On Classifying Images using Quantum Image Representation

Ankit Khandelwal^{1,2}, M Girish Chandra², Sayantan Pramanik³

¹Centre for High Energy Physics, Indian Institute of Science, Bengaluru, India

²TCS Research, India

³TCS Incubation, India

e-mail: ankit27.kh@gmail.com, m.gchandra@tcs.com, sayantan.pramanik@tcs.com

Abstract – In this paper, we consider different Quantum Image Representation Methods to encode images into quantum states and then use a Quantum Machine Learning pipeline to classify the images. We provide encouraging results on classifying benchmark datasets of grayscale and colour images using two different classifiers. We also test multi-class classification performance.

Keywords – Autoencoder, FRQI, Machine Learning, MCQI, Variational Classifier, Quantum Image Representation

I. INTRODUCTION

Quantum Image Representation (QIR) is a catch-all term for methods to encode an image as a quantum state. General data encoding methods [1] exist, such as Angle Encoding and Amplitude Encoding. But these do not take advantage of the specific structure of image data. Thus, unique representation methods have been invented to tackle image data and its representation as a quantum state using a quantum circuit.

A number of methods can be found in the literature. We have considered Flexible Representation of Quantum Images (FRQI) [2] for grayscale images and Multi-Channel Representation for Quantum Image (MCQI) [3] for colour images to encode the image and later use a quantum classifier on the encoded image to perform binary and multi-class classification.

We use a variational quantum classifier (VQC) and an autoencoder classifier (AC) to classify the images. We have obtained encouraging results using some classic benchmark datasets.

The paper is organised as follows. In Section II., we give a brief summary of the QIR methods used in this study. In Section III., we describe the quantum classifiers used for classifying the images. Section IV. details the datasets used in the paper. In Section V. we give the implementation details. Section VI.presents the classification results obtained from our simulations. In Section VII. we conclude and provide a few markers for future work.

II. Quantum Image Representation Methods

In this section we give a summary of the QIR methods used in the text. First some details of images:

- 1. Image dimension $= 2^n \times 2^n$ (n = 1 means a 2×2 image = 4 pixels)
- 2. Gray scale images \rightarrow Pixel values $\in [0, 255] \rightarrow$ In binary $\in \{0, 1\}^8$

Number of Matrix Elements $= 2^n \times 2^n$

3. Color images \rightarrow RGB, pixel values of each channel $\in [0, 255]$

Number of Matrix Elements $= 2^n \times 2^n \times 3$

A. FRQI

FRQI encodes the image data into a quantum state given by:

$$|I(\theta)\rangle = \frac{1}{2^{n}} \sum_{i=0}^{2^{2n}-1} [\cos(\theta_{i}) |0\rangle + \sin(\theta_{i}) |1\rangle] \otimes |i\rangle \qquad (1)$$
$$\theta_{i} \in \left[0, \frac{\pi}{2}\right], \theta = (\theta_{0}, \theta_{1}, \dots, \theta_{2^{2n}-1})$$

Here, $|0\rangle$ and $|1\rangle$ are single qubit computational basis states and $|i\rangle$, $i = 0, 1, ..., 2^{2n} - 1$ are 2n qubit computational basis states. The $\cos(\theta_i) |0\rangle + \sin(\theta_i) |1\rangle$ part is used to encode the pixel values while $|i\rangle$ encodes the pixel location.

The circuit to encode the image can be constructed using Hadamard (*H*) and Control Rotation, $C^{2n}R_y(2\theta)$, gates. It needs to be measured multiple times to get back the image from the state. The image retrieval process is probabilistic, and the result will depend on the number of shots used.

The number of qubits used in this representation is 2n + 1 with 2n qubits to encode the pixel location and 1 qubit to encode the pixel values. Pixel values are encoded as angles and are thus scaled to fit in the range $\left[0, \frac{\pi}{2}\right]$.

B. MCQI

MCQI representation uses 2n + 3 qubits to encode colour images while using 2n qubits to encode the pixel location like FRQI and the 3 remaining qubits to encode the pixel values of the RGB channels. This encoding is inspired by FRQI.

MCQI encodes the image into a quantum state given by:

$$|I(\theta)\rangle = \frac{1}{2^n + 1} \sum_{i=0}^{2^{2n} - 1} \left| C_{RGB}^i \right\rangle \otimes |i\rangle \tag{2}$$

The color information is encoded in:

$$\begin{aligned} \left| C_{RGB}^{i} \right\rangle &= \cos(\theta_{R}^{i}) \left| 000 \right\rangle + \cos(\theta_{G}^{i}) \left| 001 \right\rangle + \cos(\theta_{B}^{i}) \left| 010 \right\rangle \\ &+ \sin(\theta_{R}^{i}) \left| 100 \right\rangle + \sin(\theta_{G}^{i}) \left| 101 \right\rangle + \sin(\theta_{B}^{i}) \left| 110 \right\rangle \\ &+ \cos(0) \left| 011 \right\rangle + \sin(0) \left| 111 \right\rangle \end{aligned}$$

Colour encoding angle is applied to the R channel qubit using Control Rotation $(C^2R_y(2\theta))$ gates where it is controlled by the G and B channel qubits, and for each pixel, 3 $C^{2n}(C^2R_y(2\theta))$ gates are applied to encode the position and value information. The θ values are calculated from pixel values:

$$\theta = \cos^{-1} p$$

where, p is the pixel values $\in [0, 1]$. We get in this range by dividing the integer pixel values $\in [0, 255]$ by 255.

As before, the image retrieval process is probabilistic and depends on the number of shots.

III. Quantum Classifiers

We have used two different methods to classify the images in this paper. We describe these methods below.

A. Variational Quantum Classifier

To classify the images, we need some value to distinguish the two classes. The Z expectation value (ez) of the first qubit gives a natural split. We apply a variational ansatz on the quantum image and measure the expectation value of the colour qubit for FRQI and the R channel qubit for MCQI. The expectation value lies in the range [-1, 1], and thus a split can be formed such that:

$$\mathbf{class} = \begin{cases} -1 & \mathbf{if} \quad ez \le s \\ 1 & \mathbf{if} \quad ez > s \end{cases}$$
(3)

where *s* is the split set to 0 by default but can be trained to get optimal results.

1) Ansatz: We use a straightforward ansatz that consists of a general single-qubit rotation on each qubit and then a layer of CNOT gates, as shown in Fig 1. The number of layers of the ansatz is a hyperparameter. The number of parameters in the classifier = $3 \times N \times l$ where N = number of qubits and l = number of layers.



Fig. 1. Variational Classifier Ansatz with 4 qubits and 2 layers. U3 denotes a single qubit general rotation gate.

B. Autoencoder Classifier

An autoencoder is a tool to reduce the dimension of data. We use the autoencoder's quantum analogue [4] to compress the image state into a single qubit state. To use it as a classifier, we train the autoencoder to only compress the positive class. The autoencoder is trained by maximising the fidelity of the trash qubits with the zero state $(|0\rangle^{\otimes T})$ where *T* is the number of trash qubits. Once the autoencoder is trained on the positive class of the training data, we then use it to compress the validation data. We then measure the fidelity of the trash qubits again with the zero state and classify them depending on the resulting fidelity. The positive class should have higher fidelity values than the negative class. A single layer of the autoencoder is shown in Fig 2.



Fig. 2. Autoencoder with 1 data and 2 trash qubits. 'Rots' are general single-qubit rotation gates.



Fig. 3. Example of the two classes in the BAS data with n = 5.

IV. Dataset Details

We use two grayscale and one colour image datasets.

A. Bars and Stripes (BAS)

This dataset contains black and white images of dimension $2^n \times 2^n$. Example images are shown in Fig 3 for n = 5. These images are randomly generated. Horizontal stripes are one class, and vertical bars are another class. This dataset is used for binary classification.

B. MNIST

This is the famous dataset of handwritten digits [5]. We have used this for both binary (0 and 1) and multi-class (0, 1 and 2) classification. The original images are of 28×28 dimension, which first needs to be squared into $2^n \times 2^n$. Bilinear interpolation is used to transform the data for different *n*. There also exists another version of this data which contains 15 different corruption variations [6]. We also classify these corrupted datasets. Fig 4 shows the MNIST images, and Fig 5 shows the corrupted images.

C. 2×2 Color Images

This is randomly generated 2×2 colour image data for classification. The image has 4 pixels of random colours. For positive class we change the pixel values of the 4th pixel to (0,0,0). This makes the pixel black. The classification problem is then to differentiate between images with and without a black pixel. The pixel can also be modified to have dark shades instead of absolute 0 values. Fig 6 shows example images for different shades.

V. Implementation Details

We use PennyLane [7] to simulate the circuits. JAX [8] is combined with PennyLane as a high-performance simulator and to utilise the GPU. The optimisation library optax [9] is used for optimising the classifier. We use the Adam optimiser



Fig. 4. MNIST data. Row 1 and 2 show digits 0 and 1 respectively for $n \in [1, 5]$ from left to right.



Fig. 5. Corrupted MNIST data. Example of all 15 corruptions. n = 4.



Fig. 6. 2×2 Color images with different shades of the positive class.

[10] with a 0.1 step size for optimisation for 250 epochs unless otherwise mentioned. We use 5 layers of the VQC and 1 layer of the AC. We also use scikit-learn [11] for classical processing. All simulations were done with 'None' shots of the PennyLane device. This gives analytic results. We use different sizes of training datasets and a fixed 1000 data points for the validation data.

VI. Results

A. BAS

Table I shows accuracies on the validation set using different training dataset sizes and n values for the BAS data.

TABLE I
Validation Set Accuracy using the VQC and the AC on
FRQI representation for BAS data

Training Set Size	Classifier	n				
		1	2	3	4	
100	VQC	1.0	1.0	0.955	0.993	
	AC	1.0	0.818	0.820	0.889	
200	VQC	1.0	1.0	0.997	0.990	
	AC	1.0	0.929	0.881	0.732	
500	VQC	1.0	1.0	0.993	0.991	
	AC	1.0	0.858	0.842	0.866	

B. MNIST

Table II shows accuracies on the validation set using different training dataset sizes and *n* values for the MNIST data. For AC we have used 100 epochs for n < 3 with MNIST data. Table III shows accuracies on the validation set for the MNIST Corruption data for n = 4. Table IV shows the results for Multi-Class classification between 0, 1 and 2 digit images using the VQC. For this, we split the *ez* range into 3 parts.

C. 2×2 Color Images

Table V shows accuracy on the validation set using different training dataset sizes and shade values for the 2×2 colour image data. As expected the accuracy decreases as we increase the shade (see Fig 7). This is because, higher shade values imply smaller difference between the classes with a value of 255 implying no difference between the two classes.

TABLE II Validation Set Accuracy using the VQC and the AC on FRQI representation for MNIST data

Training Set Size	Classifier	n				
		1	2	3	4	
100	VQC	0.946	0.960	0.992	0.993	
	AC	0.922	0.905	0.917	0.811	
200	VQC	0.938	0.960	0.995	0.996	
	AC	0.904	0.877	0.793	0.691	
500	VQC	0.933	0.955	0.993	0.995	
	AC	0.904	0.950	0.925	0.826	

 TABLE III

 Validation Set Accuracy using the VQC and the AC on FRQI representation for MNIST Corruption data

Training Set Size	Classifier	Corruption				
		Shot Noise	Impulse Noise	Glass Blur	Motion Blur	Shear
500	VQC	0.996	0.998	0.994	0.990	0.997
	AC	0.918	0.929	0.980	0.918	0.858
		Scale	Rotate	Brightness	Translate	Stripe
500	VQC	0.988	0.993	0.993	0.965	0.993
	AC	0.982	0.960	0.977	0.880	0.973
		Fog	Spatter	Dotted Line	ZigZag	Canny Edges
500	VQC	0.988	0.995	0.997	0.992	0.989
	AC	0.561	0.904	0.956	0.955	0.481

TABLE IV
Mulit-Class Validation Set Accuracy using the VQC on
FRQI representation for MNIST data

Fraining Set Size	n					
	1	2	3	4		
100	0.588	0.838	0.798	0.824		
200	0.597	0.857	0.831	0.860		
500	0.603	0.807	0.773	0.765		

 TABLE V

 Validation Set Accuracy using the VQC and the AC on MCQI representation for 2 × 2 Color Image data

Shade	Training Set Size						
	100		200		500		
	VQC	AC	VQC	AC	VQC	AC	
0	0.970	0.845	0.975	0.839	0.995	0.854	
10	0.986	0.721	0.993	0.774	0.998	0.858	
20	0.966	0.813	0.989	0.673	0.992	0.808	
50	0.951	0.656	0.965	0.702	0.969	0.737	
100	0.907	0.539	0.897	0.523	0.918	0.512	
150	0.759	0.502	0.785	0.457	0.803	0.492	
200	0.647	0.484	0.677	0.516	0.635	0.479	
255	0.507	0.508	0.495	0.493	0.506	0.509	



Fig. 7. Validation accuracies as a function of shade with VQC on the left and AC on the right.

VII. CONCLUSIONS AND FUTURE WORK

Encouraging results on benchmark datasets have been obtained with both VQC and AC for binary and multi-class image classification. The work can be expanded to classify more involved images. The ansatz used for VQC was a simple layer. More research can be done to find a better ansatz that can provide improved performance while using a lower number of epochs. The effect of noise and shots on the performance can also be studied. Similarly, different autoencoder models, for example, denoising autoencoder, can be studied. Also, one can look into other Image Processing tasks like filtering on the representations/encodings.

References

- LaRose, R. & Coyle, B. Robust data encodings for quantum classifiers. *Phys. Rev. A* 102, 032420. https: //link.aps.org/doi/10.1103/PhysRevA.102. 032420 (3 Sept. 2020).
- Le, P. Q., Dong, F. & Hirota, K. A flexible representation of quantum images for polynomial preparation, image compression, and processing operations. *Quantum Information Processing* 10, 63–84. ISSN: 1573-1332. https://doi.org/10.1007/s11128-010-0177-y (Feb. 2011).
- Sun, B., Iliyasu, A. M., Yan, F., Dong, F. & Hirota, K. An RGB Multi-Channel Representation for Images on Quantum Computers. J. Adv. Comput. Intell. Intell. Informatics 17, 404–417 (2013).
- 4. Romero, J., Olson, J. P. & Aspuru-Guzik, A. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology* **2**, 045001 (2017).
- LeCun, Y., Cortes, C. & Burges, C. MNIST handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist 2 (2010).
- Mu, N. & Gilmer, J. MNIST-C: A Robustness Benchmark for Computer Vision. *arXiv preprint arXiv:1906.02337* (2019).
- Bergholm, V. et al. PennyLane: Automatic differentiation of hybrid quantum-classical computations 2018. https: //arxiv.org/abs/1811.04968.
- Bradbury, J. et al. JAX: composable transformations of Python+NumPy programs version 0.3.4. 2018. http:// github.com/google/jax.
- Hessel, M. et al. Optax: composable gradient transformation and optimisation, in JAX! version 0.1.2. 2022. http://github.com/deepmind/optax.
- Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization 2014. https://arxiv.org/abs/1412. 6980.
- 11. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).