Quantum Convolutional Neural Networks (QCNN) for High Energy Physics Analysis at the LHC

Google Summer of Code @ ML4SCI Machine Learning for Science

Eraraya Ricardo Muten
Quantum Technology Lab
Institut Teknologi Bandung

MCQST Mentor: Dr. Maximilian Buser, Faculty of Physics, LMU

GSoC Project Mentors: Prof. Sergei V. Gleyzer, Dr. Emanuele Usai, Raphael Koh

github.com/eraraya-ricardo/GSoC-QCNN

Introduction

Dataset and Algorithm

Results and Discussion

Summary and Future Works

Outline

Introduction:

- Background
- Goal & Related Works

Dataset & Algorithm:

- Dataset
- Overall Architecture
- Quantum Convolutional Layer

Results & Discussion

Summary & Future Works

Introduction: Background

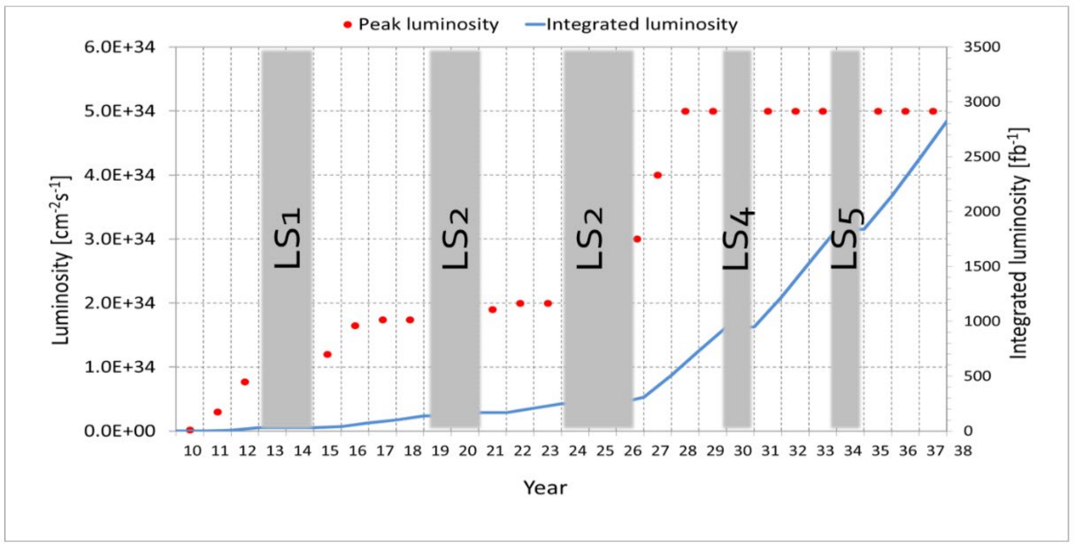
01

HL-LHC upgrades at CERN will **require enormous computing resources**[1]

02

Quantum computing has potential in improving performance of data processing and ML[2]

Can it improves HEP data analysis?



Projected LHC performance through 2038,
more luminosity = produce more data[1]

[1] Burkhard Schmidt 2016 *J. Phys.: Conf. Ser.* **706** 022002.
[2] Biamonte J, et al. *Nature* 2017;**549**.

Introduction

Project's Goal

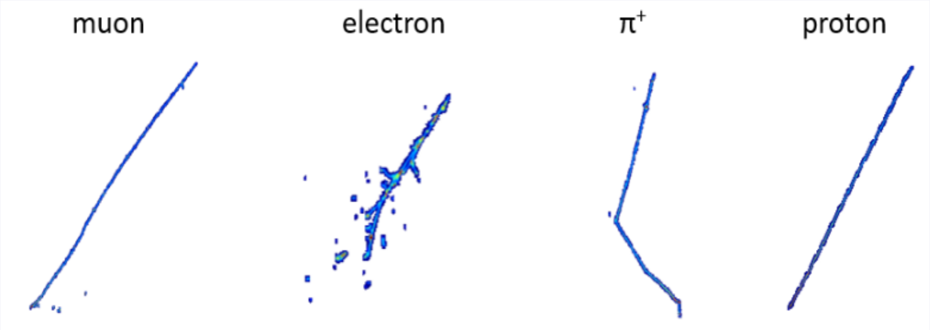
Explore quantum machine learning method performance on particle identification from ECAL image data, compared to the classical method.

Related Works

QCNN on other datasets:

- MNIST dataset [3, 4] and
- simulated particle trajectory images [5]

with performance comparable to the classical model.



Example images of simulated particle used in [5]

[3] Henderson, M., et al. *Quantum Mach. Intell*. **2**, 2 (2020).
[4] Oh, S., et al. 2020. **arXiv:2009.09423**
[5] Chen, S.Y., et al. 2020. **arXiv:2012.12177**

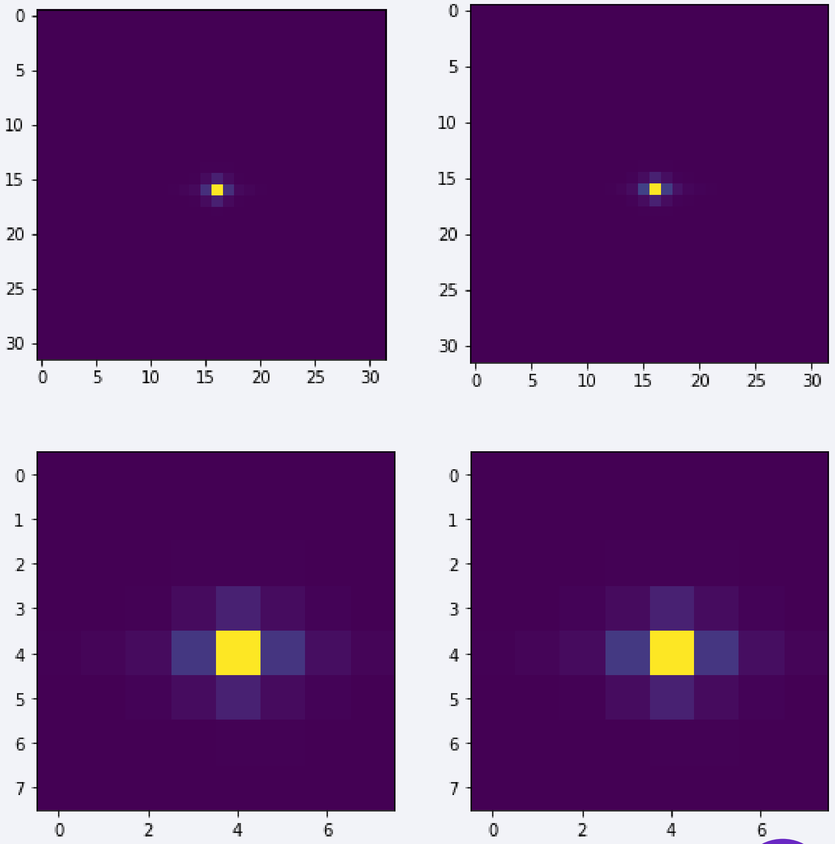
Dataset

Images of electrons and photons captured by ECAL (Electromagnetic Calorimeter).

- A pixel = a detector cell
- Pixel's intensity = energy measured in that cell
- The dataset contains 32x32 images but cropped into 8x8
- The every pixel in the dataset is standard-scaled

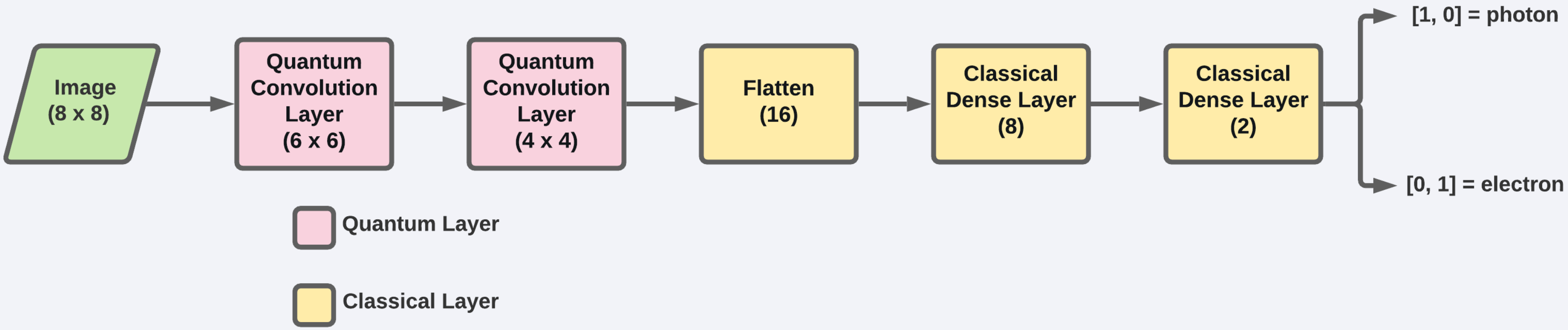
$$x' = \frac{x - \mu}{\sigma}$$

Averages of image samples from the dataset.
Left: Photon, Right: Electron.
Top: Full 32x32, Bottom: After cropping 8x8.



Algorithm

Overall Architecture of QCNN



Model creation & training: TensorFlow Quantum [6]
Quantum Simulator: Google's Cirq [7] noiseless analytic simulator

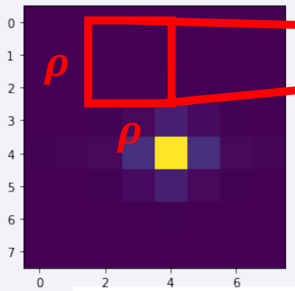
Quantum Convolution Layer:

- Transform input features to another feature map via a convolution-like operation
- The transformation is done by a trainable variational ansatz instead of a classical filter

[6] Broughton, M., et al. 2020. **arXiv:2003.02989**
[7] Cirq Developers. **doi: 10.5281/zenodo.4586899**.

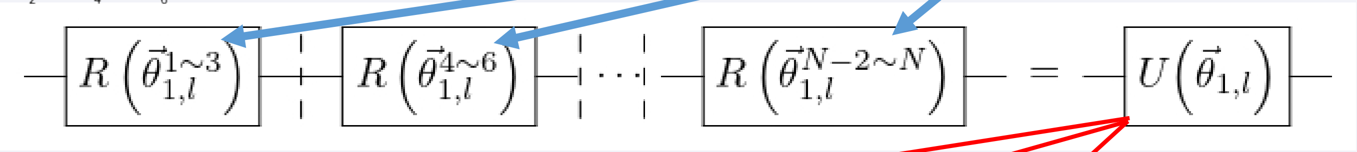
Quantum Convolution Layer

- The variational quantum circuit used for the quantum convolution layers is the data re-uploading circuit [8]
- The number of layers and qubits can be increased (ring of CZ gate will be used if 2 or more qubits are used)
- The Z expectation of the last qubit is measured

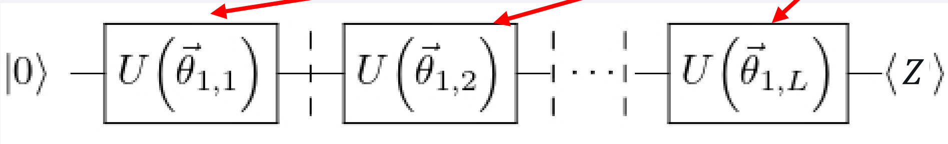


$$\vec{x} = (x^1, x^2, \ldots, x^{\zeta}), \qquad \rho \times \rho = 3 \times 3$$

$$\theta^{n}_{q,l} = w^{n}_{q,l} x^{n} + b^{n}_{q,l} \rightarrow \vec{\theta}^{\mathrm{n\sim n+2}}_{\mathrm{q,l}} = \left(\theta^{n}_{q,l}, \theta^{n+1}_{q,l}, \theta^{n+2}_{q,l}\right)$$



$$R(\phi, \theta, \omega) = RZ(\phi) RY(\theta) RX(\omega)$$

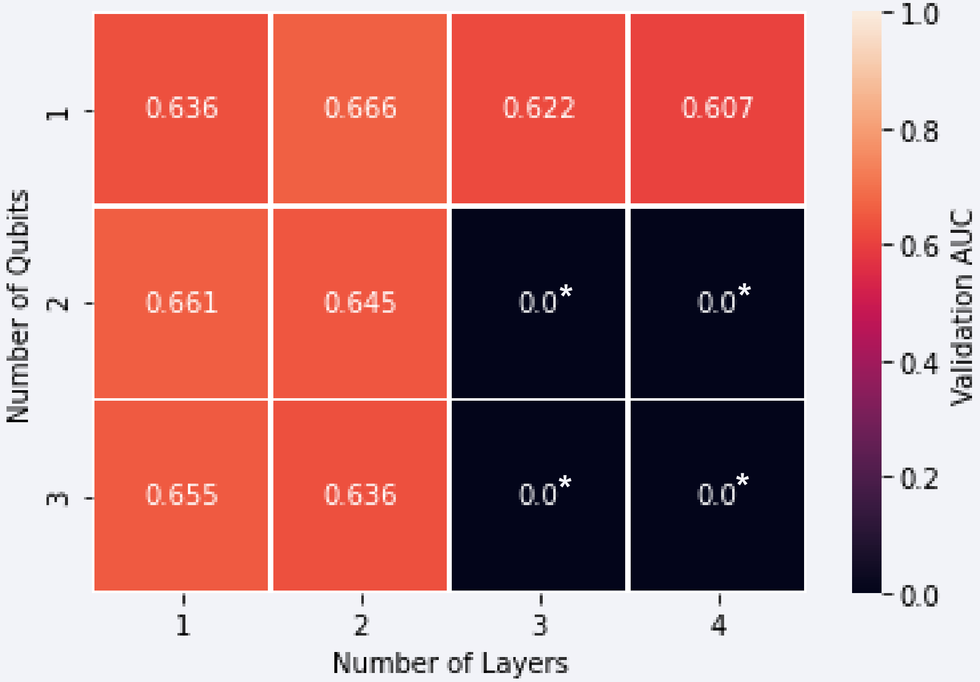


[8] Pérez-Salinas, A., et al. (2020). *Quantum*, 4, 226.

Results & Discussion

QCNN Validation AUC (8500 training samples, 1500 validation samples)

AUC = Area under the ROC (Receiver Operating Characteristic) Curve

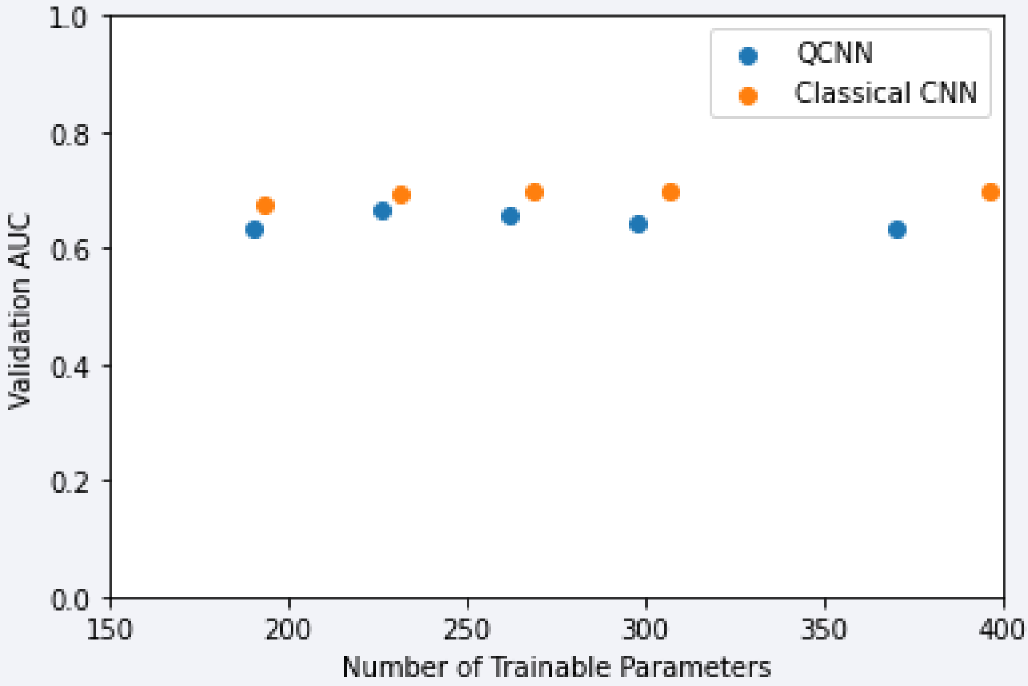


- Original paper shows more layers = higher accuracy as a classifier for 1-D dataset
- Not the case if used as convolution filter for 2-D dataset
- Choosing the right variational ansatz is not so trivial

***0.0 (black) = not tested**

Results & Discussion

Validation AUC of QCNN vs Classical CNN
(8500 training samples, 1500 validation samples)



With 423300 training samples & 74700 testing samples:

- QCNN (190 parameters): **0.730**
- Classical CNN (193 parameters): **0.738**

- With similar number of parameters, the QCNN is a little bit worse than Classical CNN

Both QCNN and Classical CNN:

- Increasing the number of training samples increases performance
- Only increasing the number of trainable parameters not necessarily increases performance (overfitting)

Summary

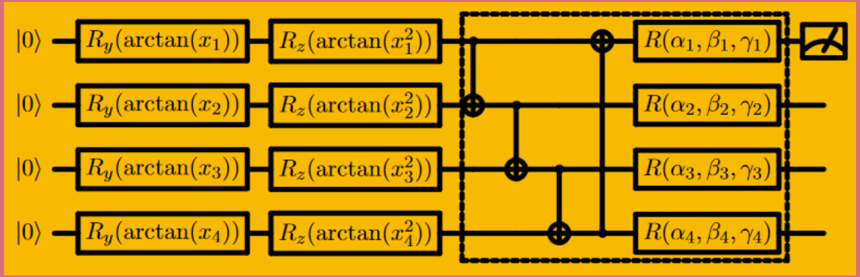
Choosing variational ansatz is not trivial

A good ansatz for one type of task doesn't always transfer to another different task

Using data re-uploading circuit as convolution filter, classical model is still a little bit better

Future Works

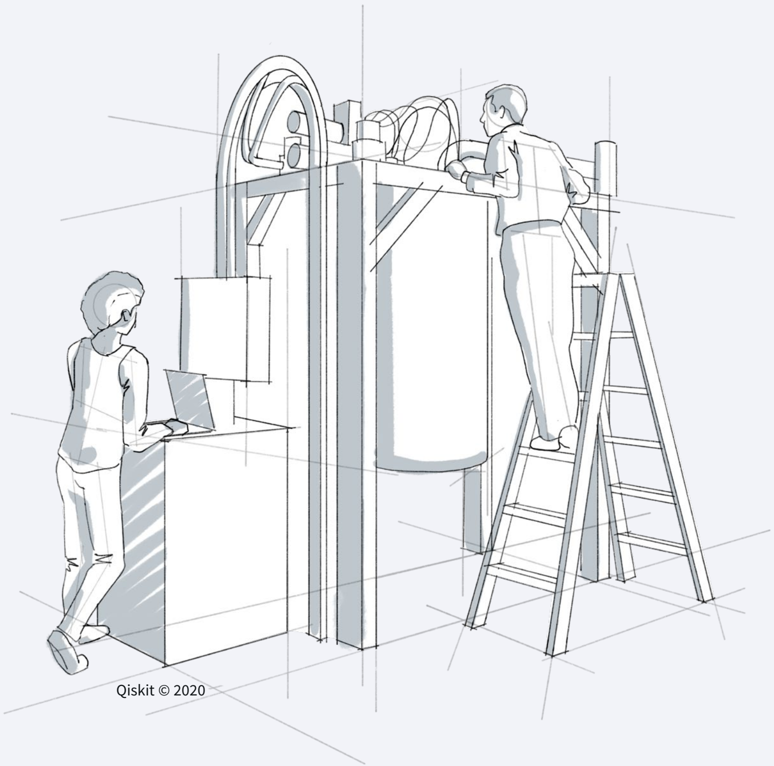
Try different ansatz, for example the one suggested in [5]



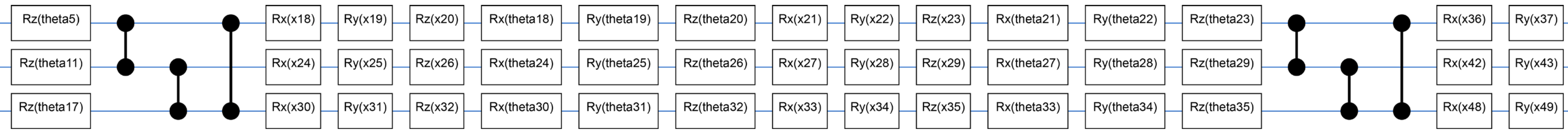
Try using several filters on every quantum convolution layer similar to what usually is done by classical CNN

[5] Chen, S.Y., et al. 2020. **arXiv:2012.12177**

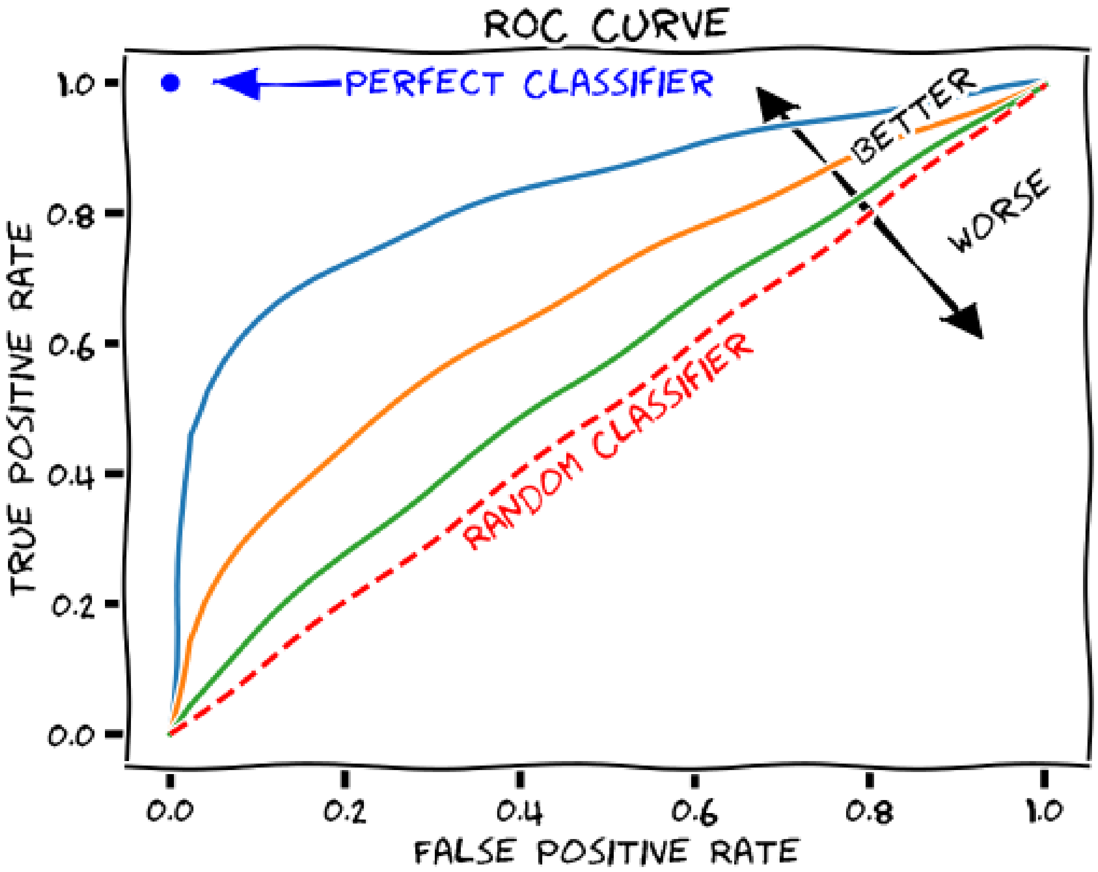
Thank You!
Any Questions?



APPENDIX



Rz(theta5)
Rz(theta11)
Rz(theta17)
Rx(x18)
Ry(x19)
Rz(x20)
Rx(theta18)
Ry(theta19)
Rz(theta20)
Rx(x21)
Ry(x22)
Rz(x23)
Rx(theta21)
Ry(theta22)
Rz(theta23)
Rx(x36)
Ry(x37)
Rx(x24)
Ry(x25)
Rz(x26)
Rx(theta24)
Ry(theta25)
Rz(theta26)
Rx(x27)
Ry(x28)
Rz(x29)
Rx(theta27)
Ry(theta28)
Rz(theta29)
Rx(x42)
Ry(x43)
Rx(x30)
Ry(x31)
Rz(x32)
Rx(theta30)
Ry(theta31)
Rz(theta32)
Rx(x33)
Ry(x34)
Rz(x35)
Rx(theta33)
Ry(theta34)
Rz(theta35)
Rx(x48)
Ry(x49)


ROC CURVE
PERFECT CLASSIFIER
BETTER
WORSE
RANDOM CLASSIFIER
TRUE POSITIVE RATE
FALSE POSITIVE RATE
0.0
0.2
0.4
0.6
0.8
1.0

QCNN Settings

- 10k samples with 15% for test samples
- 200 epochs, 128 batch size
- *varying* qubits, *varying* layers
- filter size = [3, 3], stride = [1, 1]
- followed by classical head [8, 2] with activation [relu, softmax]
- classical preprocessing = crop to 8x8 + standard scaling
- optimizer: Adam, lr = 0.001 with decay, $\beta1 = 0.9$, $\beta2 = 0.999$, $\varepsilon = 1e\text{-}07$
- cross-entropy loss

- Simulator: noiseless
- Gradient calculation: analytic

Classical CNN Settings

- 10k samples with 15% for test samples
- 200 epochs, 128 batch size
- filter size = [3, 3], stride = [1, 1]
- conv activation = [relu, relu]
- use_bias = [True, True]
- followed by classical head [8, 2] with activation [relu, softmax]
- classical preprocessing = crop to 8x8 + standard scaling
- optimizer: Adam, lr = 0.001 with decay, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ε = 1e-07
- cross-entropy loss

LR Decay

```
lr = 1e-3
    if epoch > 180:
        lr *= 0.5e-3
    elif epoch > 160:
        lr *= 1e-3
    elif epoch > 120:
        lr *= 1e-2
    elif epoch > 80:
        lr *= 1e-1
```