filters are applied to input data for specific applications, and the pooling layers reduce the spatial dimensions in the generated feature maps [3]. The reduced spatial dimensions generated from the pooling layers reduce memory requirements, which is a major concern for resource-constrained devices [4,5].

Exploiting data locality in pooling could be beneficial in fields such as quantum-to-classical (Q2C) data decoding [6], audio–visual data compression [7,8], and, in particular, quantum machine learning (QML) [1]. However, most QML algorithms do not consider the multidimensionality or data locality of input datasets by converting them into flattened 1-D arrays [9,10]. Nevertheless, quantum computing has shown great potential to outperform traditional, classical computing for specific machine learning tasks [11]. By exploiting quantum parallelism, superposition, and entanglement, quantum computers can accelerate certain computation tasks with exponential speedups. However, in the current era of noisy intermediate-scale quantum (NISQ) devices, the implementation of quantum algorithms is constrained by the number of quantum bits (qubits) and fidelity of quantum gates [12]. For contemporary QML techniques, this problem is addressed by a hybrid approach where only the highly parallel and computationally intensive part of the algorithm is executed in quantum hardware, and the remaining parts are executed using classical computers [13]. Such methods, known as variational quantum algorithms (VQAs), exploit a fixed quantum circuit structure with parameterized rotation gates, denoted as ansatz, whose parameters are optimized using classical backpropagation techniques such as gradient descent [13].

In this work, we propose two generalized techniques for efficient pooling operations in QML, namely, the quantum Haar transform (QHT) for quantum average pooling and partial quantum measurements for two-norm/Euclidean pooling.

We characterize their fidelity to the corresponding classical pooling operations using a state-of-the-art quantum simulator from IBM Quantum for a wide variety of real-world, high-resolution, multidimensional data.

The rest of the paper is organized as follows. Section 2 covers necessary background information, including various quantum operations. Section 3 discusses existing related work. Section 4 introduces our proposed methodology, with great detail given to the constituent parts along with spatial complexity (depth) analysis of the corresponding circuits. Section 5 presents our experimental results, with an explanation of our verification metrics. Finally, Section 6 concludes our work and projects potential future directions.

2. Background

In this section, we provide information about quantum computing (QC), which is essential for understanding the proposed quantum pooling techniques.

2.1. Quantum Bits and States

A quantum bit (qubit) is the most fundamental unit of quantum information. Qubits can be physically realized with a number of hardware technologies, such as photonic chips and superconducting circuits [14]. Mathematically, the quantum wave function or the pure state of a qubit can be represented by a normalized state vector $|\psi\rangle$, in Dirac (Bra-Ket) notation [14], with $N = 2^1$ elements (1).

$$|\psi\rangle = c_0|0\rangle + c_1|1\rangle = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}, \text{ where } |c_0|^2 + |c_1|^2 = 1 \tag{1}$$

For an n-qubit state, the state vector $|\psi\rangle$ grows to a length of $N = 2^n$. As shown in (2), each element $c_j \in \mathbb{C}$ of $|\psi\rangle$ represents the amplitude/coefficient of j^{th} entry in $|\psi\rangle$, or the basis state $|j\rangle$ [14].

$$|\psi\rangle = \sum_{j=0}^{N-1} c_j \cdot |j\rangle = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_j \\ \vdots \\ c_{N-2} \\ c_{N-1} \end{bmatrix}, \text{ where } \sum_{j=0}^{N-1} |c_j|^2 = 1 \text{ , and } 0 \leq j < (N = 2^n) \tag{2}$$

2.2. Quantum Gates

Operations on qubits are called quantum gates and can be represented mathematically by unitary matrix operations. Serial and parallel composite operations can be constructed using matrix multiplications and tensor products, respectively [14,15]. In this section, we will present a number of relevant single- and multi-qubit gates for the proposed quantum pooling techniques.

2.2.1. Hadamard Gate

The Hadamard gate is a single-qubit gate that puts a qubit into superposition; see (3) and Figure 1 [14].

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3}$$

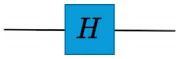


Figure 1. Hadamard gate diagram.

Parallel quantum operations acting on a different set of qubits can be combined using the tensor product [15]. For example, parallel single-qubit Hadamard gates can be represented by a unitary matrix, where each term in the resultant matrix can be directly calculated using the Walsh function [16]; see (4).

$$\begin{gathered} H^{\otimes n} \in SU(2^n) : (H^{\otimes n})_{m,i} = \frac{1}{\sqrt{2^n}} W_m(i), \text{ where} \\ W_m(i) : \mathbb{N} \to \{-1,1\} = \prod_{k=0}^{n-1} (-1)^{\left(\left\lfloor \frac{m}{2^k} \right\rfloor \cdot \left\lfloor \frac{i}{2^k} \right\rfloor\right)}, \text{ and} \\ 0 \leq (m,i) < 2^n \end{gathered} \tag{4}$$

2.2.2. Controlled-NOT (CNOT) Gate

The controlled-NOT, or CNOT, gate is a two-qubit gate, see (5) and Figure 2, that facilitates multi-qubit entanglement [17]. In this work, we will provide a complexity (depth) analysis of the proposed circuits in terms of the critical path of consecutive single-qubit gates and two-qubit CNOT gates [17].

$$\mathbf{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5}$$

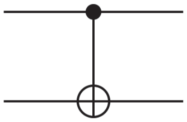


Figure 2. Controlled-NOT gate diagram.

2.2.3. SWAP Gate

The SWAP gate is a two-qubit gate that swaps the positions of the input qubits [14]. Each SWAP operation can be decomposed into three controlled-NOT (CNOT) gates [14], as shown in (6) and Figure 3.

$$\mathbf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

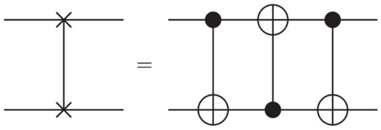
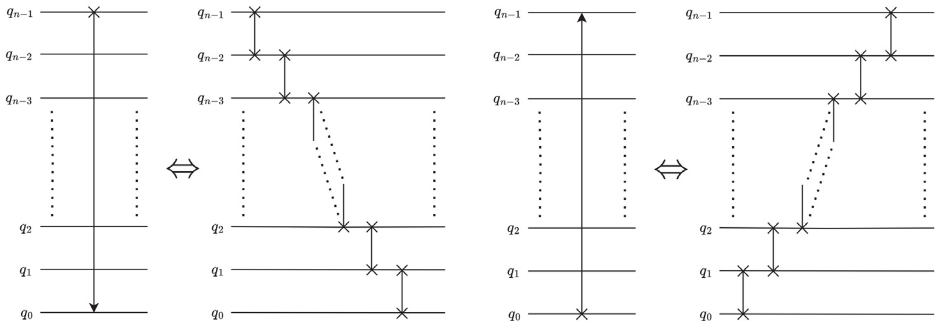


Figure 3. Diagram of swap gate and its decomposition.

2.2.4. Quantum Perfect Shuffle Permutation (PSP)

The quantum perfect shuffle permutation (PSP) is an operation that leverages SWAP gates to perform a cyclical rotation of the input qubits. The quantum PSP rotate-left (RoL) and rotate-right (RoR) operations [6] are shown in Figure 4. Each PSP operation requires $(n-1)$ SWAP operations or, equivalently, $3(n-1)$ CNOT operations; see (6) and Figure 4.



(**a**) $\text{RoL}|q_{n-1}q_{n-2}\ldots q_0\rangle = |q_{n-2}\ldots q_0 q_{n-1}\rangle$ (**b**) $\text{RoR}|q_{n-1}\ldots q_1 q_0\rangle = |q_0 q_{n-1}\ldots q_1\rangle$

Figure 4. Rotate-left and rotate-right operations.

2.2.5. Quantum Measurement

Although colloquially denoted as a measurement "gate", the measurement or observation of a qubit is an irreversible non-unitary operation that projects a qubit's quantum state $|\psi\rangle$ to one of its $|0\rangle$ or $|1\rangle$ basis states [14]. The probability of either basis state being measured is directly determined by the square of the magnitude of each corresponding basis state coefficient, i.e., $p_0 = |c_0|^2$ and $p_1 = |c_1|^2$; see (7) [6]. Here, we use the vector $\psi^{\text{classical}}_{\text{decoded-data}}$, to refer to the measured/decoded output classical data, assuming amplitude encoding [18], resulting from the measurement of the quantum state $|\psi\rangle$; see (8). We also adhere to the Dirac (Bra-Ket) notation [14] to only represent quantum states and wave functions, e.g., $|\psi\rangle$. More specifically, $\psi^{\text{classical}}_{\text{decoded-data}}$, without using the Dirac (Bra-Ket) notation, is a vector that represents the decoded/extracted classical data after measurement, as denoted in Figure 5 by the double-line rail carrying classical bit values [14], rather than a quantum state or a quantum wave function. It is also worth mentioning that $|\psi\rangle$ is assumed to be a pure state, as defined by (2) and (9) [14].

$$P(|\psi\rangle) = \begin{bmatrix} p_0 \\ p_1 \end{bmatrix} = \begin{bmatrix} |c_0|^2 \\ |c_1|^2 \end{bmatrix} \tag{7}$$

$$\psi^{\text{classical}}_{\text{decoded-data}} = \sqrt{P(|\psi\rangle)} = \begin{bmatrix} |c_0| \\ |c_1| \end{bmatrix} \tag{8}$$

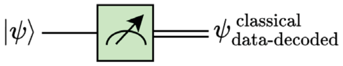


Figure 5. Single-qubit measurement diagram.

In general, an n-qubit quantum state $|\psi\rangle$ has 2^n possible basis states/measurement outcomes, as shown in (9).

$$|\psi\rangle = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_j \\ \vdots \\ c_{N-2} \\ c_{N-1} \end{bmatrix}, \text{where } 0 \leq j < N, \text{and } N = 2^n \tag{9}$$

Accordingly, given the full measurement of a quantum state vector, the probability of finding the qubits in any particular state $|j\rangle$, where $0 \leq j < 2^n$, is given by $\left|c_j\right|^2$ [6]. The overall probability distribution $P(|\psi\rangle)$ can thus be expressed according to (10). Similar to single qubit measurement, the amplitude-encoded [18] output classical data $\psi^{\text{classical}}_{\text{decoded-data}}$ resulting from the measurement of the n-qubit quantum state $|\psi\rangle$ can be calculated from the square root of the probability distribution $\sqrt{P(|\psi\rangle)}$, as shown in (11) and Figure 6. Additionally, $|\psi\rangle$ is assumed to be a pure state [14]. Furthermore, when the encoded classical data are positive real, the amplitudes/coefficients of the corresponding quantum state are accordingly positive real, i.e., $c_j \in \mathbb{Z}$, where $0 \leq j < 2^n$. This results in the amplitudes of $|\psi\rangle$ being numerically equal in values to the coefficients of $\psi^{\text{classical}}_{\text{decoded-data}}$, i.e., $|\psi\rangle = \psi^{\text{classical}}_{\text{decoded-data}}$. In other words, the quantum state $|\psi\rangle$ can be completely determined from the measurement probability distribution such that $|\psi\rangle = \sqrt{P(|\psi\rangle)}$ only when the amplitudes of the quantum state are all of positive real values. It is also worth mentioning that, in order to reconstruct the probability distribution $P(|\psi\rangle)$, it is necessary to repeatedly measure (sample) the quantum state $|\psi\rangle$ to the order of 2^n measurements (samples) where n is the number of qubits. Experimentally, the number of repeated measurements (samples) used for reconstructing the probability distribution $P(|\psi\rangle)$, and, consequently, to decode/extract the amplitude-encoded classical data $\psi^{\text{classical}}_{\text{decoded-data}}$, is usually referred to as the quantum circuit samples or shots [19]. It is recommended to repeat as many measurements and collect as many quantum circuit samples (shots) as possible to minimize the effects of quantum statistical noise [19]. In our experimental work, we used up to 1,000,000 circuit samples/shots to decode/extract the output classical data; see Section 5 for more details.

$$P(|\psi\rangle) = \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_j \\ \vdots \\ p_{N-2} \\ p_{N-1} \end{bmatrix} = \begin{bmatrix} |c_0|^2 \\ |c_1|^2 \\ \vdots \\ |c_j|^2 \\ \vdots \\ |c_{N-2}|^2 \\ |c_{N-1}|^2 \end{bmatrix}, \text{where } p_j = \left|c_j\right|^2, \text{and } 0 \leq j < (N = 2^n) \tag{10}$$

$$\psi^{\text{classical}}_{\text{decoded-data}} = \sqrt{P(|\psi\rangle)} = \begin{bmatrix} |c_0| \\ |c_1| \\ \vdots \\ |c_j| \\ \vdots \\ |c_{N-2}| \\ |c_{N-1}| \end{bmatrix} \tag{11}$$

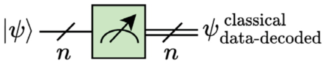


Figure 6. Multi-qubit measurement diagram.

3. Related Work

Convolutional neural networks (CNNs) [20] represent a specialized type of neural network that consists of convolutional layers, pooling layers, and fully connected layers. The convolutional layer extracts characteristic features from an image, while the pooling layer down-scales the extracted features to a smaller data size by considering specific segments of data, often referred to as "windows" [1,20]. Pooling also enhances the network's robustness to input translations and helps prevent overfitting [1]. For classical implementations on GPUs, pooling is usually limited to 3-D data [21,22], with a time complexity of $\mathcal{O}(N)$ [23], where N is the data size.

Quantum convolutional neural networks (QCNNs), as proposed by [10], explored the feasibility of extending the primary features and structures of conventional CNNs to quantum computers. However, translating the entire CNN model onto the presently available noisy intermediate-scale quantum (NISQ) devices is not practical due to limited qubit count [18], low decoherence time [24], and high gate errors [25]. To attain quantum advantage amid the constraints of NISQ devices, it is essential to develop depth-optimized convolution and pooling circuits that generate high-fidelity outputs. Most implementations of quantum pooling [9,12,26–28] leverage parameterized quantum circuits (PQCs) and mid-circuit measurement as originally proposed in Ref. [10]. These techniques, however, do not perform the classical pooling operation as used in CNNs and thus do not gain the associated benefits from exploiting data locality. Moreover, PQC-based implementations of pooling increase the number of training parameters, which makes the classical optimization step more computationally intensive. The authors in Ref. [29] implemented quantum pooling by omitting measurement gates on a subset of qubits. However, the authors do not generalize their technique for varying window sizes, levels of decomposition, or data dimensions.

In this work, we propose a quantum average pooling technique based on the quantum Haar transform (QHT) [25] and a quantum Euclidean pooling technique utilizing partial measurement of quantum circuits. These techniques are generalizable for arbitrary window size and arbitrary data dimension. We have also provided the generalizable circuits for both techniques. The proposed methods have been validated with respect to their classical-implementation counterparts, and the quality of their results has been demonstrated by reporting the metric of quantum fidelity.

4. Materials and Methods

In this section, we discuss the proposed quantum pooling methods. Pooling, or downsampling, is a critical component of CNNs that consolidates similar features into one [2]. The most commonly used pooling schemes are average and maximum (max) pooling [30], where the two differ in terms of the sharpness of the defined input features. Max pooling typically offers a sharper definition of input features while average pooling offers a smoother definition of input features [30]. Depending on the desired application or dataset, one pooling technique may be preferable over the other [30].

Average and max pooling can be represented as special cases of calculating the p-norm or ℓ^p norm [31], where the p-norm of a vector $\mathbf{x} \in \mathbb{C}^N$ of size N elements is given by (12) for $p \in \mathbb{Z}$ [31]. More specifically, average and max pooling can be defined as the 1-norm and ∞-norm, respectively; see (13) and (14). Since max pooling ($p = \infty$) is difficult to implement as a quantum (unitary) operation, a pooling scheme defined by the p-norm where $1 < p < \infty$ could establish a balance between the average and max pooling schemes. Therefore, we introduce an intermediate pooling technique based on the 2-norm/Euclidean norm named as the quantum Euclidean pooling technique; see (15). The proposed Euclidean pooling technique is limited to processing positive real data and is only compatible with the 2-norm, as norms for $p > 2$ suffer from some similar challenges to max pooling in terms of being non-unitary and thus unwieldy to implement in a quantum context.

$$\|\mathbf{x}\|_p = \left(\sum_{i=0}^{N-1} x_i^p \right)^{\frac{1}{p}} \tag{12}$$

$$\bar{\mathbf{x}} = \frac{1}{N}\|\mathbf{x}\|_1 \tag{13}$$

$$\max(\mathbf{x}) = \|\mathbf{x}\|_\infty \tag{14}$$

$$\epsilon(\mathbf{x}) = \frac{1}{\sqrt{N}}\|\mathbf{x}\|_2 \tag{15}$$

In this work, we propose quantum pooling techniques for average pooling ($p = 1$) and Euclidean pooling ($p = 2$). We implement average pooling using the QHT, a highly parallelizable quantum algorithm for performing multilevel decomposition/reduction in multidimensional data. For the implementation of Euclidean pooling, we employ partial measurement to perform dimension reduction with zero circuit depth. The average and Euclidean pooling techniques are described in greater detail in Sections 4.1 and 4.2, respectively. As detailed further in Section 5, we validated our proposed quantum pooling techniques using 1-D audio data [32], 2-D black-and-white (B/W) images [33], 3-D color (RGB) images [33], and 3-D hyperspectral images [34].

4.1. Quantum Average Pooling via Quantum Haar Transform

Our first proposed quantum pooling technique implements average pooling on quantum devices using the quantum wavelet transform (QWT). A wavelet transform decomposes the input data into low- and high-frequency components, where in the case of pooling, the low-frequency components represent the desired downsampled data [6]. For our proposed technique, we leverage the quantum variant of the first and simplest wavelet transform, the quantum Haar transform (QHT) [6]. The execution of the pooling operation using the QHT involves two main steps:

- Haar Wavelet Operation: By applying Hadamard (H) gates (see Section 2.2.1) in parallel, the high- and low-frequency components are decomposed from the input data.
- Data Rearrangement: By applying quantum rotate-right (RoR) operations (see Section 2.2.4), the high- and low-frequency components are grouped into contiguous regions.

We outline (in order) the following sections as follows, the quantum circuits and corresponding circuit depths of a single-level decomposition, the 1-D QHT, the ℓ-level 1-D QHT, and the ℓ-level d-D QHT, respectively, where ℓ is the number of decomposition levels and d is the dimensionality of the QHT operation.

4.1.1. Single-Level One-Dimensional Quantum Haar Transform

For the single-level one-dimensional (1-D) QHT, we will assume a 1-D input data of size N data points. The aforementioned data would be encoded into the quantum circuit using amplitude encoding [18] as an n-qubit quantum state $|\psi\rangle$, where $n = \lceil \log_2 N \rceil$.

The quantum circuit for single-level decomposition of the 1-D QHT is shown in Figure 7, where $|\psi_H\rangle$ represents the quantum state after the wavelet operation and $|\psi_{\text{out}}\rangle$ represents the quantum state after data rearrangement.

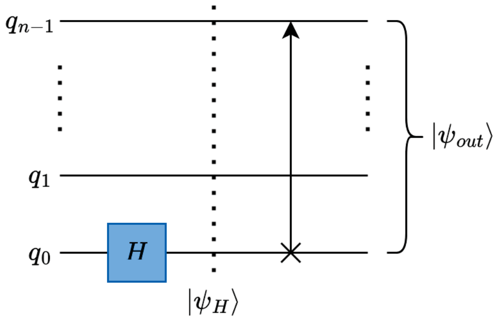


Figure 7. Single-level 1-D QHT circuit.

Haar Wavelet Operation on Single-Level One-Dimensional Data

The Haar wavelet is performed using the H gate; see Figure 7. For example, a single-level decomposition of the 1-D QHT can be performed on a state vector $|\psi\rangle$ as described by (2) by applying a single H gate to the least-significant qubit of $|\psi\rangle$, designated q_0 in Figure 7. This operation replaces the data value of the pairs $(c_{2j}, c_{2j+1}) : 0 \le j < \frac{N}{2}$ with their sum and difference, as shown in (16). It is worth mentioning that the sum terms represent the low-frequency terms and the difference terms represent the high-frequency terms for a single level of decomposition.

$$|\psi_H\rangle = \left(I^{\otimes n-1} \otimes H\right)|\psi\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} c_0 + c_1 \\ c_0 - c_1 \\ \vdots \\ c_{2j} + c_{2j+1} \\ c_{2j} - c_{2j+1} \\ \vdots \\ c_{N-2} + c_{N-1} \\ c_{N-2} - c_{N-1} \end{bmatrix}, \text{where } 0 \le j < \frac{N}{2} \tag{16}$$

Data Rearrangement Operation

The data rearrangement operation congregates the low- or high-frequency fragmented terms after decomposition. For instance, the low- and high-frequency terms are segregated after wavelet decomposition, as expressed in (16). The low- and high-frequency terms exist at the even indices $(|q_{n-1} \dots q_1\rangle|q_0 = 0\rangle)$ and odd indices $(|q_{n-1} \dots q_1\rangle|q_0 = 1\rangle)$, respectively. Ideally, the state vector is formed in a way such that the low-frequency terms should be merged into a contiguous half of the overall state vector $(|q_0 = 0\rangle|q_{n-1} \dots q_1\rangle)$, while the rest of the state vector consists of the high-frequency terms $(|q_0 = 1\rangle|q_{n-1} \dots q_1\rangle)$. This data rearrangement operation can be performed using the qubit rotation $|q_{n-1} \dots q_1 q_0\rangle \Rightarrow |q_0 q_{n-1} \dots q_1\rangle$ using a RoR operation; see Figure 7 and (17).

$$|\psi_{\text{out}}\rangle = \text{rotate-right (RoR)}|\psi_H\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} c_0 + c_1 \\ \vdots \\ c_{2j} + c_{2j+1} \\ \vdots \\ c_{N-2} + c_{N-1} \\ c_0 - c_1 \\ \vdots \\ c_{2j} - c_{2j+1} \\ \vdots \\ c_{N-2} - c_{N-1} \end{bmatrix} \begin{array}{l} \left.\begin{array}{c} \uparrow \\ \\ \\ \\ \downarrow \end{array}\right. 2^{n-1} \\ \left.\begin{array}{c} \uparrow \\ \\ \\ \\ \downarrow \end{array}\right. 2^{n-1} \end{array} \left.\begin{array}{c} \uparrow \\ \\ \\ \\ \\ \\ \\ \\ \\ \downarrow \end{array}\right. 2^n, \text{where } 0 \le j < \frac{N}{2} \tag{17}$$

Circuit Depth

The depth of the single-level 1-D QHT operation can be considered in terms of 1 H gate and 1 perfect-shuffle (RoR) gate. An RoR gate can be decomposed into $(n-1)$ SWAP gates or $3(n-1)$ controlled-NOT (CNOT) gates. Accordingly, the total circuit depth can be expressed in terms of the number of consecutive single-qubit and CNOT gates, as shown in (18).

$$\begin{aligned}\Delta_{\text{1-D QHT}}(n,\ell=1) &= \Delta_{\text{H}} + \Delta_{\text{RoR}}(n) = 1 + 3(n-1) = 3n-2 \\ &= \mathcal{O}(n)\end{aligned} \tag{18}$$

In many common quantum computing libraries, including `Qiskit` [19], it is possible to leverage arbitrary mapping of quantum registers to classical registers [35] to perform data rearrangement during quantum-to-classical (Q2C) data decoding without increasing circuit depth. Accordingly, the circuit depth of the optimized single-level 1-D QHT circuit can be expressed as shown in (19).

$$\begin{aligned}\Delta^{\text{opt}}_{\text{1-D QHT}}(n,\ell=1) &= \Delta_{\text{H}} = 1 \\ &= \mathcal{O}(1)\end{aligned} \tag{19}$$

4.1.2. Multilevel One-Dimensional Quantum Haar Transform

In this section, we discuss how multiple levels (ℓ) of decomposition can be applied to further reduce the final data size. Given the initial data are set up in the same manner as in the single-level variant, the final data size can be expressed as $\left\lceil \frac{N}{2^\ell} \right\rceil$. The corresponding quantum circuit for the multilevel 1-D QHT is shown in Figure 8. For the interested reader, more details about the multilevel 1-D QHT can be found in [6].

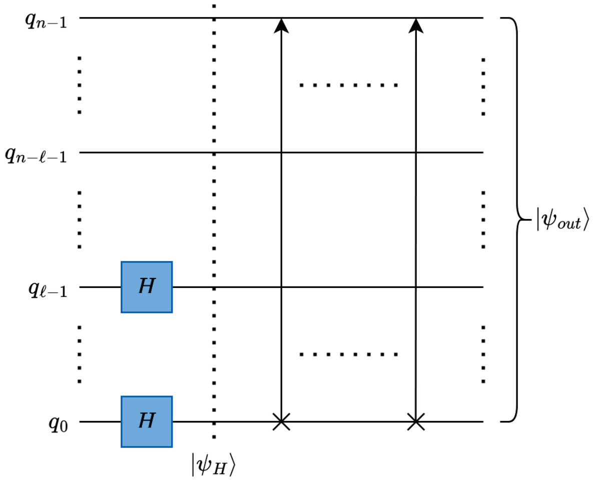


Figure 8. Multilevel One-dimensional (1-D) QHT circuit.

Haar Wavelet Operation on Multilevel One-Dimensional Data

To perform multiple levels of decomposition, additional Hadamard gates are applied on the ℓ least-significant qubits of $|\psi\rangle$, as shown in Figure 8. The multilevel 1-D wavelet operation divides $|\psi\rangle$ into $2^{n-\ell}$ groups of 2^ℓ terms and replaces them with the appropriately decomposed values according to the Walsh function [16]; see (20).

$$\begin{aligned} U_{\text{Haar}}^{\text{1-D}}|\psi\rangle &= \left(I^{\otimes n-\ell} \otimes H^{\otimes \ell}\right)|\psi\rangle = \frac{1}{\sqrt{2^{\ell}}}\begin{bmatrix} |\phi_0\rangle \\ \vdots \\ |\phi_j\rangle \\ \vdots \\ |\phi_{2^{n-\ell}-1}\rangle \end{bmatrix}, \text{where} \\ |\phi_j\rangle &= \sum_{m=0}^{\ell-1}\sum_{i=0}^{\ell-1} W_m(i)c_{\ell j+i}|\ell j+i\rangle = \begin{bmatrix} \sum_{i=0}^{\ell-1} W_0(i)c_{\ell j+i} \\ \vdots \\ \sum_{i=0}^{\ell-1} W_m(i)c_{\ell j+i} \\ \vdots \\ \sum_{i=0}^{\ell-1} W_{\ell-1}(i)c_{\ell j+i} \end{bmatrix}, \\ 0 \le j &< \left\lceil \frac{N_i}{2^{\ell_i}} \right\rceil \text{ and } W_m(i) = \prod_{k=0}^{n-1}(-1)^{\left(\left\lfloor \frac{i}{2^k}\right\rfloor \cdot \left\lfloor \frac{j}{2^k}\right\rfloor\right)} \end{aligned} \tag{20}$$

Data Rearrangement Operation

Multiple levels of 1-D decomposition can be implemented using ℓ serialized RoR operations; see (21) and Figure 8. However, parallelization of the data rearrangement operation across multiple levels of decomposition can be achieved by overlapping/interleaving the RoR operations into SWAP gates and fundamental two-qubit gates.

$$U_{\text{rearrangement}}^{\text{1-D}} = \prod_{j=\ell-1}^{0} \text{RoR}(n) \tag{21}$$

Circuit Depth

The inherent parallelizability of the wavelet and data rearrangement steps of the QHT can be used to reduce the circuit depth. In the wavelet step, all ℓ levels of decomposition can be performed by ℓ parallel Hadamard gates ($H^{\otimes \ell}$). In the data rearrangement step, the decomposition and interleaving of RoR operations can be used to reduce the depth penalty incurred by multilevel decomposition to just 2 SWAP gates, or 6 CNOT gates, per decomposition level; see (22).

$$\begin{aligned} \Delta_{\text{1-D QHT}}(n,\ell) &= \Delta_{\text{H}} + \Delta_{\text{RoR}}(n) + 3(2(\ell-1)) \\ &= 1 + 3(n-1) + 3(2(\ell-1)) = 3(n+2\ell) - 8 \\ &= \mathcal{O}(n+\ell) \end{aligned} \tag{22}$$

Additionally, if the deferral of data rearrangement is permitted, the circuit depth of the multilevel 1-D QHT can be shown to be constant (requiring just 1 Hadamard gate of depth); see (23).

$$\begin{aligned} \Delta_{\text{1-D QHT}}^{\text{opt}}(n,\ell) &= \Delta_{\text{H}} = 1 \\ &= \mathcal{O}(1) \end{aligned} \tag{23}$$

4.1.3. Multilevel Multidimensional Quantum Haar Transform

For the multidimensional QHT, we can assume the input data are d-dimensional, where each dimension has a data size of $N_i : 0 \le i < d$ for a total data size of $N = \prod_{i=0}^{d-1} N_i$. We can denote the largest dimension of data as $N_{\max} = \max_{i=0}^{d-1} N_i$, where it is encoded by $n_{\max}$ qubits. Similarly, the smallest dimension of data is denoted as $N_{\min} = \min_{i=0}^{d-1} n_i$ and is encoded by $n_{\min}$ qubits. Similar to the 1-D case, the data are encoded as an n-qubit quantum state $|\psi\rangle$, such that each dimension i of data requires $n_i = \lceil \log_2 N_i \rceil$ qubits and ℓ_i decomposition levels. It is worth mentioning that the total number of required qubits is $n = \sum_{i=0}^{d-1} n_i$ qubits and the final size of each data dimension i is $\left\lceil \frac{N_i}{2^{\ell_i}} \right\rceil$.

Based on the nature of the encoding scheme (amplitude encoding) and quantum circuit structures, the multidimensional QHT can be performed by parallel application of d 1-D QHT circuits. Thus, the transformation of each data dimension can be performed independently of the other data dimensions. More specifically, using a column-major vectorization of the multidimensional data, the i^{th} dimension of the data is represented by the contiguous region of qubits $q_{\sum_{j=0}^{i-1} n_j}$ to $q_{\sum_{j=0}^{i} n_j - 1}$. In other words, the multidimensional d-D QHT can be performed by stacking d 1-D QHT circuits in parallel, each of which performs the transformation on the respective contiguous region/data dimension, as shown in Figure 9. For the interested reader, more details about the multilevel multidimensional (d-D) QHT can be found in [6].

Haar Wavelet Operation on Multidimensional Data

Exploiting the parallelization offered by stacking, the multilevel d-D QHT wavelet operation can also be performed with constant circuit depth; see (24) and Figure 9.

$$U_{\text{Haar}}^{d\text{-D}} = \bigotimes_{i=d-1}^{0} \left(I^{\otimes n_i - \ell_i} \otimes H^{\otimes \ell_i} \right) \tag{24}$$

Data Rearrangement Operation

The multilevel d-D QHT data rearrangement operation is given by (25) and shown in Figure 9.

$$U_{\text{rearrangement}}^{d\text{-D}} = \bigotimes_{i=d-1}^{0} \prod_{j=\ell_i - 1}^{0} \text{RoR}(n_i) \tag{25}$$

Circuit Depth

Since the multidimensional QHT can be parallelized across dimensions, the circuit depth is determined by the dimension with the largest total data size and number of decomposition levels, as shown in (26).

$$\begin{aligned} \Delta_{d\text{-D QHT}}(n, \ell) &= \max_{i=0}^{d-1} (\Delta_{\text{H}} + \Delta_{\text{RoR}}(n_i) + 3(2(\ell_i - 1))) \\ &= 1 + 3(n_{\max} - 1) + 3(2(\ell_{\max} - 1)) = 3(n_{\max} + 2\ell_{\max}) - 8 \\ &= \mathcal{O}(n_{\max} + \ell_{\max}) \end{aligned} \tag{26}$$

If data rearrangement can be performed in the classical post-processing of Q2C data decoding, as discussed in Section 4.1.1, the wavelet operation is completely parallelized for the multilevel multidimensional QHT, resulting in an optimal, constant circuit depth; see (27).

$$\begin{aligned} \Delta_{d\text{-D QHT}}^{\text{opt}} &= \max_{i=0}^{d-1} (\Delta_{\text{H}}) = 1 \\ &= \mathcal{O}(1) \end{aligned} \tag{27}$$

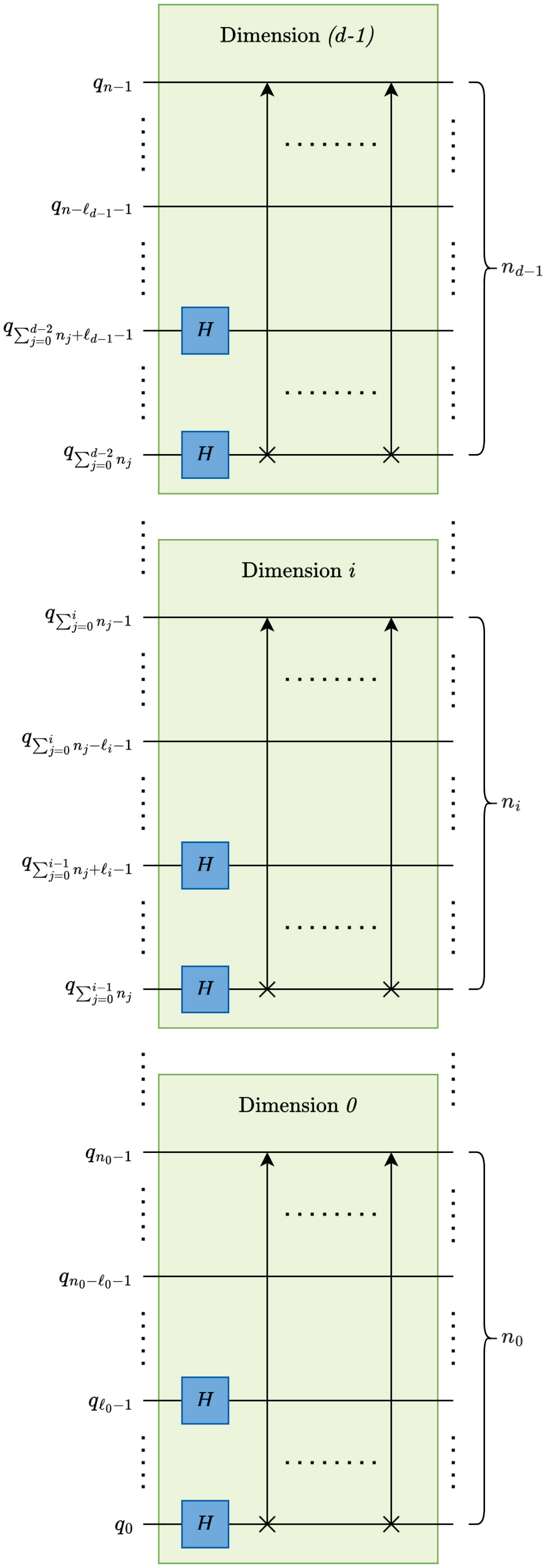

Dimension *(d-1)*
q_{n-1}
$q_{n-\ell_{d-1}-1}$
$q_{\sum_{j=0}^{d-2} n_j+\ell_{d-1}-1}$
$q_{\sum_{j=0}^{d-2} n_j}$
H
H
n_{d-1}
Dimension i
$q_{\sum_{j=0}^{i} n_j-1}$
$q_{\sum_{j=0}^{i} n_j-\ell_i-1}$
$q_{\sum_{j=0}^{i-1} n_j+\ell_i-1}$
$q_{\sum_{j=0}^{i-1} n_j}$
H
H
n_i
Dimension *0*
q_{n_0-1}
$q_{n_0-\ell_0-1}$
q_{ℓ_0-1}
q_0
H
H
n_0

Figure 9. Multilevel Multidimensional (*d*-D) QHT circuit.

4.2. Quantum Euclidean Pooling Using Partial Measurement

Our second proposed quantum pooling technique applies the 2-norm or Euclidean norm over a given window of positive real data. We implement the proposed Euclidean pooling technique using partial quantum measurement, which can be expressed mathematically either using conditional probabilities or partial traces of the density operator/matrix [14,15].

As discussed in Section 2.2.5 and expressed in (10), full measurement of an n-qubit quantum state has 2^n possible outcomes, one for each basis state, where the probability of each outcome can be derived from the corresponding state vector $|\psi\rangle$. A subset of m qubits would only have 2^m possible outcomes, where $m < n$. Thus, the probability distribution of the partial measurement can be derived from the probability distribution of the full measurement using conditional probability, where each qubit of the unmeasured qubits could arbitrarily be in either a $|0\rangle$ or $|1\rangle$ state [6]. For example, if the least-significant qubit q_0 is excluded from the measurements of the quantum state $|\psi\rangle$, the conditional probability distribution $P(|\psi\rangle|q_0)$ for the partial measurements can be derived as shown in (28). Accordingly, the decoded output classical data $\psi^{\text{classical}}_{\text{decoded-data}}$ resulting from the partial measurement of the n-qubit quantum state $|\psi\rangle$ can be experimentally calculated from the square root of the conditional probability distribution $\sqrt{P(|\psi\rangle \mid q_0)}$, as shown in (11). Here, we would like to mention, as discussed in Section 2.2.5, that the quantum state $|\psi\rangle$ is assumed to be a pure state [14], amplitude encoding [18] is used, and the encoded classical data are of positive real values; please refer to Section 2.2.5 for more details.

$$\begin{aligned} P(|\psi\rangle \mid q_0) &\equiv P(|\psi\rangle \mid q_0 = 0 \text{ or } q_0 = 1) \equiv P(|\psi\rangle \mid q_0 = 0) + P(|\psi\rangle \mid q_0 = 1) \\ &= \begin{bmatrix} p_{0|q_0} \\ \vdots \\ p_{j|q_0} \\ \vdots \\ p_{\frac{N}{2}-1|q_0} \end{bmatrix} = \begin{bmatrix} |c_0|^2 + |c_1|^2 \\ \vdots \\ |c_{2j}|^2 + |c_{2j+1}|^2 \\ \vdots \\ |c_{N-2}|^2 + |c_{N-1}|^2 \end{bmatrix}, \text{where} \\ P(|\psi\rangle = j \mid q_0) &= p_{j|q_0} = |c_{2j}|^2 + |c_{2j+1}|^2, \text{and } 0 \le j < \left(\frac{N}{2} = 2^{(n-1)}\right) \end{aligned} \tag{28}$$

In general, the probability distribution $P(|\psi\rangle \mid q_{\ell-1}, \cdots, q_0)$ of a partial measurement of the most significant $(n-\ell)$ qubits out of n qubits, where ℓ is the number of unmeasured least significant qubits, can be derived from the diagonal elements of the reduced density operator/matrix $\rho_{(n-\ell)}$ (partial trace of the density operator/matrix ρ_n) [14,15]. More specifically, $P(|\psi\rangle \mid q_{\ell-1}, \cdots, q_0)$ can be calculated using the partial trace, as shown in (29)–(32).

$$\begin{aligned} \rho_n = |\psi\rangle\langle\psi| &= \left(\sum_{i=0}^{(2^n-1)} c_i |i\rangle\right)\left(\sum_{j=0}^{(2^n-1)} c_j^* \langle j|\right) = \sum_{i=0}^{(2^n-1)}\sum_{j=0}^{(2^n-1)} (c_i \cdot c_j^*)|i\rangle\langle j| \\ &= \sum_{i=0}^{(2^n-1)}\sum_{j=0}^{(2^n-1)} \rho_n(i,j)|i\rangle\langle j| = \sum_{i=0}^{(2^n-1)}\sum_{j=0}^{(2^n-1)} (\langle i|\rho_n|j\rangle)|i\rangle\langle j| \\ &\text{where } \rho_n(i,j) = \langle i|\rho_n|j\rangle = c_i \cdot c_j^* \end{aligned} \tag{29}$$

$$\begin{aligned}
\rho_{(n-\ell)} &= \mathrm{tr}_{(q_{\ell-1},\cdots,q_0)}(\rho_n) \\
&= \sum_{i=0}^{(2^{(n-\ell)}-1)} \sum_{j=0}^{(2^{(n-\ell)}-1)} \rho_{(n-\ell)}(i,j)|i\rangle\langle j| = \sum_{i=0}^{(2^{(n-\ell)}-1)} \sum_{j=0}^{(2^{(n-\ell)}-1)} \Big(\langle i|\rho_{(n-\ell)}|j\rangle\Big)|i\rangle\langle j| \\
&= \sum_{i=0}^{(2^{(n-\ell)}-1)} \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} \rho_n(i\cdot 2^{\ell}+k, j\cdot 2^{\ell}+k)|i\rangle\langle j| \\
&= \sum_{i=0}^{(2^{(n-\ell)}-1)} \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} \Big\langle i\cdot 2^{\ell}+k\Big|\rho_n\Big|j\cdot 2^{\ell}+k\Big\rangle|i\rangle\langle j| \\
&= \sum_{i=0}^{(2^{(n-\ell)}-1)} \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} c_{(i\cdot 2^{\ell}+k)}\cdot c^*_{(j\cdot 2^{\ell}+k)}|i\rangle\langle j|
\end{aligned} \tag{30}$$

where $\rho_{(n-1)}(i,j) = \langle i|\rho_{(n-1)}|j\rangle = \sum_{k=0}^{(2^{\ell}-1)} \rho_n(i\cdot 2^{\ell}+k, j\cdot 2^{\ell}+k)$, and

$\rho_n(i\cdot 2^{\ell}+k, j\cdot 2^{\ell}+k) = \Big\langle i\cdot 2^{\ell}+k\Big|\rho_n\Big|j\cdot 2^{\ell}+k\Big\rangle = c_{(i\cdot 2^{\ell}+k)}\cdot c^*_{(j\cdot 2^{\ell}+k)}$

$$\begin{aligned}
P(|\psi\rangle \mid q_{\ell-1},\cdots,q_0) &= \sum_{j=0}^{(2^{(n-\ell)}-1)} p_{j\mid(q_{\ell-1},\ldots,q_0)}|j\rangle = \sum_{j=0}^{(2^{(n-\ell)}-1)} \Big(\langle j|\rho_{(n-1)}|j\rangle\Big)|j\rangle \\
&= \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} \Big\langle j\cdot 2^{\ell}+k\Big|\rho_n\Big|j\cdot 2^{\ell}+k\Big\rangle|j\rangle \\
&= \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} c_{(j\cdot 2^{\ell}+k)}\cdot c^*_{(j\cdot 2^{\ell}+k)}|j\rangle
\end{aligned} \tag{31}$$

$$\to P(|\psi\rangle \mid q_{\ell-1},\cdots,q_0) = \sum_{j=0}^{(2^{(n-\ell)}-1)} \sum_{k=0}^{(2^{\ell}-1)} \Big|c_{(j\cdot 2^{\ell}+k)}\Big|^2|j\rangle$$

$$
\begin{aligned}
P(|\psi\rangle \mid q_{\ell-1}, \cdots, q_0) &= \begin{bmatrix} p_{0 \mid (q_{\ell-1},\ldots,q_0)} \\ \vdots \\ p_{j \mid (q_{\ell-1},\ldots,q_0)} \\ \vdots \\ p_{\left(2^{(n-\ell)}-1\right) \mid (q_{\ell-1},\ldots,q_0)} \end{bmatrix} = \begin{bmatrix} \langle 0|\rho_{(n-\ell)}|0\rangle \\ \vdots \\ \langle j|\rho_{(n-\ell)}|j\rangle \\ \vdots \\ \left\langle 2^{(n-\ell)}-1\middle|\rho_{(n-\ell)}\middle|2^{(n-\ell)}-1\right\rangle \end{bmatrix} \\
&= \begin{bmatrix} \sum_{k=0}^{(2^\ell-1)} \langle k|\rho_n|k\rangle \\ \vdots \\ \sum_{k=0}^{(2^\ell-1)} \left\langle j\cdot 2^\ell + k\middle|\rho_n\middle|j\cdot 2^\ell + k\right\rangle \\ \vdots \\ \sum_{k=0}^{(2^\ell-1)} \left\langle 2^n - 2^\ell + k\middle|\rho_n\middle|2^n - 2^\ell + k\right\rangle \end{bmatrix} \\
\rightarrow P(|\psi\rangle \mid q_{\ell-1}, \cdots, q_0) &= \begin{bmatrix} \sum_{k=0}^{(2^\ell-1)} |c_k|^2 \\ \vdots \\ \sum_{k=0}^{(2^\ell-1)} \left|c_{(j\cdot 2^\ell+k)}\right|^2 \\ \vdots \\ \sum_{k=0}^{(2^\ell-1)} \left|c_{(2^n-2^\ell+k)}\right|^2 \end{bmatrix} \Bigg\updownarrow 2^{(n-\ell)}, \text{ where } 0 \le j < 2^{(n-\ell)}
\end{aligned}
\tag{32}
$$

4.2.1. Single-Level One-Dimensional Quantum Euclidean Pooling

For single-level one-dimensional Euclidean pooling, we will assume the input data $\mathbf{x}$ is 1-D with a data size of N, in terms of the number of data points. The input data are encoded using amplitude encoding [18] as an n-qubit quantum state $|\psi\rangle$, where $n = \lceil \log_2 N \rceil$.

$$
\begin{aligned}
\psi^{\text{classical}}_{\text{decoded-data}} &= \sqrt{P(|\psi\rangle \mid q_0)} \\
&= \begin{bmatrix} \sqrt{|c_0|^2 + |c_1|^2} \\ \vdots \\ \sqrt{\left|c_{2j}\right|^2 + \left|c_{2j+1}\right|^2} \\ \vdots \\ \sqrt{\left|c_{2^n-2}\right|^2 + \left|c_{2^n-1}\right|^2} \end{bmatrix} \Bigg\updownarrow 2^{n-1}, \text{ where } 0 \le j < 2^{n-1}
\end{aligned}
\tag{33}
$$

After applying one level of 1-D Euclidean pooling to the positive real quantum state $|\psi\rangle$, the resultant classical state $\psi^{\text{classical}}_{\text{decoded-data}}$ can be expressed as shown in (33) as derived from (32) when $\ell = 1$. As discussed previously, it is possible to extract this partial quantum state using partial measurement, as shown in (28), as long as the quantum state encodes positive real data; see Section 2.2.5. The corresponding quantum circuit for the single-level 1-D Euclidean pooling operation is presented in Figure 10.

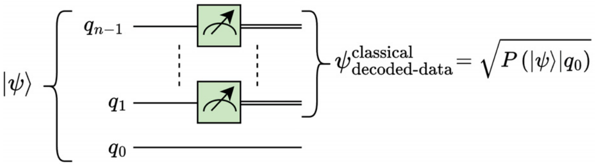

$|\psi\rangle$
q_{n-1}
q_1
q_0
$\psi^{\text{classical}}_{\text{decoded-data}} = \sqrt{P\left(|\psi\rangle|q_0\right)}$

Figure 10. Single-level 1-D Euclidean pooling circuit.

4.2.2. Multilevel One-Dimensional Quantum Euclidean Pooling

In multilevel 1-D decomposition on positive real data, as shown in Figure 11, the Euclidean norm (2-norm) of $|\psi\rangle$ is taken with a window of 2^{ℓ}, where the number of decomposition levels is ℓ; see (34) and Figure 11.

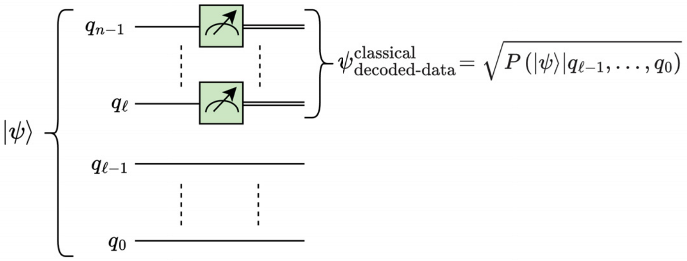

$|\psi\rangle$
q_{n-1}
q_ℓ
$q_{\ell-1}$
q_0
$\psi^{\text{classical}}_{\text{decoded-data}} = \sqrt{P\left(|\psi\rangle|q_{\ell-1},\ldots,q_0\right)}$

Figure 11. Multilevel 1-D Euclidean pooling circuit.

$$\begin{aligned} \psi^{\text{classical}}_{\text{decoded-data}} &= \sqrt{P(|\psi\rangle \mid q_{\ell-1},\cdots,q_0)} \\ &= \begin{bmatrix} \sqrt{\sum_{k=0}^{2^{\ell}-1} |c_k|^2} \\ \vdots \\ \sqrt{\sum_{k=0}^{2^{\ell}-1} \left|c_{(2^{\ell}\cdot j+k)}\right|^2} \\ \vdots \\ \sqrt{\sum_{k=0}^{2^{\ell}-1} \left|c_{(2^{n}-2^{\ell}+k)}\right|^2} \end{bmatrix} \updownarrow 2^{(n-\ell)}, \text{ where } 0 \le j < 2^{(n-\ell)} \end{aligned} \tag{34}$$

The change in normalization for 1-D pooling can be generalized with the corresponding increase in window size for the Euclidean norm, to $\frac{1}{\sqrt{2^{\ell}}}$, as shown in (35).

$$\mathbf{x}_{\text{out}} = \frac{1}{\sqrt{2^{\ell}}} \cdot \psi^{\text{classical}}_{\text{decoded-data}} \tag{35}$$

4.2.3. Multilevel Multidimensional Quantum Euclidean Pooling

The multilevel d-dimensional quantum Euclidean pooling circuit is illustrated in Figure 12, where ℓ_i is the number of decomposition levels for dimension i and $0 \le i < d$. Similar to the multilevel multidimensional QHT circuit discussed in Section 4.1.3, parallelization can be also applied to Euclidean pooling across dimensions using a stacked quantum circuit.

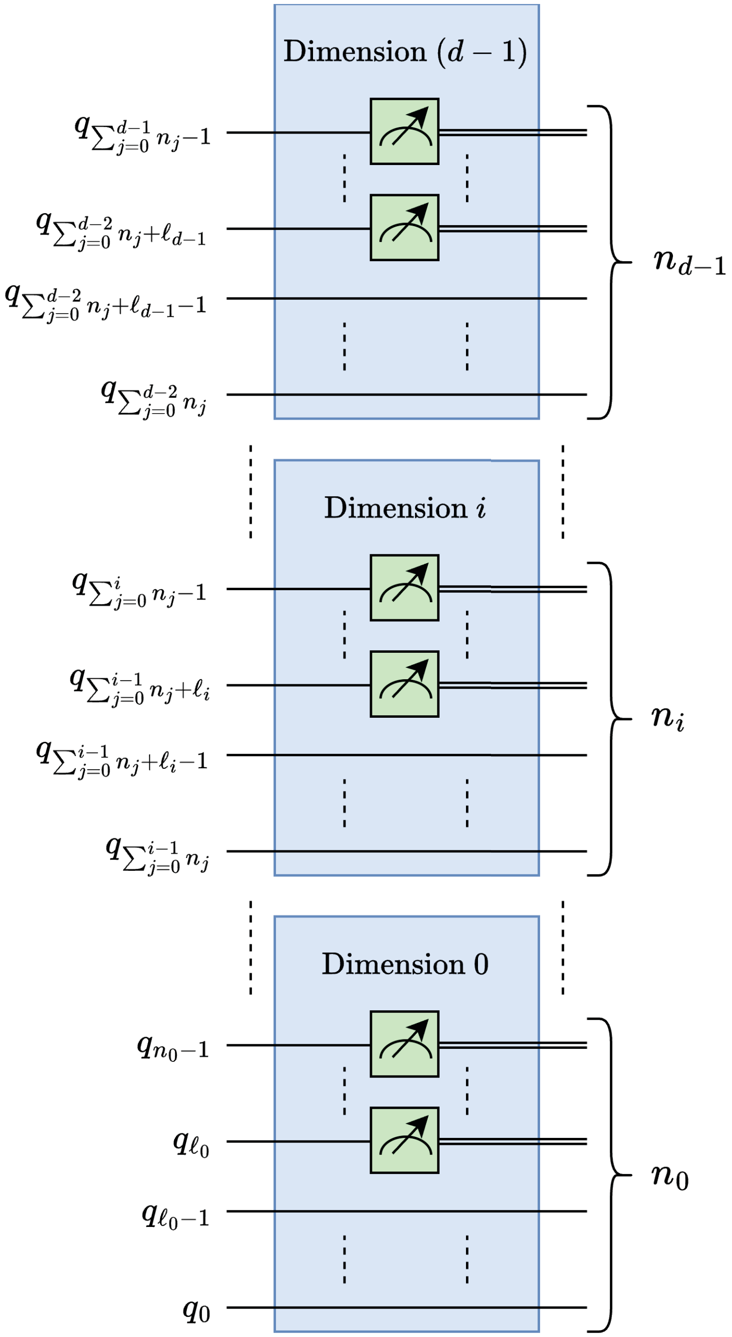

Dimension $(d-1)$
$q_{\sum_{j=0}^{d-1} n_j - 1}$
$q_{\sum_{j=0}^{d-2} n_j + \ell_{d-1}}$
$q_{\sum_{j=0}^{d-2} n_j + \ell_{d-1} - 1}$
$q_{\sum_{j=0}^{d-2} n_j}$
n_{d-1}
Dimension i
$q_{\sum_{j=0}^{i} n_j - 1}$
$q_{\sum_{j=0}^{i-1} n_j + \ell_i}$
$q_{\sum_{j=0}^{i-1} n_j + \ell_i - 1}$
$q_{\sum_{j=0}^{i-1} n_j}$
n_i
Dimension 0
$q_{n_0 - 1}$
q_{ℓ_0}
$q_{\ell_0 - 1}$
q_0
n_0

Figure 12. Multilevel decomposition d-dimensional Euclidean pooling circuit.

5. Experimental Work

In this section, we discuss our experimental setup and results. Experiments were conducted using real-world, high-resolution data, using both the quantum average and Euclidean pooling techniques. Section 5.1 delves into further detail on the experimental setup while Section 5.2 analyzes the obtained results.

5.1. Experimental Setup

The efficacy of the two proposed pooling methods was examined through tests using real-world, high-resolution data of varying dimensions and data sizes. One-dimensional pooling was performed on selected publicly available sound quality assessment material published by the European Broadcasting Union, which was pre-processed into a single channel with data sizes ranging from 2^8 data points to 2^{20} data points when sampled at 44.1 kHz [32]. Two-dimensional pooling was evaluated on black-and-white (B/W) and color (RGB) images of Jayhawks [33], as shown in Figure 13, sized from (8×8) pixels to $(512 \times 512 \times 3)$ pixels. Additionally, 3-D pooling was performed on hyperspectral images from the Kennedy Space Center (KSC) dataset (see [34]) after pre-processing and resizing, with sizes ranging from $(8 \times 8 \times 8)$ pixels to $(128 \times 128 \times 128)$ pixels.

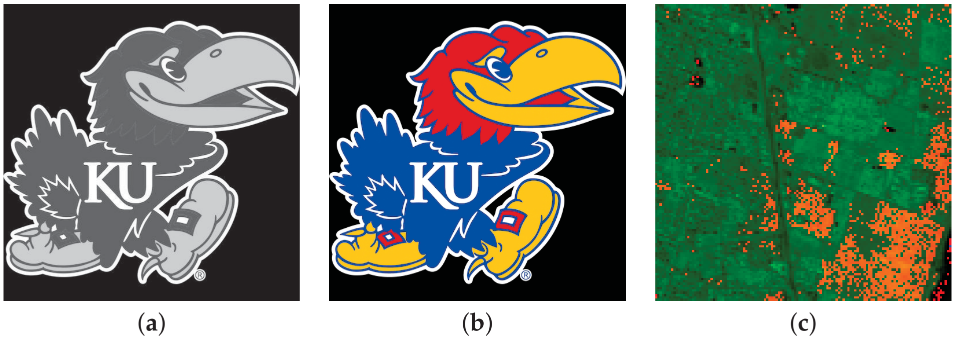


Figure 13. Real-world, high-resolution, multidimensional input data used in experimental trials: (**a**) 2-D B/W image [33], (**b**) 3-D RGB image [33], and (**c**) 3-D hyperspectral image [34].

To validate the accuracy of the proposed pooling techniques, fidelity was measured over multiple levels of decomposition. The metric of data fidelity (see (36)) is used to measure the similarity of the quantum-pooled data $\mathbf{X}$ compared to the classically pooled data $\mathbf{Y}$ [36]. As expressed during testing, pooling was performed on all tested dimensions until one dimension could not be further decomposed. For example, for a hyperspectral image of $(128 \times 128 \times 128)$ pixels, ℓ was varied from 1 to $\lceil \log_2 \min(128, 128, 128) \rceil = 7$, i.e., $\ell = 1, 2, 3, 4, 5, 6,$ and 7.

Classical average and Euclidean pooling were performed using the `PyWavelets` library [37]. Using the `Qiskit` SDK (v0.45.0) from IBM Quantum [19], simulations were run with the quantum average and Euclidean pooling circuits over the given data in both noise-free and noisy (with 32,000 and 1,000,000 circuit samples/shots) environments to display the effect of quantum statistical noise on the fidelity of the results. The experiments were performed at the University of Kansas on a computer cluster node [38] populated with a 48-Core Intel Xeon Gold 6342 CPU, 3×NVIDIA A100 80 GB GPUs (CUDA version 11.7), 256 GB of 3200 MHz DDR4 RAM, and PCIe 4.0 connectivity.

$$\text{Fidelity}(\mathbf{X}, \mathbf{Y}) = \frac{\langle \mathbf{X}, \mathbf{Y} \rangle}{\|\mathbf{X}\|_F \|\mathbf{Y}\|_F} \tag{36}$$

5.2. Results and Analysis

Across all our experiments, the noise-free quantum results showed 100% fidelity compared to the corresponding classical results, validating the correctness and theoretical soundness of our proposed quantum average and Euclidean pooling methods. However, for practical quantum environments, we observe measurement and statistical errors that are intrinsic to noisy quantum hardware, which results in a decrease in fidelity. Sample average and Euclidean pooling results are presented in Tables 1–4 for 1-D audio data, 2-D B/W images, 3-D RGB images, and 3-D hyperspectral images, respectively, for noisy trials of 32,000 and 1,000,000 circuit samples (shots).

Table 1. Noisy simulation outputs for 1-D average and Euclidean pooling on audio (1-D) data [32] with 1,048,576 audio samples.

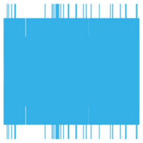
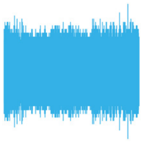
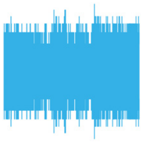
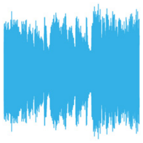
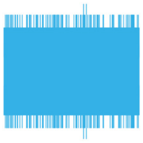
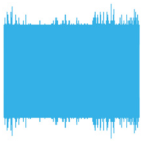
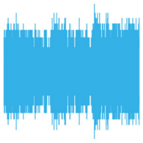
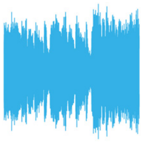
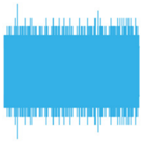
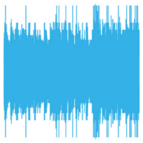
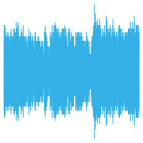
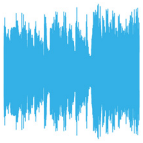
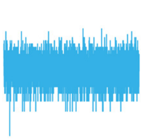
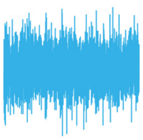
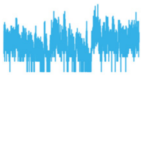
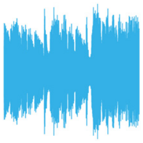
Levels of Decomposition	Average Pooling (32,000 Shots)	Average Pooling (1,000,000 Shots)	Euclidean Pooling (32,000 Shots)	Euclidean Pooling (1,000,000 Shots)
1 Level				
2 Levels				
4 Levels				
8 Levels				

Table 2. Noisy simulation outputs for 2-D average and Euclidean pooling on B/W (2-D) data of size (512 × 512) pixels.

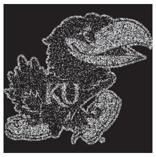
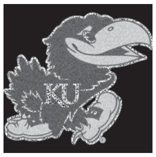
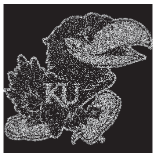
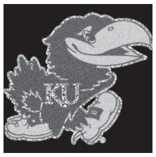
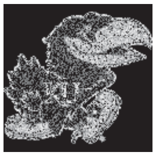
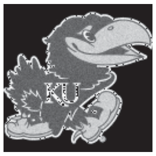
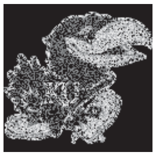
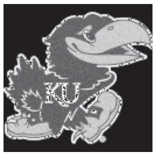
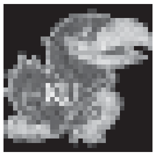
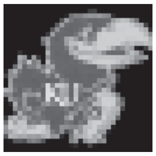
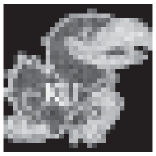
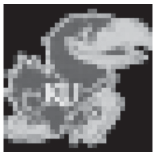
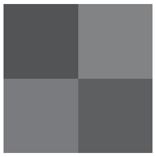
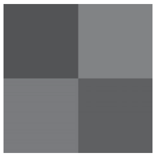
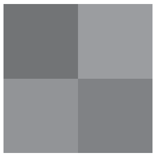
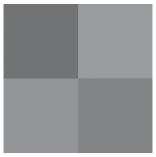
Levels of Decomposition	Average Pooling (32,000 Shots)	Average Pooling (1,000,000 Shots)	Euclidean Pooling (32,000 Shots)	Euclidean Pooling (1,000,000 Shots)
1 Level				
2 Levels				
4 Levels				
8 Levels				

Table 3. Noisy simulation outputs for 2-D average and Euclidean pooling on RGB (3-D) data of size (512 × 512 × 3) pixels.

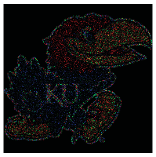
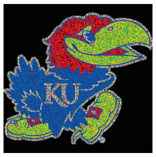
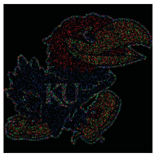
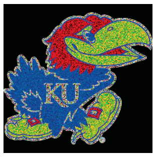
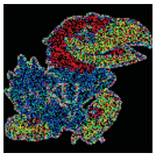
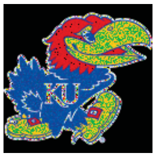
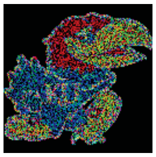
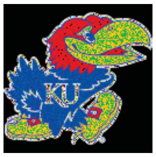
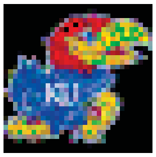
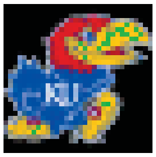
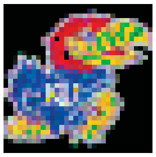
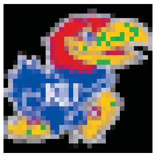
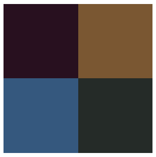
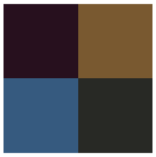
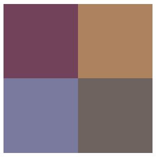
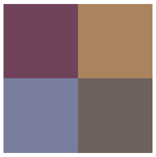
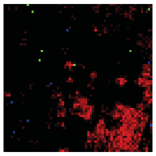
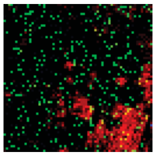
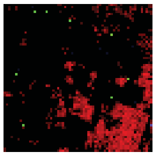
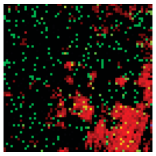
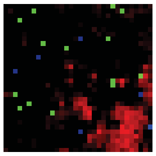
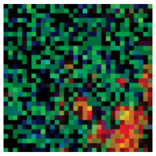
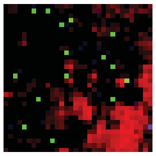
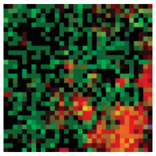
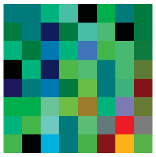
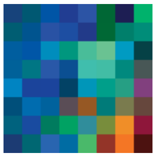
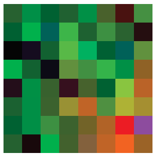
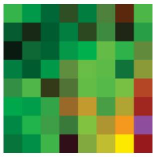
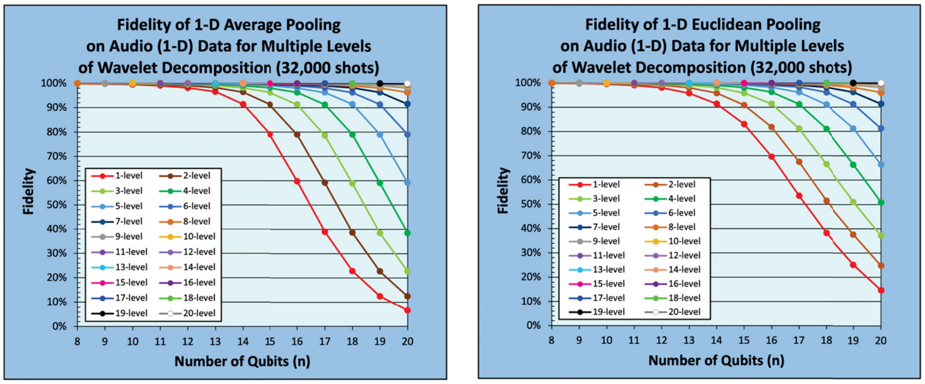
Levels of Decomposition	Average Pooling (32,000 Shots)	Average Pooling (1,000,000 Shots)	Euclidean Pooling (32,000 Shots)	Euclidean Pooling (1,000,000 Shots)
1 Level				
2 Levels				
4 Levels				
8 Levels				

Table 4. Noisy simulation outputs for 3-D average and Euclidean pooling on hyperspectral (3-D) data of size (128 × 128 × 128) pixels.

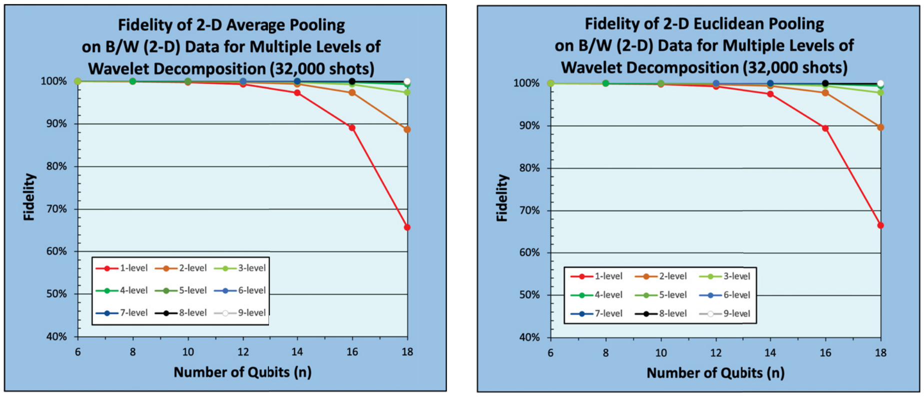
Levels of Decomposition	Average Pooling (32,000 Shots)	Average Pooling (1,000,000 Shots)	Euclidean Pooling (32,000 Shots)	Euclidean Pooling (1,000,000 Shots)
1 Level				
2 Levels				
4 Levels				
7 Levels				

The presented results in Figures 14–21 report the fidelity of the quantum-pooled data using our proposed quantum average and quantum Euclidean pooling techniques with respect to the corresponding classically pooled data in terms of the data size indicated by the number of required qubits n for different levels of decomposition ℓ. The one-dimensional audio data results are shown in Figures 14 and 18 for 32,000 and 1,000,000 shots, respectively, while results for the 2-D B/W images are shown in Figures 15 and 19. In a similar fashion, results for the 3-D RGB images are shown in Figures 16 and 20, and, finally, results for the 3-D hyperspectral data are shown in Figures 17 and 21, all for 32,000 and 1,000,000 shots, respectively.



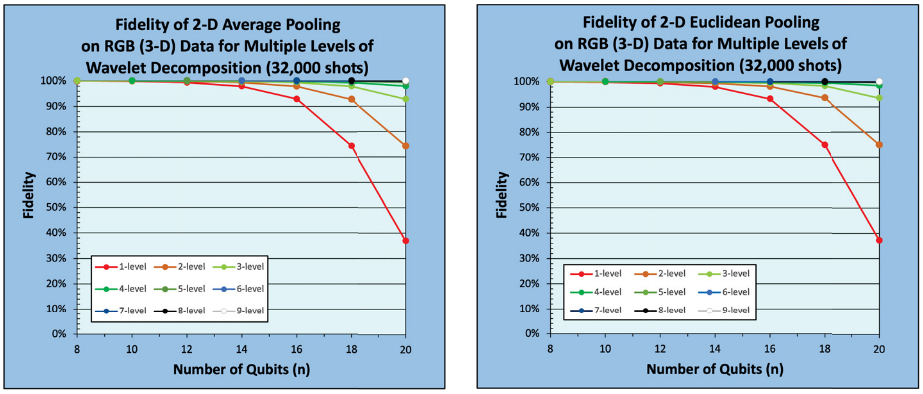
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 14. Fidelity of 1-D pooling on 1-D audio data (32,000 shots).



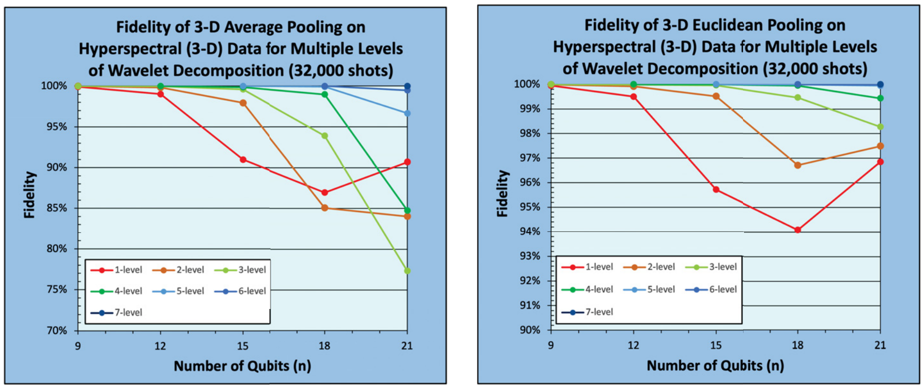
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 15. Fidelity of 2-D pooling on 2-D B/W images (32,000 shots).



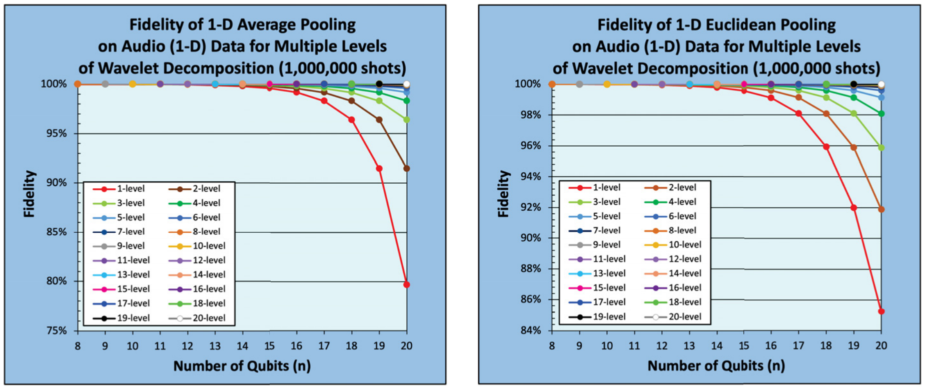
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 16. Fidelity of 2-D pooling on 3-D RGB images (32,000 shots).



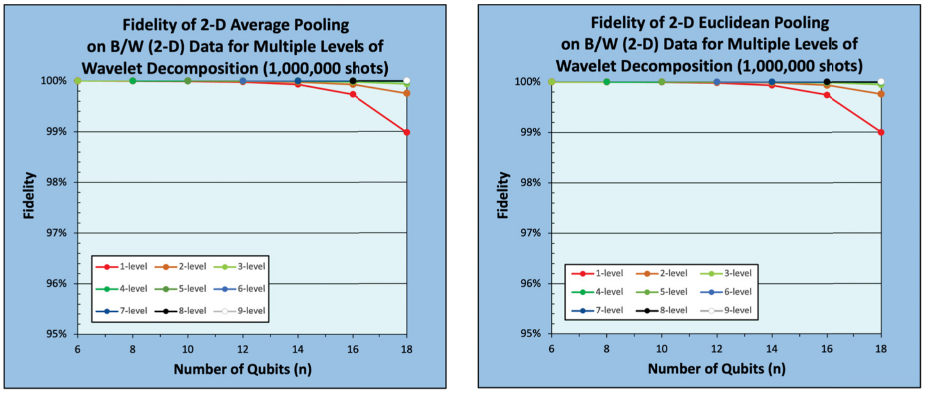
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 17. Fidelity of 3-D pooling on hyperspectral images (32,000 shots).



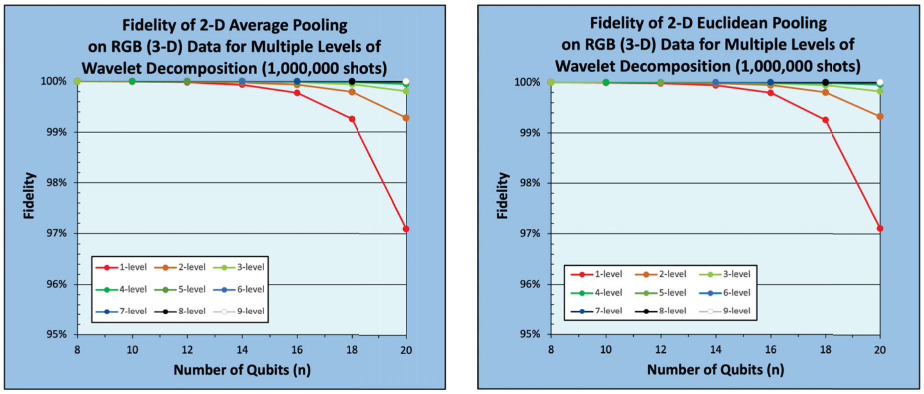
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 18. Fidelity of 1-D Pooling on 1-D Audio data (1,000,000 shots).



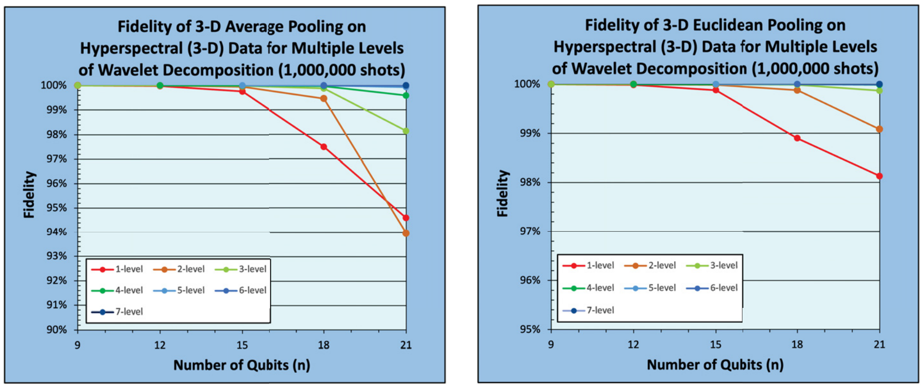
(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 19. Fidelity of 2-D Pooling on 2-D B/W Images (1,000,000 shots).



(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 20. Fidelity of 2-D Pooling on 3-D RGB Images (1,000,000 shots).



(**a**) Average Pooling (**b**) Euclidean Pooling

Figure 21. Fidelity of 3-D Pooling on 3-D Hyperspectral Images (1,000,000 shots).

From Figures 14–21, it can be easily observed that fidelity monotonically decreases with respect to data size (number of qubits) for a given decomposition level. In contrast, a monotonic increase in fidelity with respect to the number of decomposition levels for a given data size is observed; see Figures 14–21. As the data size increases, the size of the corresponding quantum state also increases, which leads to statistical undersampling [6], a phenomenon that occurs when the number of measurement shots is insufficient to accurately characterize the measured quantum state. In quantum Euclidean pooling, partial measurement helps mitigate undersampling because the increase in decomposition levels reduces the number of qubits being measured, resulting in reduced effects of statistical undersampling/noise. A similar behavior occurs with quantum average pooling since the high-frequency terms are sparse and/or close to 0. Nevertheless, quantum Euclidean pooling tends to achieve a slightly higher fidelity compared to quantum average pooling; see Figures 14–21.

Table 5 compares the time complexity and generalizability of our proposed quantum pooling techniques to the existing classical and quantum pooling techniques. Compared to classical pooling techniques, our methods of average and Euclidean pooling can be performed in constant time for arbitrary data dimension and arbitrary pooling window size. The PQC-based techniques used in QCNNs [10], and their derivatives [9,12,26–28], are difficult to compare to other techniques since they do not perform the same pooling operation. We can determine, however, that the additional ansatz for the PQC-based techniques would cause deeper quantum circuits compared to our proposed techniques. Finally, the measurement-based technique proposed in [29] is similar to our technique of single-level decomposition of 2-D Euclidean pooling (although their work inaccurately claims to perform average pooling). However, our work is more generalizable for arbitrary window sizes and data dimensions without compromising performance.

Table 5. Comparison of Related Work to Proposed Methods.

	Classical [21–23]	**PQC-Based [10]**	**Measurement-Based [29]**	**Proposed**
Pooling Method	Arbitrary	N/A	Euclidean	Average, Euclidean
Time Complexity	$\mathcal{O}(2^n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Data Dimension	2-D, 3-D	N/A	2-D	Arbitrary
Window Size	Arbitrary	N/A	(2×2)	Arbitrary

6. Conclusions

In this work, we proposed efficient quantum average and Euclidean pooling methods for multidimensional data that can be used in quantum machine learning (QML). Compared to existing classical and quantum techniques of pooling, our proposed techniques are highly generalizable for any dimensionality of data or levels of decomposition. Moreover, compared to the existing classical pooling techniques on GPUs, our proposed techniques can achieve significant speedup—from $\mathcal{O}(N)$ to $\mathcal{O}(1)$ for a data size of N values. We experimentally validated the correctness of our proposed quantum pooling techniques against the corresponding classical pooling techniques on 1-D audio data, 2-D image data, and 3-D hyperspectral data in a noise-free quantum simulator. We also presented results illustrating the effect on fidelity due to statistical and measurement errors using noisy quantum simulation. In future work, we will explore applications of the proposed pooling layers in QML algorithms.

Author Contributions: Conceptualization: M.J., V.J., D.K. and E.E.-A.; Methodology: M.J., V.J., D.K. and E.E.-A.; Software: M.J., D.L., D.K. and E.E.-A.; Validation: M.J., V.J., D.L., A.N., D.K. and E.E.-A.; Formal analysis: M.J., V.J., D.L., D.K. and E.E.-A.; Investigation: M.J., A.N., V.J., D.L., D.K., M.C., I.I., E.B., E.V., A.F., M.S., A.A. and E.E.-A.; Resources: M.J., A.N., V.J., D.L., D.K., M.C., I.I., E.B., E.V., A.F., M.S., A.A. and E.E.-A.; Data curation: M.J., A.N., V.J., D.L., D.K., M.C., I.I., E.B., E.V., A.F., M.S., A.A. and E.E.-A.; Writing—original draft preparation: M.J., A.N., V.J., D.L., D.K., M.C., I.I. and E.E.-A.; Writing—review and editing: M.J., A.N., V.J., D.L., D.K., M.C., I.I., E.B., E.V., A.F., M.S., A.A. and E.E.-A.; Visualization: M.J., A.N., V.J., D.L., D.K., M.C., I.I., E.B., E.V., A.F., M.S., A.A. and E.E.-A.; Supervision: E.E.-A.; Project administration: E.E.-A.; Funding acquisition: E.E.-A. All authors have read and agreed to the published version of the manuscript.

Funding: This research used resources from the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

Data Availability Statement: The Jayhawk images used in this work are available on request from Brand Center, University of Kansas, at https://brand.ku.edu/ (last accessed on 12 December 2023) [33]. The audio samples used in this work are publicly available from the European Broadcasting Union at https://tech.ebu.ch/publications/sqamcd (accessed on 19 October 2023) as file `64.flac` [32]. The hyperspectral data used in this work are publicly available from the Grupo de Inteligencia Computacional (GIC) at https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Kennedy_Space_Center_(KSC) (accessed on 19 October 2023) under the heading Kennedy Space Center (KSC) [34].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. LeCun, Y.; Kavukcuoglu, K.; Farabet, C. Convolutional networks and applications in vision. In Proceedings of the 2010 IEEE International Symposium on Circuits and Systems, Paris, France, 30 May–2 June 2010; IEEE: New York, NY, USA, 2010, pp. 253–256.
2. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
3. Gholamalinezhad, H.; Khosravi, H. Pooling methods in deep neural networks, a review. *arXiv* **2020**, arXiv:2009.07485.
4. Chen, F.; Datta, G.; Kundu, S.; Beerel, P.A. Self-Attentive Pooling for Efficient Deep Learning. In Proceedings of the the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–7 January 2023; pp. 3974–3983.
5. Tabani, H.; Balasubramaniam, A.; Marzban, S.; Arani, E.; Zonooz, B. Improving the efficiency of transformers for resource-constrained devices. In Proceedings of the 2021 24th Euromicro Conference on Digital System Design (DSD), Sicily, Italy, 1–3 September 2021; IEEE: New York, NY, USA, 2021, pp. 449–456.
6. Jeng, M.; Islam, S.I.U.; Levy, D.; Riachi, A.; Chaudhary, M.; Nobel, M.A.I.; Kneidel, D.; Jha, V.; Bauer, J.; Maurya, A.; et al. Improving quantum-to-classical data decoding using optimized quantum wavelet transform. *J. Supercomput.* **2023**, *79*, 20532–20561. [CrossRef]
7. Rohlfing, C.; Cohen, J.E.; Liutkus, A. Very low bitrate spatial audio coding with dimensionality reduction. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 741–745. [CrossRef]
8. Ye, J.; Janardan, R.; Li, Q. GPCA: an efficient dimension reduction scheme for image compression and retrieval. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 22–25 August 2004; ACM: New York, NY, USA, 2004; pp. 354–363. [CrossRef]
9. Hur, T.; Kim, L.; Park, D.K. Quantum convolutional neural network for classical data classification. *Quantum Mach. Intell.* **2022**, *4*, 3. [CrossRef]
10. Cong, I.; Choi, S.; Lukin, M.D. Quantum convolutional neural networks. *Nature Phys.* **2019**, *15*, 1273–1278. [CrossRef]
11. Biamonte, J.; Wittek, P.; Pancotti, N.; Rebentrost, P.; Wiebe, N.; Lloyd, S. Quantum machine learning. *Nature* **2017**, *549*, 195–202. [CrossRef] [PubMed]
12. Monnet, M.; Gebran, H.; Matic-Flierl, A.; Kiwit, F.; Schachtner, B.; Bentellis, A.; Lorenz, J.M. Pooling techniques in hybrid quantum-classical convolutional neural networks. *arXiv* **2023**, arXiv:2305.05603.
13. Peruzzo, A.; McClean, J.; Shadbolt, P.; Yung, M.H.; Zhou, X.Q.; Love, P.J.; Aspuru-Guzik, A.; O'Brien, J.L. A variational eigenvalue solver on a photonic quantum processor. *Nature Commun.* **2014**, *5*, 4213. [CrossRef] [PubMed]
14. Williams, C.P.; *Explorations in Quantum Computing*, 2nd ed.; Springer: London, UK, 2011.
15. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*; Cambridge University Press: Cambridge, UK, 2010. [CrossRef]
16. Walsh, J.L. A Closed Set of Normal Orthogonal Functions. *Am. J. Math.* **1923**, *45*, 5–24. [CrossRef]
17. Shende, V.; Bullock, S.; Markov, I. Synthesis of quantum-logic circuits. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2006**, *25*, 1000–1010. [CrossRef]
18. El-Araby, E.; Mahmud, N.; Jeng, M.J.; MacGillivray, A.; Chaudhary, M.; Nobel, M.A.I.; Islam, S.I.U.; Levy, D.; Kneidel, D.; Watson, M.R.; et al. Towards Complete and Scalable Emulation of Quantum Algorithms on High-Performance Reconfigurable Computers. *IEEE Trans. Comput.* **2023**, *72*, 2350–2364. [CrossRef]
19. IBM Quantum. Qiskit: An Open-Source Framework for Quantum Computing. Zenodo. 2023. Available online: https://zenodo.org/records/8190968 (accessed on 30 December 2023).
20. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
21. NVIDIA. cuDNN—cudnnPoolingForward() [Computer Software]. 2023. Available online: https://docs.nvidia.com/deeplearning/cudnn/api/index.html#cudnnPoolingForward (accessed on 30 December 2023).
22. NVIDIA. TensorRT—Pooling [Computer Software]. 2023. Available online: https://docs.nvidia.com/deeplearning/tensorrt/operators/docs/Pooling.html (accessed on 30 December 2023).
23. NVIDIA. Pooling. 2023. Available online: https://docs.nvidia.com/deeplearning/performance/dl-performance-memory-limited/index.html#pooling (accessed on 30 December 2023).
24. Schlosshauer, M. Quantum decoherence. *Phys. Rep.* **2019**, *831*, 1–57. [CrossRef]
25. Mahmud, N.; MacGillivray, A.; Chaudhary, M.; El-Araby, E. Decoherence-optimized circuits for multidimensional and multilevel-decomposable quantum wavelet transform. *IEEE Internet Comput.* **2021**, *26*, 15–25. [CrossRef]
26. MacCormack, I.; Delaney, C.; Galda, A.; Aggarwal, N.; Narang, P. Branching quantum convolutional neural networks. *Phys. Rev. Res.* **2022**, *4*, 013117. [CrossRef]
27. Chen, G.; Chen, Q.; Long, S.; Zhu, W.; Yuan, Z.; Wu, Y. Quantum convolutional neural network for image classification. *Pattern Anal. Appl.* **2023**, *26*, 655–667. [CrossRef]
28. Zheng, J.; Gao, Q.; Lü, Y. Quantum graph convolutional neural networks. In Proceedings of the 2021 40th Chinese Control Conference (CCC), Shanghai, China, 26–28 July 2021; IEEE: Piscataway, NY, USA, 2021; pp. 6335–6340.
29. Wei, S.; Chen, Y.; Zhou, Z.; Long, G. A quantum convolutional neural network on NISQ devices. *AAPPS Bull.* **2022**, *32*, 1–11. [CrossRef]

30. Bieder, F.; Sandkühler, R.; Cattin, P.C. Comparison of Methods Generalizing Max- and Average-Pooling. *arXiv* **2021**, arXiv:2104.06918.
31. PyTorch. torch.nn.LPPool1d [Computer Software]. 2023. Available online: https://pytorch.org/docs/stable/generated/torch.nn.LPPool1d.html (accessed on 30 December 2023).
32. Geneva, S. Sound Quality Assessment Material: Recordings for Subjective Tests. 1988. Available online: https://tech.ebu.ch/publications/sqamcd (accessed on 30 December 2023).
33. Brand Center, University of Kansas. Jayhawk Images. Available online: https://brand.ku.edu/ (accessed on 30 December 2023).
34. Graña, M.; Veganzons, M.A.; Ayerdi, B. Hyperspectral Remote Sensing Scenes. Available online: https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Kennedy_Space_Center_(KSC) (accessed on 30 December 2023).
35. IBM Quantum. qiskit.circuit.QuantumCircuit.measure [Computer Software]. 2023. Available online: https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.QuantumCircuit#measure (accessed on 30 December 2023).
36. Jeng, M.; Nobel, M.A.I.; Jha, V.; Levy, D.; Kneidel, D.; Chaudhary, M.; Islam, S.I.U.; El-Araby, E. Multidimensional Quantum Convolution with Arbitrary Filtering and Unity Stride. In Proceedings of the IEEE International Conference on Quantum Computing and Engineering (QCE23), Bellevue, WA, USA, 17–22 September 2023.
37. Lee, G.R.; Gommers, R.; Waselewski, F.; Wohlfahrt, K.; O'Leary, A. PyWavelets: A Python package for wavelet analysis. *J. Open Source Softw.* **2019**, *4*, 1237. [CrossRef]
38. KU Community Cluster, Center for Research Computing, University of Kansas. Available online: https://crc.ku.edu/systems-services/ku-community-cluster (accessed on 30 December 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.