Quantum Image Processing

Alok Anand
Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, USA
aloka@andrew.cmu.edu

Meizhong Lyu
Material Sciences and Engineering
Carnegie Mellon University
Pittsburgh, USA
meizhonl@andrew.cmu.edu

Prabh Simran Baweja
Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, USA
pbaweja@andrew.cmu.edu

Vinay Patil
Electrical & Computer Engineering
Carnegie Mellon University
Pittsburgh, USA
vspatil@andrew.cmu.edu

I. Introduction

A. Motivation

Image processing is popular in our daily life because of the need to extract essential information from our 3D world, including a variety of applications in widely separated fields like biomedicine, economics, entertainment, and industry. The nature of visual information, algorithm complexity, and the representation of 3D scenes in 2D spaces are all popular research topics. In particular, the rapidly increasing volume of image data as well as increasingly challenging computational tasks have become important driving forces for further improving the efficiency of image processing and analysis.

B. Why Quantum?

Since the concept of quantum computing was proposed by Feynman in 1982, many achievements have shown that quantum computing has dramatically improved computational efficiency [1]. Quantum information processing exploit quantum mechanical properties, such as quantum superposition, entanglement and parallelism, and effectively accelerate many classical problems like factoring large numbers, searching an unsorted database, Boson sampling, quantum simulation, solving linear systems of equations, and machine learning. These unique quantum properties may also be used to speed up signal and data processing. In quantum image processing, quantum image representation plays a key role, which substantively determines the kinds of processing tasks and how well they can be performed.

II. Literature Review

A. Classical Image Representation

Images in classical computers are defined as matrices of numbers representing the discrete color or intensity values present in every image pixel. Each image is considered as input data displayable in numerous ways, whether as arrays of pixel values or either multidimensional plots representing the distribution of pixel intensities. Images can be rendered in color layered with 3 channels (Blue, Green, and Red), Grayscale with pixel values varying from 0 (black) to 255 (white), and binary portraying black or white values (0 or 1) only.

B. Quantum Image Representation (QIR)

Quantum image processing is one of the most attractive and promising tools within the Quantum Technology toolbox. The representation of an image on a quantum computer in the form of normalized states facilitates numerous image processing problems [2].

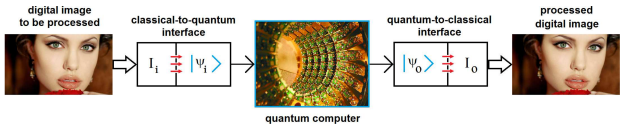

Fig. 1. Scheme of Quantum Image Processing (QImP)

The following methods have been proposed for QIR:

1) ***Qubit Lattice****:* In 2003, Venegas-Andraca and Bose suggested the *Qubit Lattice* model [3] to map image's spatial information with the amplitude of a single qubit without using quantum properties, therefore requiring the same number of qubits as pixels. This is a quantum-analog representation of classical images. The pixel value of i^{th} row and the j^{th} column can be stored as:

$$|pixel_{i,j}\rangle = (\cos\left(\frac{\theta_{i,j}}{2}\right)|0\rangle + \sin\left(\frac{\theta_{i,j}}{2}\right)|1\rangle) \quad (1)$$

2) ***Flexible Representation of Quantum Images (FRQI)****:* FRQI, proposed by Le et al. [4], maps each pixel's grayscale value to the amplitude as well as captures the corresponding positions in an image and integrates them into a quantum state. The FRQI representation is expressed as:

$$|I(\theta)\rangle = \frac{1}{2^n}\sum_{i=0}^{2n-1}(\sin(\theta_i)|0\rangle + \cos(\theta_i)|1\rangle)|i\rangle \quad (2)$$

where θ_i encodes pixel value of the corresponding position $|i>$. The FRQI representation maintains a normalized state and the representation space decreases exponentially compared to the classical image due to the quantum states' superposition effect. FRQCI and IFRQI are two improved flexible representation of quantum images reported by Li et al. [5] and Khan et al. [6] respectively.

*3) **Novel Enhanced Quantum Representation (NEQR)**:* Zhang et al. [7] reported a representation that uses the basis state of a qubit sequence to store the grayscale value of every pixel instead of probability amplitude encoded in a qubit as in FRQI. The images are stored by entangling color information, represented by $|f(y,x)\rangle$, with location information, represented by $|yx\rangle$. The NEQR representation for a $2^n \times 2^n$ image is expressed as:

$$|I\rangle = \frac{1}{2^n} \sum_{y=0}^{2^{2n}-1} \sum_{x=0}^{2^{2n}-1} |f(y,x)\rangle |yx\rangle \tag{3}$$

where $f(y,x)$ refers to the pixel intensity at $f(y,x)$. NEQR could perform the complex and elaborate color operations more conveniently than FRQI does. NEQR can achieve a quadratic speedup in quantum image preparation and retrieve digital images from quantum images accurately. However, NEQR representation uses more qubits to encode a quantum image. Researchers have proposed a few variations for NEQR as well: Improved NEQR (INEQR) [8], generalized model of NEQR (GNEQR) [9] and CQIR [10].

*4) **Normal Arbitrary Quantum Superposition State (NAQSS)**:* NAQSS was proposed to solve multi-dimensional color image processing by Li et al. [11]. The NAQSS is $a(n+1)$- qubit quantum representation, which can be used to represent multi-dimensional color images. N qubits represent colors and coordinates of 2^n pixels and the remaining 1 qubit represents an image segmentation information to improve the accuracy of image segmentation. The NAQSS corresponds to the color and angle of the image one to one, mapping the color information to a certain value on the interval $[0, \pi/2]$. The NAQSS can be represented as:

$$|I\rangle = \sum_{i=0}^{2^n-1} \theta_i |v_1\rangle |v_2\rangle \ldots |v_k\rangle \otimes |\chi_i\rangle \tag{4}$$

where

$$|\chi_i\rangle = \cos\gamma_i |0\rangle + \sin\gamma_i |1\rangle \tag{5}$$

represents the segmentation information. NAQSS can improve the efficiency and accuracy of image segmentation. However, it cannot accurately measure the pixels of the image.

*5) **Quantum Probability Image Encoding (QPIE)**:* Yao et al.'s QPIE [12] representation has an immediate advantage over the other encoding methods previously discussed is that it allows to encode rectangular r × c images. Moreover, the number of qubits necessary to encode the image are further reduced. The QPIE representation is expressed as:

$$|I\rangle = \sum_{i=0}^{2^{2n}-1} c_i |i\rangle, n = [\log_2(rc)] \tag{6}$$

where

$$I' = (I_{1,1}, I_{2,1}, \ldots, I_{r,1}, I_{1,2}, \ldots, I_{r,c})^T \tag{7}$$

and $c_i = \frac{I'^{(i)}}{||I'||}$ is the normalized value of the i^{th} element of the vector.

This method allows one to encode images of arbitrary size r × c However, this encoding has the problem of the extraction of the exact original image from its encoding quantum circuit. This is because the pixel value is stored in the state amplitude, so that its approximate value can only be restored by multiple measurements.

*6) **Quantum Feautre Map for Data Encoding**:* Classical data is encoded to the quantum state space using a quantum feature map. The choice of the feature map to use is important and depends on the dataset to be classified. A quantum feature map utilises classical feature vector to the quantum state, that includes applying the unitary operation on the initial state used for data encoding. Here we are interested in ZZFeature Map based Quantum SVM, as introduced in Havlicek et al [13]

C. Classical Edge and Corner Detection

1) Edge Detection: Edge detection is an image processing technique for finding the boundaries of an object in the given image. The edges are the part of the image that represents the boundary or the shape of the object in the image. the general method of edge detection is to study the changes of a single image pixel in a gray area, use the variation of the edge neighboring first-order or second-order to detect the edge. There are various methods in edge detection, and following are some of the most commonly used methods:

1) **Prewitt edge detection [14]**: This method is commonly used to detect edges based applying a horizontal and vertical filter in sequence. This gradient based edge detector is estimated in the 3x3 neighborhood for eight directions. All the eight convolution masks are calculated.
2) **Sobel edge detection [15]**: Sobel detection finds edges using the Sobel approximation to the derivative. It precedes the edges at those points where the gradient is highest. It is one of the most commonly used edge detectors and helps reduce noise and provides differentiating, giving edge response simultaneously.
3) **Laplacian edge detection [16]**: This method uses only one filter (also called a kernel). In a single pass, Laplacian edge detection performs second-order derivatives and hence are sensitive to noise. The Laplacian is generally used to found whether a pixel is on the dark or light side of an edge.
4) **Canny edge detection [17]**: This is the most commonly used highly effective and complex compared to many

other methods. It is a better method that without disturbing the features of the edges in the image afterwards and it apply the tendency to find edges and serious value of threshold. The algorithmic steps are as follows:

a) Convolve an image $f(r,c)$ with a Gaussian function to get smooth image $\hat{f(r,c)}.\hat{f(r,c)} = f(r,c) * G(r,c,6)$.
b) Apply first difference gradient operator to compute edge strength, then edge magnitude and direction are obtained as before.
c) Apply non-maximal or critical suppression to the gradient magnitude.
d) Apply threshold to the non-maximal suppression image.

2) Corner Detection: Corner detection works on the principle that if a small window is placed over an image, and that window is placed on a corner. If that window is moved in any direction, then there will be a large change in intensity.

1) **The Moravec corner detection [18]**: This is one of the earliest corner detection algorithms and defines a corner to be a point with low self-similarity. The algorithm tests each pixel in the image to see if a corner is present, by considering how similar a patch entered on the pixel is to nearby, largely overlapping patches.
2) **Harris corner detector [19]**: This method was developed to identify the internal corners of an image and it is realized by calculating each pixel's gradient. It is based on the local auto-correlation function of a signal which measures the local changes of the signal with patches shifted by a small amount in different directions.
3) **The Förstner corner detector [20]**: This method solves for the point closest to all the tangent lines of the corner in a given window and is a least-square solution in order to achieve an approximate solution. The algorithm relies on the fact that for an ideal corner, tangent lines cross at a single point.
4) **SUSAN (Smallest Univalue Segment Assimilating Nucleus) corner detector [21]**: This method is based on brightness comparison and realized by a circular mask. If the brightness of each pixel within a mask is compared with the brightness of that mask's nucleus, then an area of the mask can be defined which has the same (or similar) brightness as the nucleus. The SUSAN area will reach a minimum while the nucleus lies on a corner point.
5) **Fuzzy System [22]**: The measure of "cornerness" for each pixel in the image is computed by fuzzy rules (represented as templates) which are applied to a set of pixels belonging to a rectangular window. The possible uncertainty contained in the window-neighborhood is handled by using an appropriate rule base (template set). The fuzzy system is simple to implement and still fast in computation when compared to some existing fuzzy methods. Also, it can be easily extended to detect other features.

D. Quantum Methods in Edge Detection

In the known classical edge extraction algorithms, for a typical image of 2^n x 2^n processing work with the computational complexity less than $O(2^{2n})$, it is impossible to complete such a task [23]. By contrast, quantum image processing utilizes the superposition and entanglement characteristics of quantum mechanics to simultaneously calculate all the pixels of the image, thus realizing the acceleration of the algorithm. Basically, the quantum edge detection methods are the combination of QIR methods and classical edge methods.

1) **QSobel and other quantum image model based methods**: Early in 2014, Zhang et al. reported a novel quantum image edge extraction algorithm called QSobel [24]. It is designed based on the FRQI and the famous edge extraction algorithm Sobel. Fan et al. proposed an image edge detection method based on Laplacian operator and Sobel in 2019 [25]. In 2020, Xu et al. proposed a quantum image processing algorithm using edge extraction based on Kirsch operator [23]. A quantum edge detection algorithm based on improved eight-direction Sobel operator was proposed by Ma et al. in 2020 [26]. The flow chart is shown in the Figure 2, which is divided into 4 stages more specifically. The classical digital image is quantized into quantum image NEQR model first. Then, the X-Shift and Y-Shift transformations are used to obtain the shifted image set. Following that, very pixel's gradient is further calculated according to Sobel mask using the shifted image set simultaneously. Finally, the edge of the original image is extracted through the threshold operation U_T.

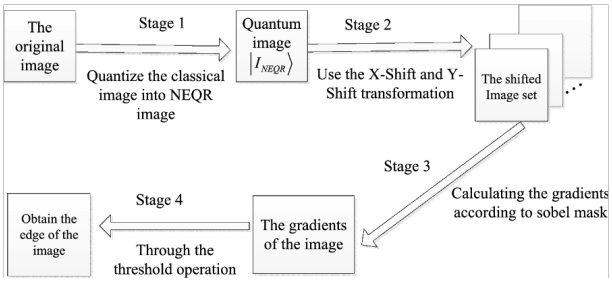

Fig. 2. Basic workflow of quantum image edge extraction, image from Fan et al.'s work [25]

2) **Quantum Hadamard Edge Detection (QHED)**: Another kind of quantum edge detection algorithm calculated the difference between adjacent pixels in the quantum image via Hadamard transform, this method is called Quantum Hadamard Edge Detection (QHED), which is reported by Yao et al. in 2017 [27]. This QHED algorithm encode the pixel values of the image in the probability amplitudes and the pixel positions in the computational basis states. In a N-pixel image, the pixels of the image are numbered in the form of $|b_1 b_2 ... b_{n-1} 0>$, where b_i = 0 or 1. The positions of any pair of neighboring pixels in a picture column are given by the

binary sequences $|b_1 b_2 ... b_{n-1} 0>$ and $|b_1 b_2 ... b_{n-1} 1>$, and the corresponding pixel intensity values are stored as the $c_{b_1 b_2 ... b_{n-1} 0}$ and $c_{b_1 b_2 ... b_{n-1} 1}$. By applying the Hadamard transform to the least significant bit on an arbitrary size quantum register. The total operation is then,

$$\mathrm{I}_{2^{n-1}} \otimes \mathrm{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 & & 0 & 0 \\ 1 & -1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & -1 & & 0 & 0 \\ & & \vdots & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & 1 \\ 0 & 0 & 0 & 0 & & 1 & -1 \end{bmatrix} \tag{8}$$

where $I_{2^{n-1}}$ is the 2^{n-1}X 2^{n-1} unit matrix.
Applying this unitary to a quantum register containing pixel values encoded using the QPIE representation, as

$$(\mathrm{I}_{2^{n-1}} \otimes \mathrm{H}) \cdot \begin{bmatrix} \mathrm{c}_0 \\ \mathrm{c}_1 \\ \mathrm{c}_2 \\ \mathrm{c}_3 \\ \vdots \\ \mathrm{c}_{\mathrm{N}-2} \\ \mathrm{c}_{\mathrm{N}-1} \end{bmatrix} \rightarrow \frac{1}{\sqrt{2}} \begin{bmatrix} \mathrm{c}_0 + \mathrm{c}_1 \\ \mathrm{c}_0 - \mathrm{c}_1 \\ \mathrm{c}_2 + \mathrm{c}_3 \\ \mathrm{c}_2 - \mathrm{c}_3 \\ \vdots \\ \mathrm{c}_{\mathrm{N}-2} + \mathrm{c}_{\mathrm{N}-1} \\ \mathrm{c}_{\mathrm{N}-2} - \mathrm{c}_{\mathrm{N}-1} \end{bmatrix}. \tag{9}$$

It is obvious that we now have access to the gradient between the pixel intensities of neighboring pixels in the form of $c_i - c_{i+1}$. The logic here is: When two pixels belong to the same region, their intensity values are identical and the difference vanishes, otherwise their difference is non-vanishing, which indicates a region boundary. This process results in the detection of horizontal boundaries between the even-pixels-pairs (0 1, 2 3, and so on). For detection of horizontal boundaries between odd-pixel-pairs (1 2, 3 4, etc.), we can perform an amplitude permutation on the quantum register to convert the amplitude vector $(c_0, c_1, c_2, \dots, c_{N1})^T$ to $(c_1, c_1, c_2, \dots, c_{N1}, c_0)^T$, and then applying the H-gate and measuring the quantum register conditioned on LSB being $|1\rangle$.

E. Image Classification

1) ***Support Vector Machine(SVM) and Quantum SVM****:* SVM classification is essentially a binary (two-class) classification technique and kernel methods can achieve more complexity in classical SVM. Similar to support vector machines, the Quantum SVM algorithm applies to classification problems that require a feature-map implicitly specified by a kernel. Quantum SVM using minimization quantum subroutine GroverOptim and provides convergence to a global optimum for non-convex cost function. Quantum SVM and kernels can exploit the higher dimensional space efficiently and can generate feature maps and subsequent decision boundaries that are difficult for classical kernel functions to match. As it mentioned in Park et al.'s work [28], Quantum SVM can potentially provide better performance if the underlying boundary is a complex one not captured well by traditional classical kernels.

2) ***K-means and Quantum K-means****:* K-means algorithm belongs to the family of unsupervised machine learning algorithms that groups together n observations into K clusters making sure intra cluster variance is minimized. The most resource consuming operation for k-means algorithm is calculation of a distance between vectors. The quantum version of the K-means algorithm provides an exponential speed-up for very high dimensional input vectors by the fact that only log N qubits are required to load N-dimensional input vectors using the amplitude encoding [29].

3) ***Convolutional Neural Network(CNN) and Quantum CNN****:* CNN has shown very high performance in computer vision with the advantage of making good use of the correlation information of data. However, CNN is challenging to learn efficiently if the given dimension of data or model becomes too large. Cong extends the convolution layer and the pooling layer,the main features of CNN, to quantum systems as a model of Quantum CNN [30]. The convolution circuit will find the hidden state by applying multiple qubit gates between adjacent qubits and the pooling circuit will reduce the size of the quantum system by observing the fraction of qubits or applying 2-qubit gates such as CNOT gates. After repeating the convolution circuit and pooling circuit until the size of the system is sufficiently small, the fully connected circuit can predict the classification result. The construction of the quantum convolutional layer can be divided into four steps: (1) The encoding process stores the pixel data corresponding to the filter size in qubits; (2) The learnable quantum circuits apply the filters that can find the hidden state from the input state; (3) The decoding process gets new classical data by measurement; (4) Repeat steps (1) to (3) to complete the new feature map. [31]

4) ***Quantum Annealer based feature selection and Image classification****:* Quantum Annealers are in use for operation planning and feature extraction from remotely sensed images. As reported [32], for a real-world dataset in Earth observation, remotely sensed images differ in their image content representations due to the diverse satellite platforms with their different types of sensors. Current implementation includes D-Wave Quantum Annealer (QA) with an Earth observation dataset. Hyperspectral images (HSIs) along with mutual information (MI)-based optimization method is used to select highly informative band subset which is embedded and optimized by the MI-based optimization method in the Pegasus topology of a D-Wave QA. At first, feature selection is performed on a D-Wave QA i.e. MI-based

band subset selection. In second step, binary classification is performed using quantum classifiers characterized by selected bands. Finally, multi-label classification on D-wave QA, where error-encoding output code (ECOC) generates a multilabel dataset that is used by binary quantum classifiers.

III. Datasets

A. MNIST Dataset

This dataset [33] is an extensive set of 60,000 handwritten digits, every image has a dimension of 28x28 pixels. Along with training data it also provides 10,000 additional test dataset consisting of equally distributed digits from 0-9.

B. CIFAR-10 Dataset

This dataset (Learning Multiple Layers of Features from Tiny Images) [34] consists of 60,000 images distributed equally across 10 different classes, each image is 32x32 pixels out of which 10,000 images are used as test dataset. This dataset is divided in 6 batches, 5 batches are used for training the model and 1 batch is used for model evaluation as a test dataset.

C. Synthetic Datasets

Current quantum devices (NISQ) have a limited number of qubits, thereby, limiting the exploration of images with large dimensions. We plan to create synthetic datasets for evaluation of the quantum algorithms. Furthermore, we also plan to apply dimensionality reduction techniques such as Principal Component Analysis (PCA) on publicly available datasets before passing them through the quantum devices.

IV. Results

A. Classical Methods

1) Edge detection with MNIST dataset:

- **Stage 1**: Normalization and Smoothing the image: An example of this stage is shown in Figure 3.
- **Stage 2**: Getting edge related information using gaussian, laplace, sobel filter as demonstrated in Figure 4.
- **Stage 3**: Harris corner detector to detect edges in the digits as demonstrated in Figure 6.
- **Stage 4**: Conversion to bag-of-words representation.
- **Stage 5**: Histogram representation.
- **Stage 6**: Classification using SVM.

The code for the model and the training can be found here and run directly on Google Colab.

2) Deep Learning Model for MNIST dataset: We use the architecture with Resnet [35] blocks, Batch normalization and ReLU activation for classification of MNIST dataset. The architecture provides an accuracy of 99.6% in only 12 epochs. We used the Learning Rate Finder provided by FastAI library [36] and found that the best learning rate for our architecture is 0.05. We used Stochastic Gradient Descent (SGD) as the optimizer and the loss function as Cross Entropy Loss as MNIST is a multi-class classification problem. The code for the model and the training can be found here and run directly on Google Colab.

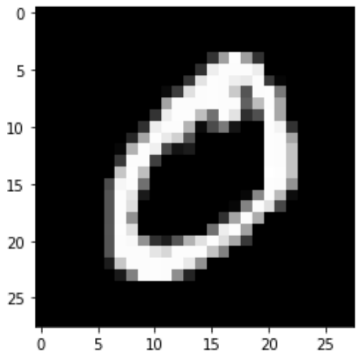

Fig. 3. stage-1 : Normalization and Smoothing the image

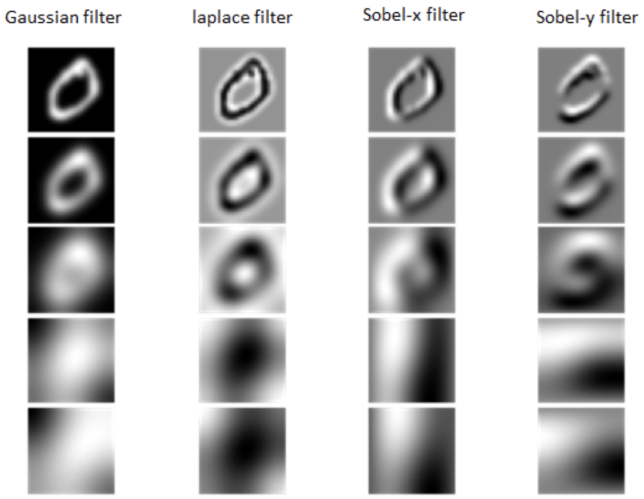

Fig. 4. stage-2 : Getting edge related information using gaussian, laplace, sobel filter.

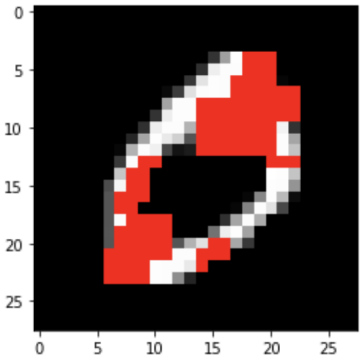

Fig. 5. stage-3 : Harris corner detector to detect edges in the digits.

3) Deep Learning Model for CIFAR-10 dataset: Here we have used SOFTMAX activation function, with 10 units in output layers. In compiling of model, we took account for loss function using Sparse Categorical-Cross Entropy as classes are totally distinctive. We have used Adam Optimizer to adapt

```
(array([[575,   0,   0,   0,   0, 405,   0,   0,   0,   0],
        [  0, 698,   0,   0,   0, 437,   0,   0,   0,   0],
        [  0,   0, 645,   0,   0, 387,   0,   0,   0,   0],
        [  0,   0,   0, 607,   0, 403,   0,   0,   0,   0],
        [  0,   0,   0,   0, 622, 360,   0,   0,   0,   0],
        [  0,   0,   0,   0,   0, 892,   0,   0,   0,   0],
        [  0,   0,   0,   0,   0, 365, 593,   0,   0,   0],
        [  0,   0,   0,   0,   0, 418,   0, 610,   0,   0],
        [  0,   0,   0,   0,   0, 404,   0,   0, 570,   0],
        [  0,   0,   0,   0,   0, 383,   0,   0,   0, 626]]), 0.6438)
```

Fig. 6. Confusion Matrix and Prediction accuracy on Test dataset of MNIST

learning rate, as it updates weights after every iteration. After running 15 epochs, final results obtained with accuracy of 77.5%. Model implementation, training code can be found at following link.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 4096)              0
_________________________________________________________________
dropout (Dropout)            (None, 4096)              0
_________________________________________________________________
dense (Dense)                (None, 128)               524416
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 591,274
Trainable params: 591,274
Non-trainable params: 0
_________________________________________________________________
```

Fig. 7. Deep learning model summary (CNN)

```
[ ] test_loss, test_accuracy = cifar10_model.evaluate(x_test, y_test)

    313/313 [==============================] - 2s 6ms/step - loss: 0.7296 - sparse_categorical_accuracy: 0.7754

[ ] print("Test accuracy: {}".format(test_accuracy))

    Test accuracy: 0.7753999829292297
```

Fig. 8. Deep learning model results (CNN)

B. Quantum Methods

1) Quantum Image classification of MNIST dataset: Here, we have used IBM Qiskit for MNIST data binary classification. We have used smaller samples for training with ZZfeature Map, with feature dimension 5 and circular entanglement. Obtained feature is fed to Quantum kernel with quantum simulator "QASM Simulator", and in next stage using Quantum SVM for training the samples. Testing is performed on different labels from small samples of test data, getting results for image classification of that labels verses rest and prediction probability of image is obtained depicting classification results. The code for the model and the training can be found here.

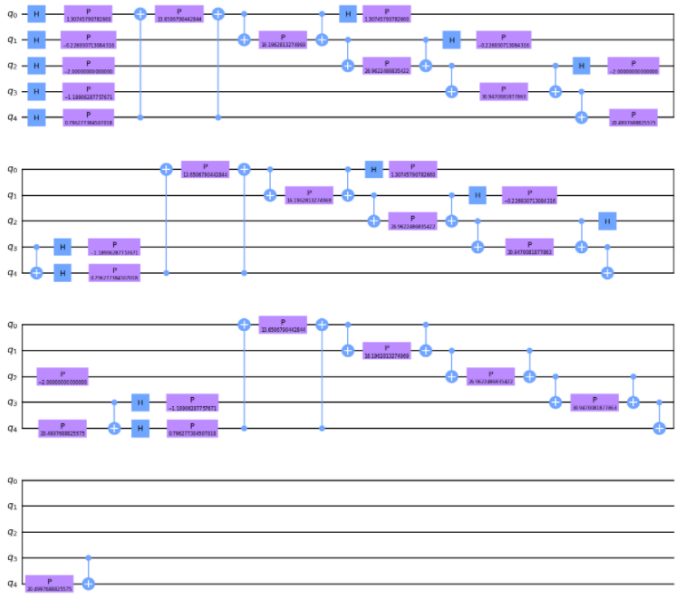

Fig. 9. ZZfeature Map Quantum circuit

```
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [0 0 0 1 0 1 0 0 0 1]
Accuracy of discriminating between label and others: 70.0%
Probability of label: [0.12 0.22 0.1  0.11 0.09 0.09 0.17 0.16 0.09 0.14]
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [0 0 0 0 1 0 0 0 0 0]
Accuracy of discriminating between label and others: 90.0%
Probability of label: [0.07 0.06 0.07 0.1  0.07 0.09 0.09 0.09 0.11 0.1 ]
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [0 0 0 0 0 0 0 0 0 0]
Accuracy of discriminating between label and others: 100.0%
Probability of label: [0.03 0.16 0.07 0.08 0.08 0.14 0.11 0.13 0.08 0.1 ]
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [1 1 0 0 0 0 0 0 1 0]
Accuracy of discriminating between label and others: 70.0%
Probability of label: [0.06 0.06 0.06 0.06 0.06 0.06 0.06 0.06 0.06 0.06]
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [0 0 0 0 0 0 0 0 0 0]
Accuracy of discriminating between label and others: 100.0%
Probability of label: [0.2  0.08 0.   0.2  0.02 0.02 0.01 0.14 0.37 0.07]
Original validation labels:      [6 6 3 9 8 9 4 2 6 9]
Validation labels for 0 vs Rest: [0 0 0 0 0 0 1 0 0 0]
Accuracy of discriminating between label and others: 90.0%
```

Fig. 10. MNIST classification Quantum result

2) Quantum Image Classification using Edge detection of MNIST dataset: This method is splitted into two stages, first stage includes finding edges of the image and in second stage the edges are then used for image classification. The edge detection with classical algorithm is much slower compare to hadamard based edge detection. The time complexity of classical operation is O(2^n) however the quantum system can perform the gradient calculation between pixels much faster using haradamard operation O($mn.log(mn)$). Hence leveraging the faster operating speed this method performs edge detection using quantum circuit and perform classification

using Logistic Regression to get the accuracy of the model. For image representation QPIE (Quantum Probability Image Encoding) is used, where the image is using n = $log_2 N$ qbits to represent an image where N is the number of pixel from the image.

Code: Code can be found here and run directly on Google Colab.

Results:

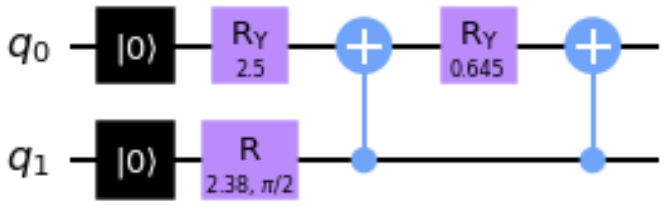

Fig. 11. QPIE image representation

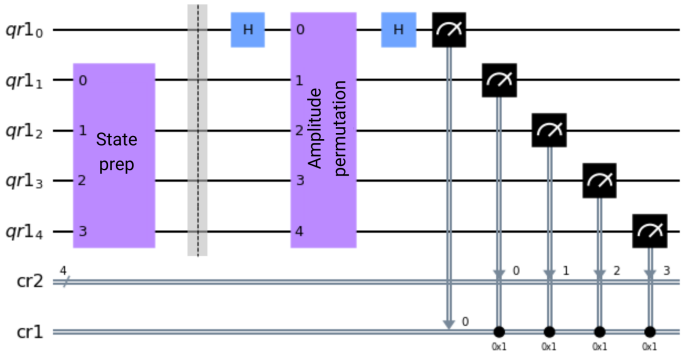

Fig. 12. Quantum Hadamard Edge detection using Simulator

The Algorithm performs two scan for gradient calculations, in first scan only horizontal edges are detected and in second scan the vertical edges are detected. this operation is equivalent to SobelX and SobelY operation in classical algorithms. The

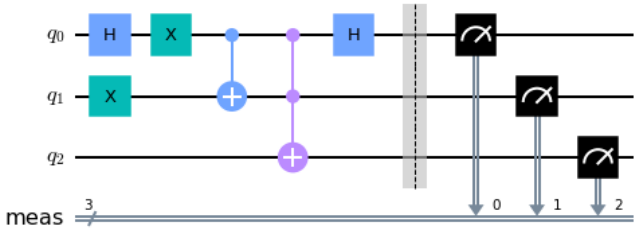

Fig. 13. Quantum Hadamard Edge detection horizontal scan

method, also handles large images by splitting them in smaller patches and applies edge detection separately this is majorly done due to limitation of qbits in actual quantum computers,

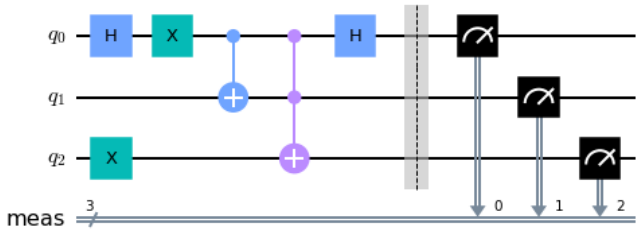

Fig. 14. Quantum Hadamard Edge detection Vertical scan

coupling between qbits and also to reduce the probability of noise impacting the operation. For MNIST the image is scaled to 32x32 pixels and then divided into 4x4 pixel patches, the reason for choosing 32 is to provide support for CIFAR dataset and also increase the quality of edge detection because of up scaling. Coming to stage 2 of the method, classification is

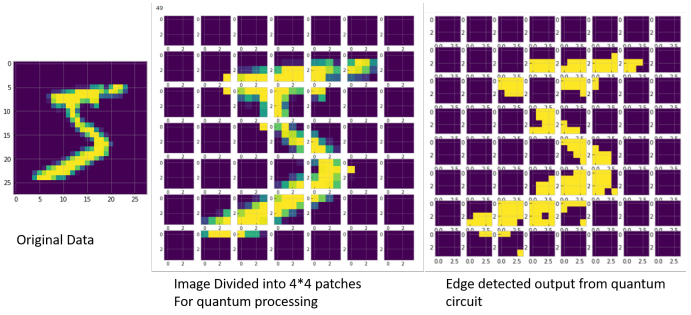

Fig. 15. The flow of QHED with large images.

done using a logistic regression model with mutli-class model including intercept term with Lasso regression as penalty.

```
[Parallel(n_jobs=5)]: Using backend ThreadingBackend with 5 concurrent workers.
convergence after 30 epochs took 1 seconds
Accuracy on test dataset = 0.78
[Parallel(n_jobs=5)]: Done   1 out of   1 | elapsed:    0.5s finished
```

Fig. 16. Accuracy of Method2 with sample MNIST dataset

V. Proposed Plan

- **Quantum Image Representation**: We plan to design quantum circuits for Quantum Image Representation and Quantum Edge Detection techniques. We will evaluate the performance of QPIE and zFeatureMap techniques mentioned in the literature review.
- **Quantum Edge Detection**: We will evaluate the performance of QSobel and it's successors. We will also design the circuit for QHED.
- **Execution on Quantum Devices**: We plan to Run the circuits on actual quantum computers such as qiskit, Dwave and compare the performance

- **Impact of Noise in Quantum Devices**: We will further analyze the impact of various noise models on the algorithms.

1) Test Instance Selection: We will be using the following test instances for our quantum experiments:

- Gate based quantum computing models: We will perform computations on IBM Quantum Devices.
- Annealer based quantum computing models: We will perform computations on D-Wave QA provided by Amazon Braket.

REFERENCES

[1] Yue Ruan, Xiling Xue, and Yuanxia Shen. Quantum image processing: Opportunities and challenges. *Mathematical Problems in Engineering*, 2021, 2021.

[2] Fei Yan, Abdullah M Iliyasu, and Salvador E Venegas-Andraca. A survey of quantum image representations. *Quantum Information Processing*, 15(1):1–35, 2016.

[3] Salvador E Venegas-Andraca and Sougato Bose. Storing, processing, and retrieving an image using quantum mechanics. In Eric Donkor, Andrew R. Pirich, and Howard E. Brandt, editors, *Quantum Information and Computation*, volume 5105, pages 137 – 147. International Society for Optics and Photonics, SPIE, 2003.

[4] Phuc Q Le, Fangyan Dong, and Kaoru Hirota. A flexible representation of quantum images for polynomial preparation, image compression, and processing operations. *Quantum Information Processing*, 10(1):63–84, 2011.

[5] Panchi Li, Hong Xiao, and Binxu Li. Quantum representation and watermark strategy for color images based on the controlled rotation of qubits. *Quantum Information Processing*, 15(11):4415–4440, 2016.

[6] Rabia Amin Khan. An improved flexible representation of quantum images. *Quantum Information Processing*, 18(7):1–19, 2019.

[7] Yi Zhang, Kai Lu, Yinghui Gao, and Mo Wang. Neqr: a novel enhanced quantum representation of digital images. *Quantum information processing*, 12(8):2833–2860, 2013.

[8] RiGui Zhou, WenWen Hu, GaoFeng Luo, XingAo Liu, and Ping Fan. Quantum realization of the nearest neighbor value interpolation method for ineqr. *Quantum Information Processing*, 17(7):1–37, 2018.

[9] Hai-Sheng Li, Xiao Chen, Shuxiang Song, Zhixian Liao, and Jianying Fang. A block-based quantum image scrambling for gneqr. *IEEE Access*, 7:138233–138243, 2019.

[10] Simona Caraiman and Vasile Manta. Image representation and processing using ternary quantum computing. In *International Conference on Adaptive and Natural Computing Algorithms*, pages 366–375. Springer, 2013.

[11] Hai-Sheng Li, Qingxin Zhu, Ri-Gui Zhou, Lan Song, and Xing-jiang Yang. Multi-dimensional color image storage and retrieval for a normal arbitrary quantum superposition state. *Quantum Information Processing*, 13(4):991–1011, 2014.

[12] Xi-Wei Yao, Hengyan Wang, Zeyang Liao, Ming-Cheng Chen, Jian Pan, Jun Li, Kechao Zhang, Xingcheng Lin, Zhehui Wang, Zhihuang Luo, et al. Quantum image processing and its application to edge detection: theory and experiment. *Physical Review X*, 7(3):031041, 2017.

[13] Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, Mar 2019.

[14] John Francis Canny. Finding edges and lines in images. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, 1983.

[15] FG Irwin et al. An isotropic 3x3 image gradient operator. *Presentation at Stanford AI Project*, 2014(02), 1968.

[16] Tamar Peli and David Malah. A study of edge detection algorithms. *Computer graphics and image processing*, 20(1):1–21, 1982.

[17] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[18] Hans Peter Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. PhD thesis, Stanford University, 1980.

[19] Chris Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[20] Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, pages 281–305. Interlaken, 1987.

[21] Stephen M Smith and J Michael Brady. Susan—a new approach to low level image processing. *International journal of computer vision*, 23(1):45–78, 1997.

[22] Minakshi Banerjee and Malay K Kundu. Handling of impreciseness in gray level corner detection using fuzzy set theoretic approach. *Applied Soft Computing*, 8(4):1680–1691, 2008.

[23] Pengao Xu, Zhenxing He, Tianhui Qiu, and Hongyang Ma. Quantum image processing algorithm using edge extraction based on kirsch operator. *Optics express*, 28(9):12508–12517, 2020.

[24] Yi Zhang, Kai Lu, and YingHui Gao. Qsobel: A novel quantum image edge extraction algorithm. *Science China Information Sciences*, 58, 12 2014.

[25] Ping Fan, Ri-Gui Zhou, Wen Wen Hu, and NaiHuan Jing. Quantum image edge extraction based on laplacian operator and zero-cross method. *Quantum Information Processing*, 18(1):1–23, 2019.

[26] Yulin Ma, Hongyang Ma, and Pengcheng Chu. Demonstration of quantum image edge extration enhancement through improved sobel operator. *IEEE Access*, 8:210277–210285, 2020.

[27] Xi-Wei Yao, Hengyan Wang, Zeyang Liao, Ming-Cheng Chen, Jian Pan, Jun Li, Kechao Zhang, Xingcheng Lin, Zhehui Wang, Zhihuang Luo, and et al. Quantum image processing and its application to edge detection: Theory and experiment. *Physical Review X*, 7(3), Sep 2017.

[28] Jae-Eun Park, Brian Quanz, Steve Wood, Heather Higgins, and Ray Harishankar. Practical application improvement to quantum svm: theory to practice. *arXiv preprint arXiv:2012.07725*, 2020.

[29] Dawid Kopczyk. Quantum machine learning for data scientists. *arXiv preprint arXiv:1804.10068*, 2018.

[30] Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.

[31] Seunghyeok Oh, Jaeho Choi, and Joongheon Kim. A tutorial on quantum convolutional neural networks (qcnn). In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 236–239. IEEE, 2020.

[32] Soronzonbold Otgonbaatar and Mihai Datcu. A quantum annealer for subset feature selection and the classification of hyperspectral images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14(5):7057–7065, 2021.

[33] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[34] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[36] Jeremy Howard and Sylvain Gugger. Fastai: A layered api for deep learning. *Information*, 11(2):108, Feb 2020.