WOMANIUM | QUANTUM

QML for Conspicuity Detection in Production Parts

Team “Q-WeldGuards”

Ahmed Aboukouta

Project Statement

The goal of this project is to design and implement a Quantum Machine Learning (QML) model to learn and classify Aluminum weld defects on the dataset produced by Bacioiua et al (see Ref [1].) This is a multi-class image classification problem.

- ❑ Implement a Variational Classifier with Pennylane to learn the binary parity function
- ❑ Implement a QML model to learn the sine function
- ❑ Implement a QML model to detect a defective production part (Aluminum welds)

‹ WOMANIUM | QUANTUM ›

PENNYLANE

Variational Classifier

- Goal: Train a quantum variational circuit to learn the n-bit parity function:

$$f : x \in \{0,1\}^{\otimes n} \rightarrow y = \begin{cases} 1 & \text{if odd number of 1's in } x \\ \textbf{-1} & \text{else.} \end{cases}$$

To do that, we follow this workflow:

1. **Encoding**: we encode the model's input into the initial state of the quantum circuit. In the present example, basis encoding lends itself most naturally, since the model's input is a bit string.

2. **Evolving**: we evolve the input state by applying a series of parameterized unitary operators, with the parameters acting as the weights of our quantum neural network.

3. **Measurement**: we perform a specific measurement on the final state of our circuit. The resulting value will be our model's prediction of the function's output for the specified encoded input in step 1.

4. **Optimizing**: we use an optimizer to update our parameters used in step 2 using a gradient-based method. In our present case, we use `NesterovMomentumOptimizer`, which implements the Nesterov Momentum optimization technique that improves the convergence speed of gradient descent algorithms by incorporating momentum.

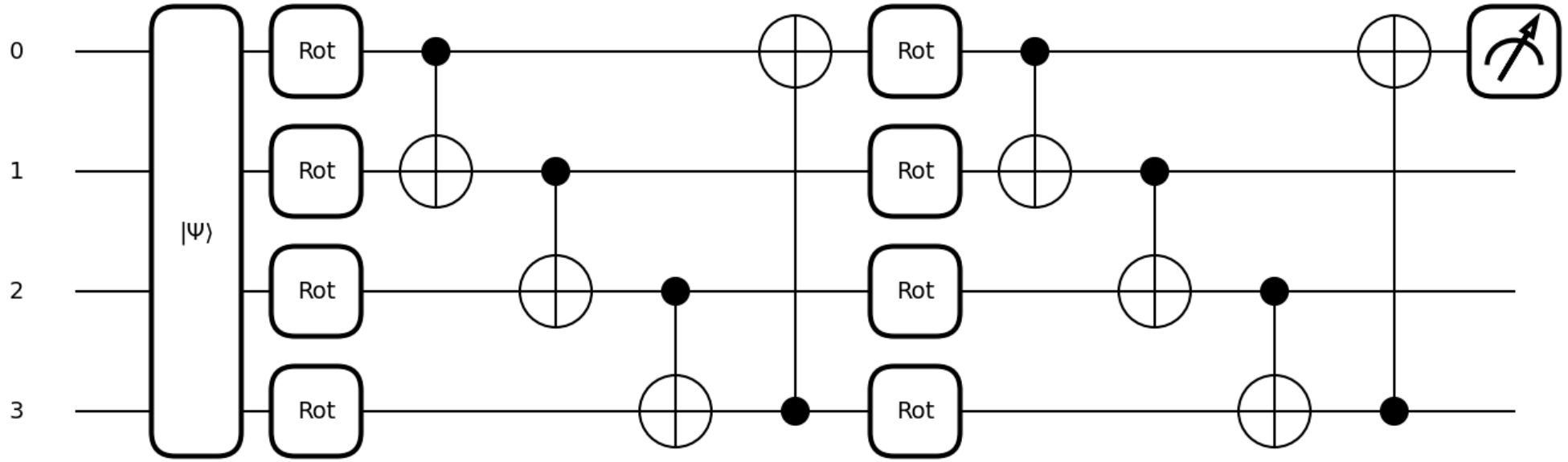
Variational Classifier

1. **Encoding**: we encode the model's input into the initial state of the quantum circuit. In the present example, basis encoding lends itself most naturally, since the model's input is a bit string.

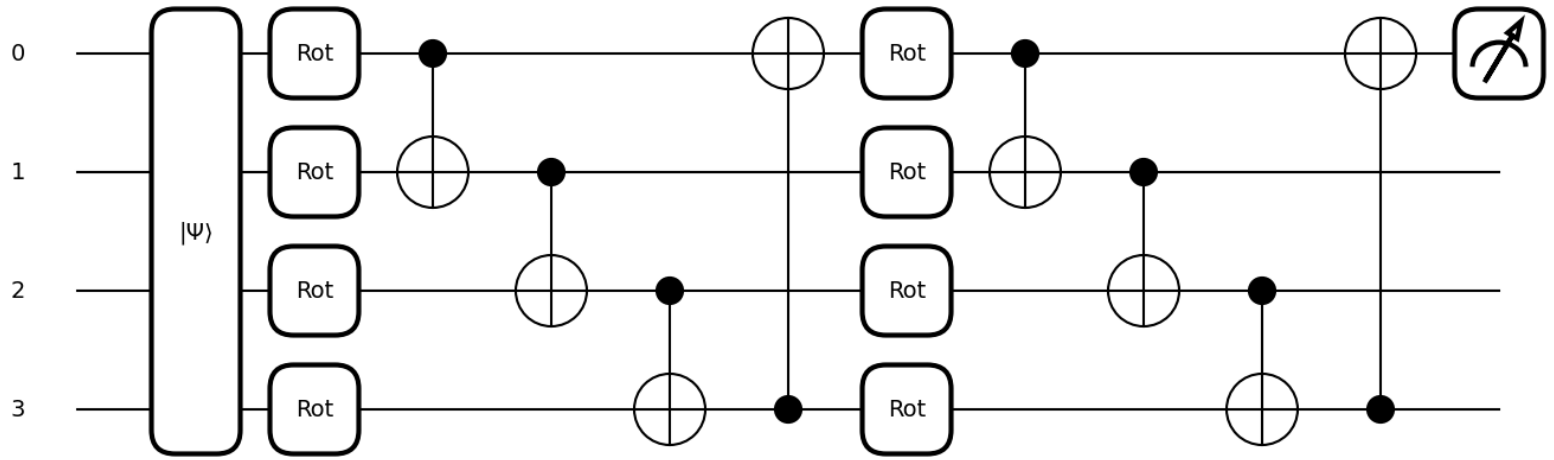
2. **Evolving**: we evolve the input state by applying a series of parameterized unitary operators, with the parameters acting as the weights of our quantum neural network.

3. **Measurement**: we perform a specific measurement on the final state of our circuit. The resulting value will be our model's prediction of the function's output for the specified encoded input in step 1.

4. **Optimizing**: we use an optimizer to update our parameters used in step 2 using a gradient-based method. In our present case, we use `NesterovMomentumOptimizer`, which implements the Nesterov Momentum optimization technique that improves the convergence speed of gradient descent algorithms by incorporating momentum.



Variational Classifier



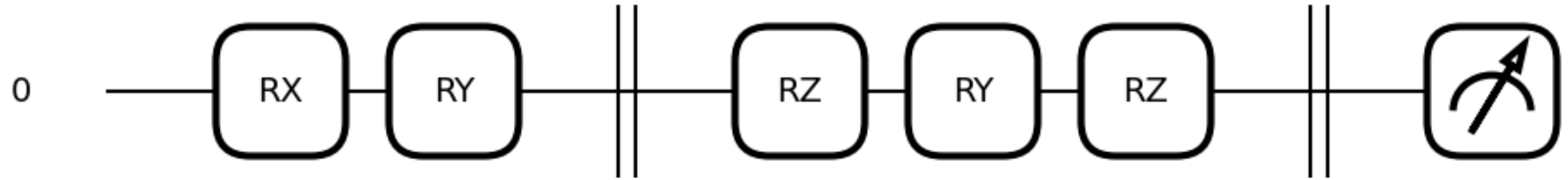

```
Iter:     1 | Cost: 2.3147651 | Accuracy: 0.5000000
Iter:     2 | Cost: 1.9664866 | Accuracy: 0.5000000
Iter:     3 | Cost: 1.9208589 | Accuracy: 0.5000000
Iter:     4 | Cost: 2.6276126 | Accuracy: 0.5000000
Iter:     5 | Cost: 0.9323119 | Accuracy: 0.6000000
Iter:     6 | Cost: 1.1903549 | Accuracy: 0.5000000
Iter:     7 | Cost: 2.0508989 | Accuracy: 0.4000000
Iter:     8 | Cost: 1.1275531 | Accuracy: 0.6000000
Iter:     9 | Cost: 1.1659803 | Accuracy: 0.6000000
Iter:    10 | Cost: 1.1349618 | Accuracy: 0.6000000
```

```
Iter:    50 | Cost: 0.8549226 | Accuracy: 0.6000000
Iter:    51 | Cost: 0.5136787 | Accuracy: 1.0000000
Iter:    52 | Cost: 0.2488031 | Accuracy: 1.0000000
Iter:    53 | Cost: 0.0461277 | Accuracy: 1.0000000
Iter:    54 | Cost: 0.0293518 | Accuracy: 1.0000000
Iter:    55 | Cost: 0.0205454 | Accuracy: 1.0000000
Iter:    56 | Cost: 0.0352514 | Accuracy: 1.0000000
Iter:    57 | Cost: 0.0576767 | Accuracy: 1.0000000
Iter:    58 | Cost: 0.0291305 | Accuracy: 1.0000000
Iter:    59 | Cost: 0.0127137 | Accuracy: 1.0000000
Iter:    60 | Cost: 0.0058108 | Accuracy: 1.0000000
```

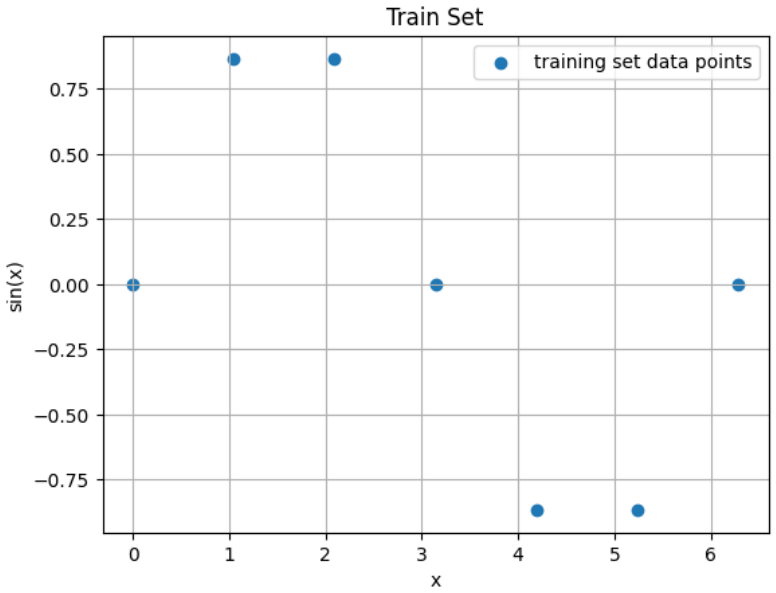
Learning the Sine Function

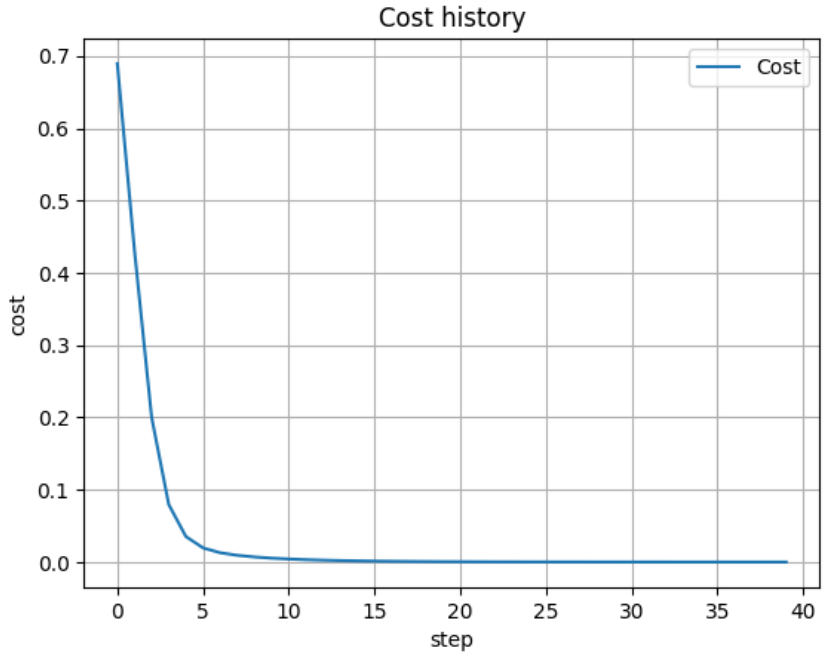
- Goal: train a quantum variational circuit to learn the sine function on the interval $x \in [0, 2\pi]$

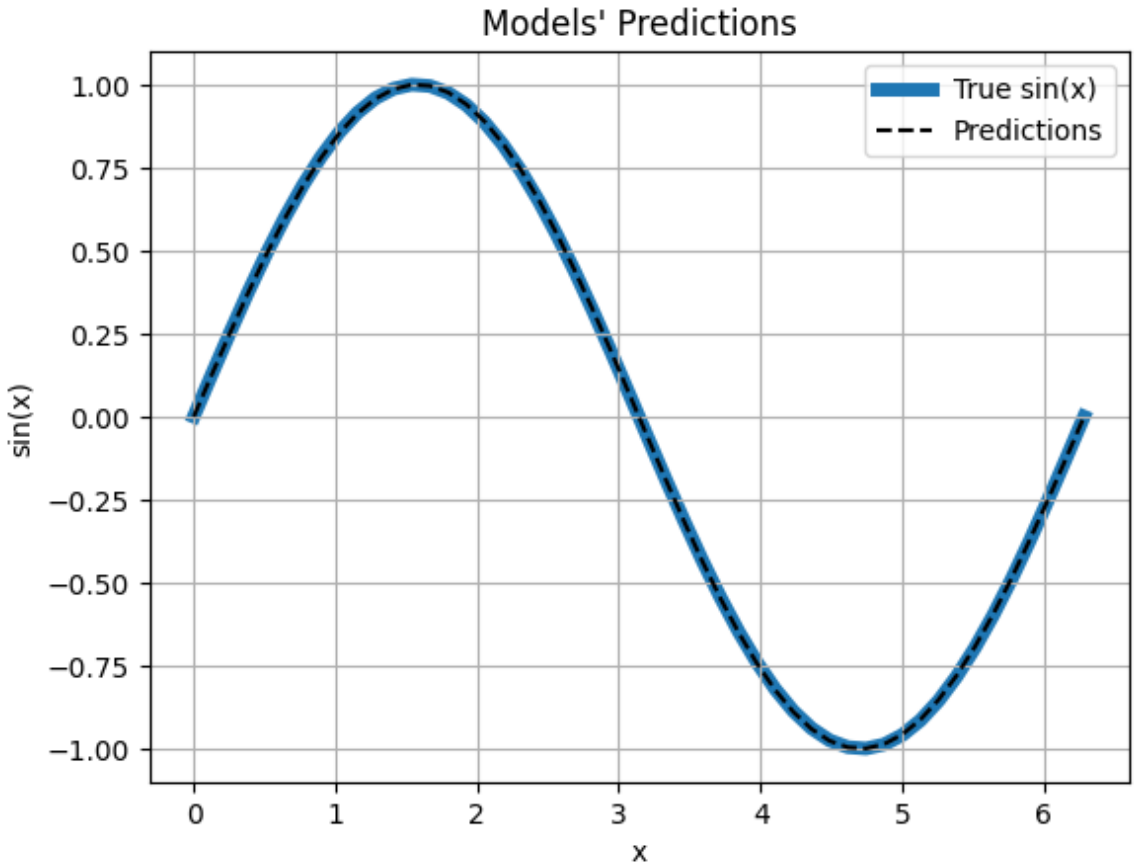
State Preparation and Ansatz:



$$U(\omega,\theta,\phi) = \begin{pmatrix} e^{-i(\phi+\omega)/2}\cos(\theta/2) & -e^{-i(\phi-\omega)/2}\sin(\theta/2) \\ e^{i(\phi-\omega)/2}\sin(\theta/2) & e^{i(\phi+\omega)/2}\cos(\theta/2) \end{pmatrix} = RZ(\omega)RY(\theta)RZ(\phi)$$


Train Set
training set data points
sin(x)
x
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
0
1
2
3
4
5
6


Cost history
Cost
cost
step
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
30
35
40


Models' Predictions
True sin(x)
Predictions
sin(x)
x
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
1
2
3
4
5
6

QML model to Detect Defective Aluminum Welds

- The goal of this project is to design and implement a Quantum Machine Learning (QML) model to classify Aluminum weld defects on the dataset produced by Bacioiua et al (see Ref [1].) I.e., this is a multi-class image classification task.
- The dataset consists of many high-resolution HDR images of Aluminum welding processes subject to different conditions such as to produce defective or non-defective welds. The defective welds are further categorized into 5 different defects as shown here.
- Our solution is to implement a Quanvolutional Neural Network as described introduced in Ref [2]. To evaluate this solution, we benchmarked its performance against a classical model ML of the same architecture

arXiv > quant-ph > arXiv:1904.04767

Quantum Physics

[Submitted on 9 Apr 2019]

Quanvolutional Neural Networks: Powering Image Recognition with Quantum Circuits

Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, Tristan Cook

Convolutional neural networks (CNNs) have rapidly risen in popularity for many machine learning applications, particularly in the field of image recognition. Much of the benefit generated from these networks comes from their ability to extract features from the data in a hierarchical manner. These features are extracted using various transformational layers, notably the convolutional layer which gives the model its name. In this work, we introduce a new type of transformational layer called a quantum convolution, or quanvolutional layer. Quanvolutional layers operate on input data by locally transforming the data using a number of random quantum circuits, in a way that is similar to the transformations performed by random convolutional filter layers. Provided these quantum transformations produce meaningful features for classification purposes, then the overall algorithm could be quite useful for near term quantum computing, because it requires small quantum circuits with little to no error correction. In this work, we empirically evaluated the potential benefit of these quantum transformations by comparing three types of models built on the MNIST dataset: CNNs, quantum convolutional neural networks (QNNs), and CNNs with additional non-linearities introduced. Our results showed that the QNN models had both higher test set accuracy as well as faster training compared to the purely classical CNNs.

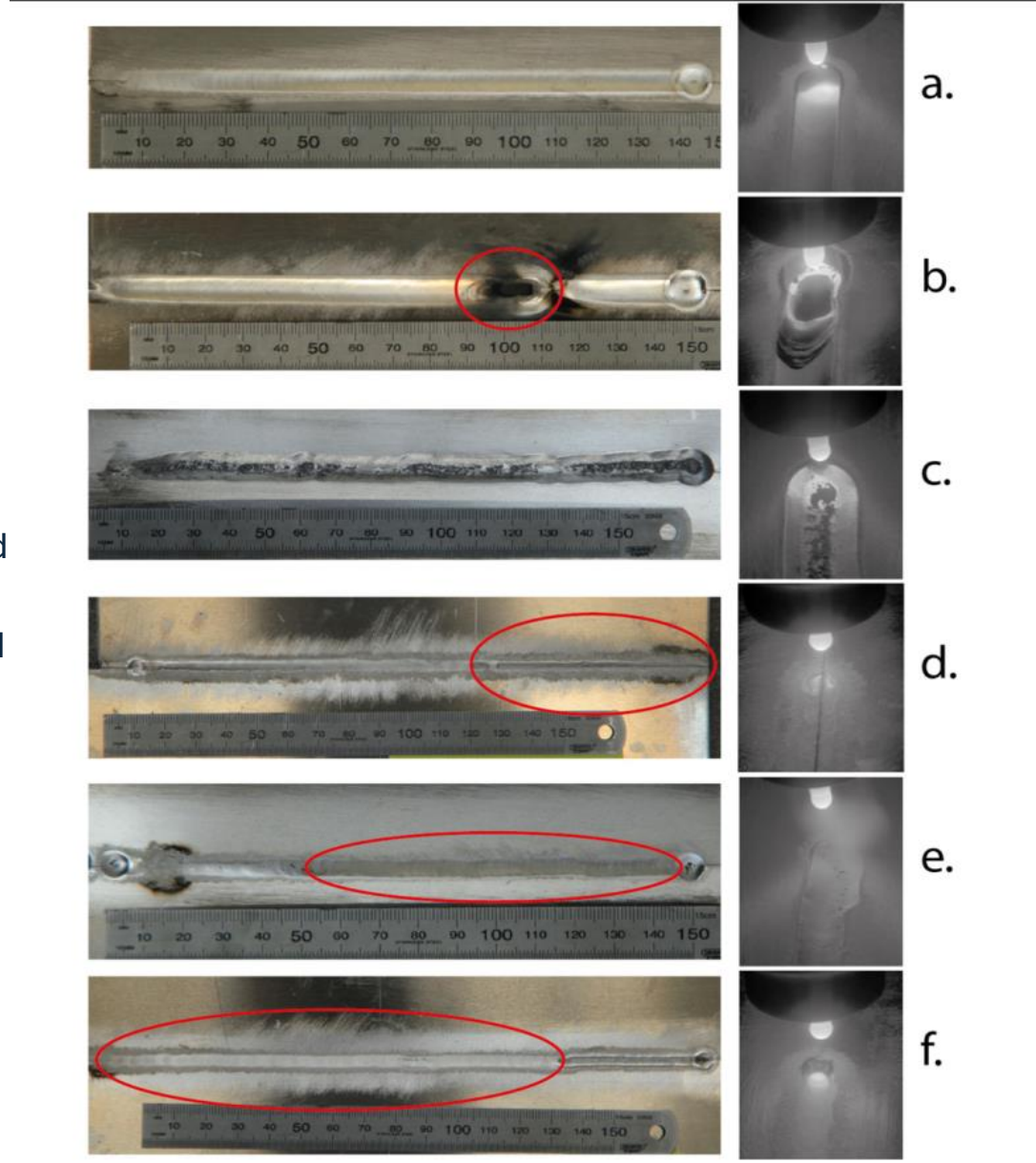
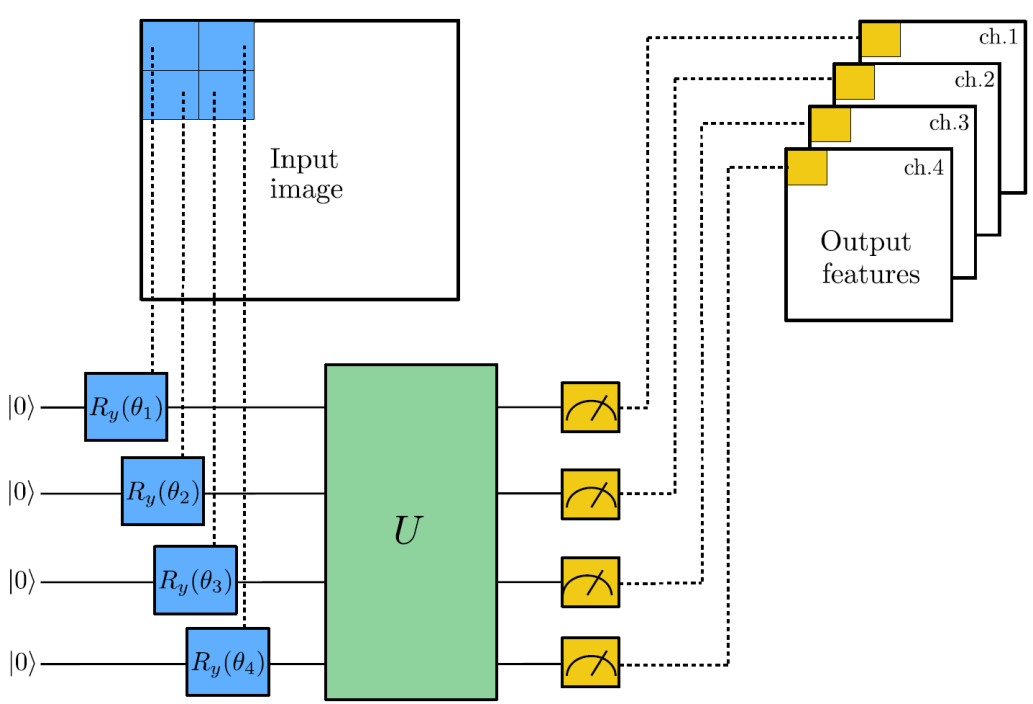
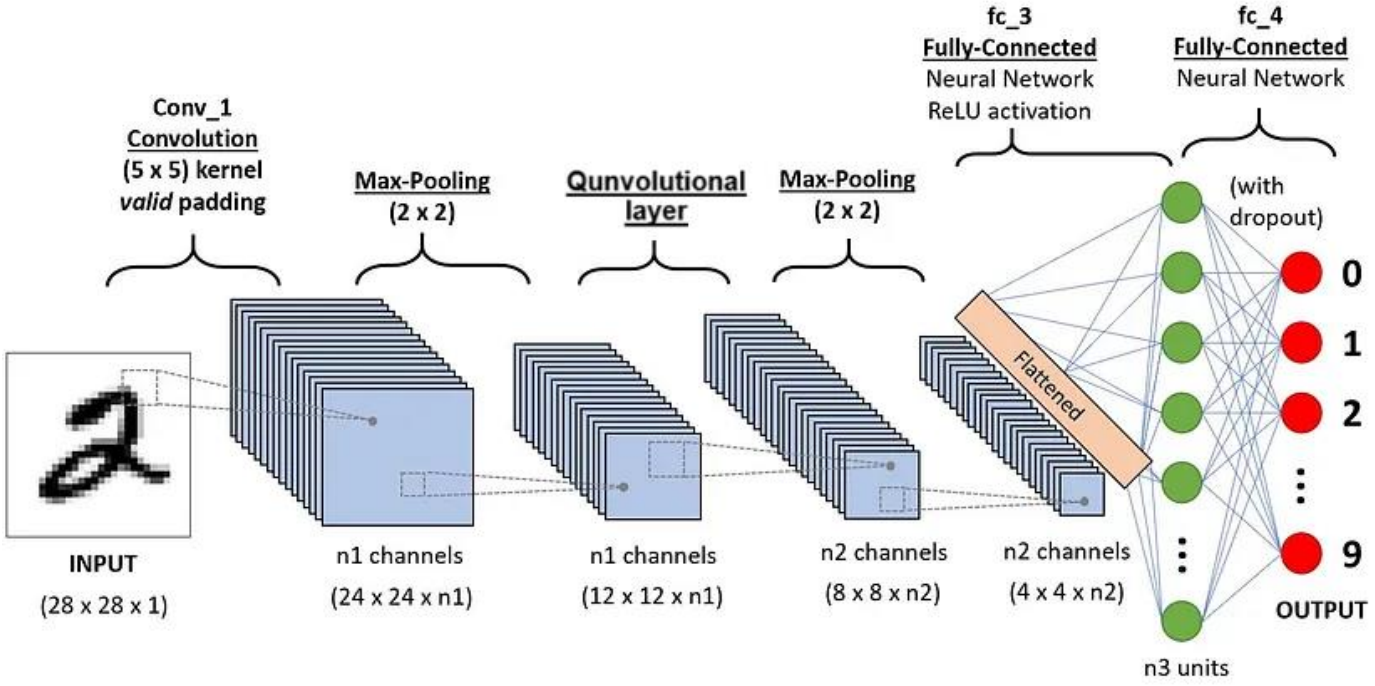


Fig. 5. Dataset samples of aluminium TIG welding. (a) good weld; (b) burn through; (c) contamination; (d) lack of fusion; (e) misalignment; (f) lack of penetration.

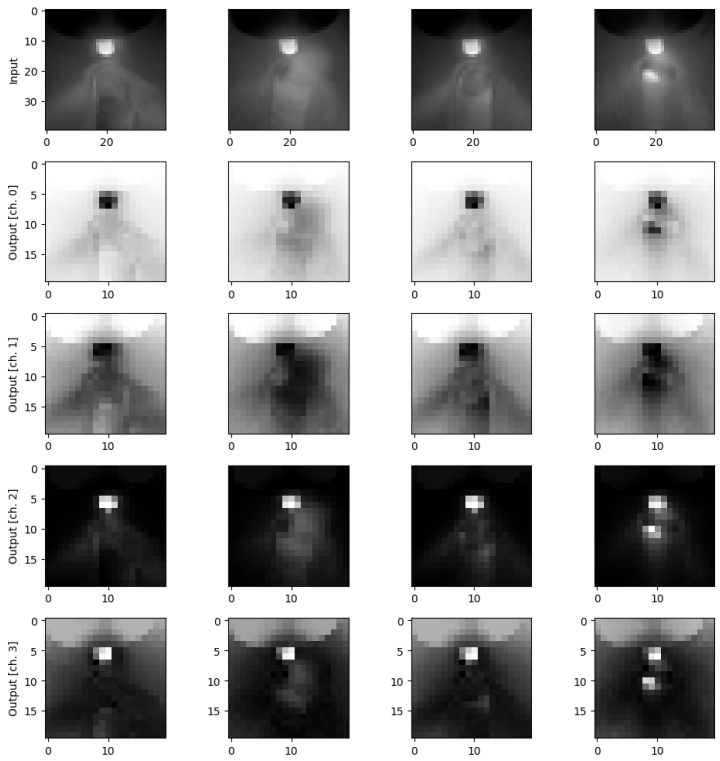
QML model to Detect Defective Aluminum Welds

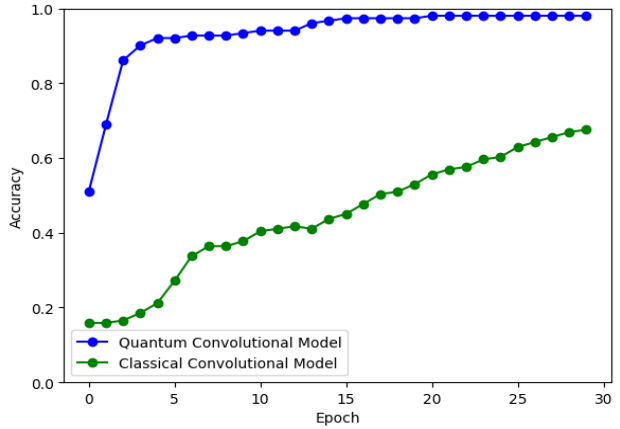


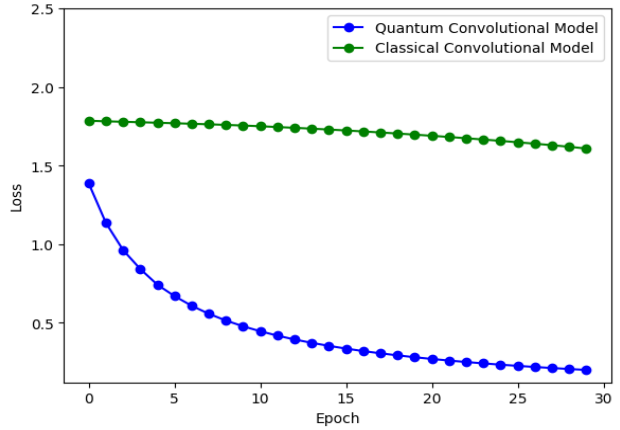


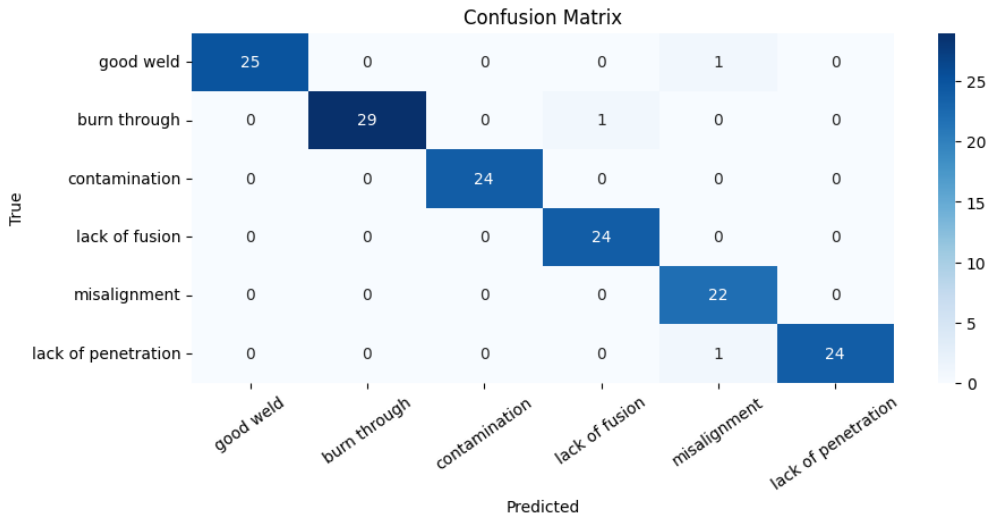
QML model to Detect Defective Aluminum Welds: Results

The Quantum Model learns quicker and is insensitive to noise.









Womanium Quantum+AI Project

References

[1] Bacioiu, D., Melton, G., Papaelias, M., & Shaw, R. (2019). Automated defect classification of Aluminium 5083 TIG welding using HDR camera and neural networks. Journal of manufacturing processes, 45, 603-613. DOI: https://doi.org/10.1016/j.jmapro.2019.07.020

[2] Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, Tristan Cook. “Quanvolutional Neural Networks: Powering Image Recognition with Quantum Circuits.” arXiv:1904.04767, 2019.