MNIST Classification with Quantum Phase Estimation

Angelo Caponnetto

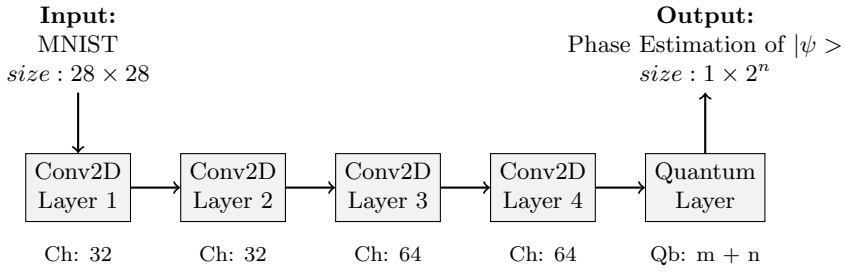
May 2025

1 Introduction

MNIST classification is a well-known benchmark that allows researchers to evaluate their models on a relatively simple task. In this work, the goal is to develop a hybrid neural network capable of classifying MNIST digits by leveraging Quantum Phase Estimation (QPE).

2 Neural Network

The neural network consists of two main components: a classical block and a quantum block. The classical block is responsible for extracting features from the input images and is composed of four convolutional layers.



The quantum block performs the phase estimation. It consists of a single layer with $m + n$ qubits: the first m qubits are used to construct the state $|\psi\rangle$, while the remaining n qubits are used to estimate its phase.

This measurement yields a probability vector in $\mathbb{R}^{2^n}$, where each component corresponds to a specific bit string among the 2^n possible outcomes (since n qubits yield 2^n bit strings).

Quantum Phase Estimation (QPE) involves selecting the bit string with the highest probability and converting it into a phase $\phi \in [0, 1]$, according to

Equation (1). The larger the value of m, the smaller the error in estimating ϕ, with an upper bound of $\epsilon_\phi < \frac{1}{2^m}$.

$$\phi = \sum_{\substack{k=1, \\ j \in \{0,1\}^n}}^{n} \frac{j}{2^k} \tag{1}$$

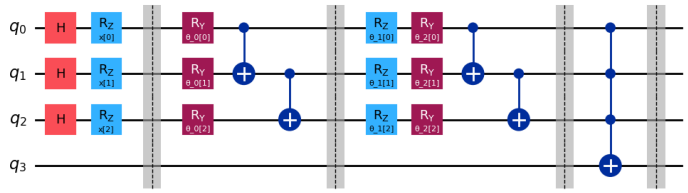
Starting from images of size 28×28, the classical block outputs scalar features x_i, with $i = 1, 2, \ldots, m-1$. Each x_i is then used as a rotation parameter in the quantum layer.

3 Quantum Layer

The quantum layer is made of three main blocks: *Init + Ansatz, U Gates* and *IFT* (inverse Fourier Transform).

3.1 Init + Ansatz

The block has the following structure (illustrated here with 4 qubits for the sake of simplicity):



The quantum block is composed of multiple R_z and R_y gates. The first rotation on each qubit uses the features produced by the classical convolutional layers, while the subsequent rotations use learnable parameters. The sequence $R_z + R_y +$ CNOT can be applied multiple times. Finally, a multi-controlled CNOT is applied to entangle the last qubit with all the others. The state of the last qubit can be expressed as:

$$|\psi >= a|1 > +b|0 > \tag{2}$$

where a and b depend on all the parameters.

3.2 U Gates

QPE requires a unitary gates such that:

$$U|\psi>= e^{-i2\pi\phi}|\psi> \tag{3}$$

$$U = V\Lambda V^{\dagger} \quad \text{where} \quad V = \begin{bmatrix} b & -a^* \\ a & b^* \end{bmatrix}, \quad \Lambda = \begin{bmatrix} e^{-i2\pi\phi} & 0 \\ 0 & e^{i2\pi\phi} \end{bmatrix} \tag{4}$$

Here ϕ is the phase we want to estimate.
By calculating, the unitary U becomes:

$$U = \begin{bmatrix} \cos(2\pi\phi) - ic\sin(2\pi\phi) & 2ia^*b\sin(2\pi\phi) \\ 2iab^*\sin(2\pi\phi) & \cos(2\pi\phi) + ic\sin(2\pi\phi) \end{bmatrix} \tag{5}$$

where $c = |b|^2 - |a|^2 = \frac{2^m-2}{2^m}$.

U can also be written as a composition of standard gates, $R_z + U_3$:

$$U = U_3R_z = \begin{bmatrix} e^{-i\alpha}\cos\theta & -e^{i(\alpha+\lambda)}\sin\theta \\ e^{i(\beta-\alpha)}\sin\theta & e^{i(\alpha+\beta+\lambda)}\cos\theta \end{bmatrix} \tag{6}$$

Comparing equations 5 and 6, all the angles can be written as:

$$\begin{aligned} &\alpha = \arctan\left[c\tan(2\pi\phi)\right], \quad \theta = \arccos\left(\frac{\cos(2\pi\phi)}{\cos\alpha}\right) \\ &\lambda = -\alpha + \arctan(\cot\phi), \quad \beta = -\lambda \end{aligned} \tag{7}$$

Knowing U allows us to construct the U-gate block. It consists of n qubits (2 in the figure below for simplicity).

Note that q_0 is the qubit where $|\psi>$ was prepared.

3.3 IFT

At the end of the circuit, a IFT gate is applied to measure the global phase ϕ. This gate produces a state (on the last m qubits) of the form:

$$|\psi_{IFT}>= \sum_x \alpha_x|x> \quad where \quad \alpha_x = \frac{1}{2^m}\sum_k e^{-i2\pi k(\phi-\phi_x)} \tag{8}$$

The probability of measuring the state $|x>$ is then:

$$P_x = \frac{1}{2^{2m}} \left(\frac{\sin(2^m \pi(\phi - \phi_x))}{\sin(\pi(\phi - \phi_x))} \right)^2 \tag{9}$$

In Equations (8) and (9), ϕ is the real (continuous) phase of $|\psi\rangle$, while ϕ_x is the quantized phase associated with the state $|x\rangle$, which is evaluated using Equation (1).

Any arbitrary function can be chosen for the global phase of $|\psi\rangle$. Here:

$$\phi = \sum_{weights} \omega_i^2 + \sum_{inputs} x_i^2 \tag{10}$$

where x_i are the features that come from convolutions, while ω_i are the learnable weights of the quantum layer.

4 Training

Once the neural network is built, we can focus on its training, which is organized in two steps: first, we search for different phases for each label we want to classify. This is achieved using the **Triplet Loss** implementation:

$$L_T = \sum_{inputs} \max\left[0, d(anchor, pos) - d(anchor, neg) + margin\right] + \lambda\Omega \tag{11}$$

where d is the KL-divergence and $\Omega = -\min_{i \neq j} ||c_i - c_j||^2$.

The training set is then organized into triples consisting of an *anchor*, a *positive*, and a *negative*. The anchor and positive belong to the same class, while the negative belongs to a different class.

Once these phases are obtained, we aim to maximize the probability of obtaining a specific phase for a specific digit. This is achieved using a **Cross-Entropy Loss**:

$$L_C = -\sum_i \ln(x_i[argmax(x_i)]) \tag{12}$$

5 Results

Different experiments have been conducted, with the neural network being trained to classify a different number of digits in each experiment. For each configuration:

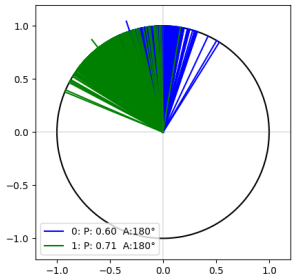
- $m = 7$
- $n\_iter = 5$

where n_iter is the number of repetition of the block $R_z + R_y + CNOT$. The number of qubits for phase estimation n changes in each configuration.

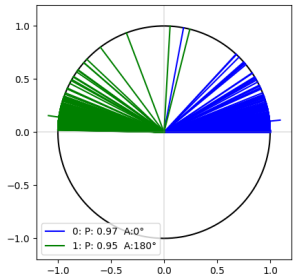
5.1 0 - 1 Digits

The model was trained to distinguish 0 and 1 digits. In this configuration:

- $training\_images = 1200$
- $n = 1 \quad \Rightarrow \quad \phi_x = \{0, 0.5\}$ allowed phases (from eq 1)
- **accuracy**: 0.998



(a) Classification before training.



(b) Classification after training.

Figure 1: Before training (a), all the digits have the highest probability ($P = 0.6$ for digit 0 and $P = 0.71$ for digit 1) corresponding to the same phase ($0.5 \Rightarrow 0.5 \times 360 = 180°$). After training (b), the highest probability for digit 0 corresponds to phase $0 \Rightarrow 0°$, while the highest probability for digit 1 corresponds to phase $0.5 \Rightarrow 180°$. Note that, to avoid representing all the rays at the same angles, the ones with the highest probability (A in both figures), angles were evaluated as $\sum_x p_x \phi_x$, where ϕ_x is the quantized angle related to the probability p_x.

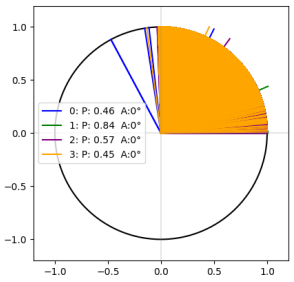
As shown in Figure 1, before training, all the images were classified with the same phase. After training, each digit has its own phase.

5.2 0 - 3 Digits

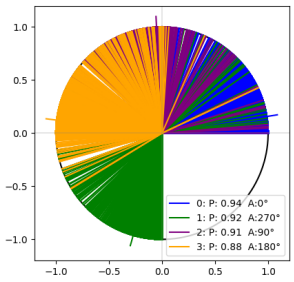
The model was trained to distinguish digits from 0 to 3. In this configuration:

- $training\_images = 1200$

- $n = 2 \quad \Rightarrow \quad \phi_x = \{0, 0.5, 0.25, 0.75\}$ allowed phases (from eq 1)
- **accuracy**: 0.998



(a) Classification before training.



(b) Classification after training.

Figure 2: Before training (a) all the images were classified with the same phase. After the training, every digit has its own phase. Drawn angle was evaluated as in Figure 1.

5.3 0 - 4 Digits

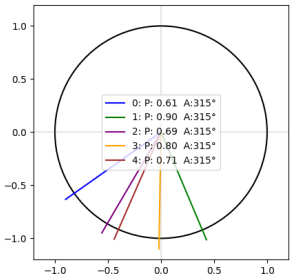
The model was trained to distinguish digits from 0 to 4. In this configuration:

- $training\_images = 6000$
- $n = 3$
- **accuracy**: 0.920

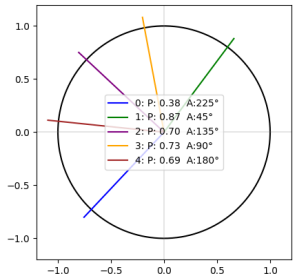
5.4 0 - 5 Digits

The model was trained to distinguish digits from 0 to 5. In this configuration:

- $training\_images = 15000$
- $n = 3$
- **accuracy**: 0.90



(a) Classification before training.



(b) Classification after training.

Figure 3: Before training (a) all the images were classified with the same phase. After the training, every digit has its own phase. Drawn angle was evaluated considering only the angle with highest probability.

5.5 0 - 7 Digits

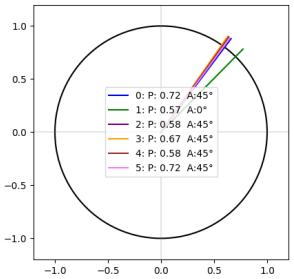
The model was trained to distinguish digits from 0 to 7. In this configuration:

- $training\_images = 33600$
- $n = 3$
- **accuracy**: 0.60

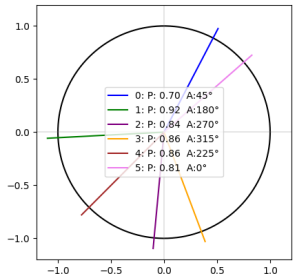
As shown in Figure 5, after training, not every digit has its own phase. In particular, digits 1 and 7 share the same phase. The model is unable to distinguish between these two classes, and this fact reduces the accuracy on the test set.

6 Conclusions

As reported in Section 5, the model achieves good results when managing a relatively low number of classes. However, as this number increases, the neural network is unable to assign different phases to each class, leading to an inevitable drop in performance. This is likely related to the triplet loss: to achieve good performance, it requires a high number of (hard) examples.

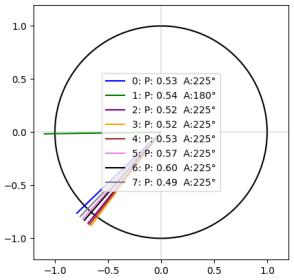


(a) Classification before training.

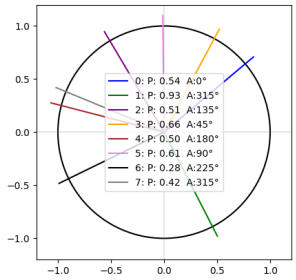


(b) Classification after training.

Figure 4: Before training (a) all the images were classified with the same phase. After the training, every digit has its own phase. Drawn angle was evaluated as in Figure 3.



(a) Classification before training.



(b) Classification after training.

Figure 5: Before training (a) all the images were classified with the same phase. After the training, not every digit has its own phase. Drawn angle was evaluated as in Figure 3.