Can quantum machine learning really outperform classical models on real-world datasets?

High-Performance
Computing Center
Stuttgart

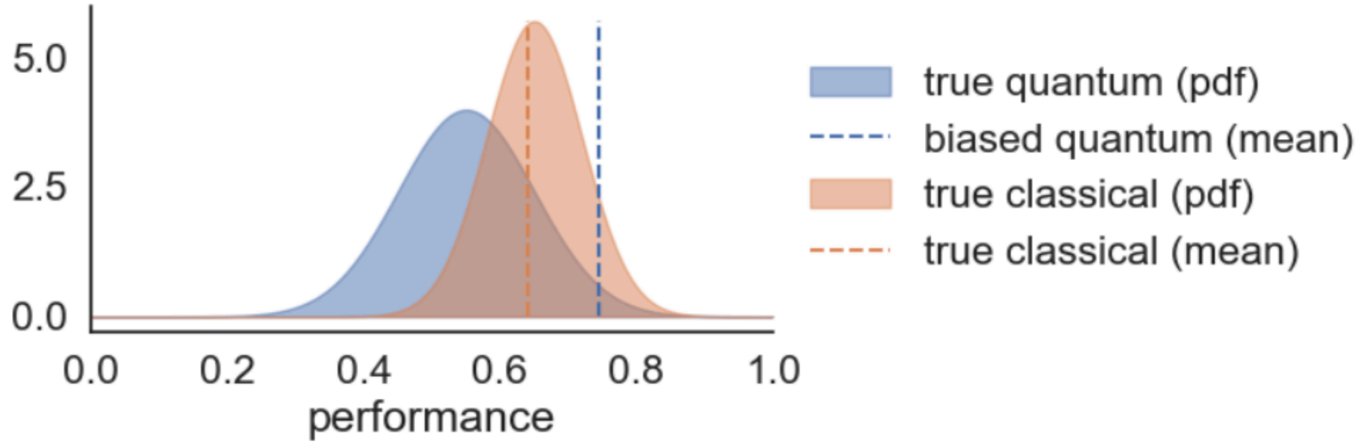
Dr. Ashley Smith

Collaborators: Dr. Qifeng Pan, Prof. Gerhard Hellstern

Background

As a research community are we all cherry picking?

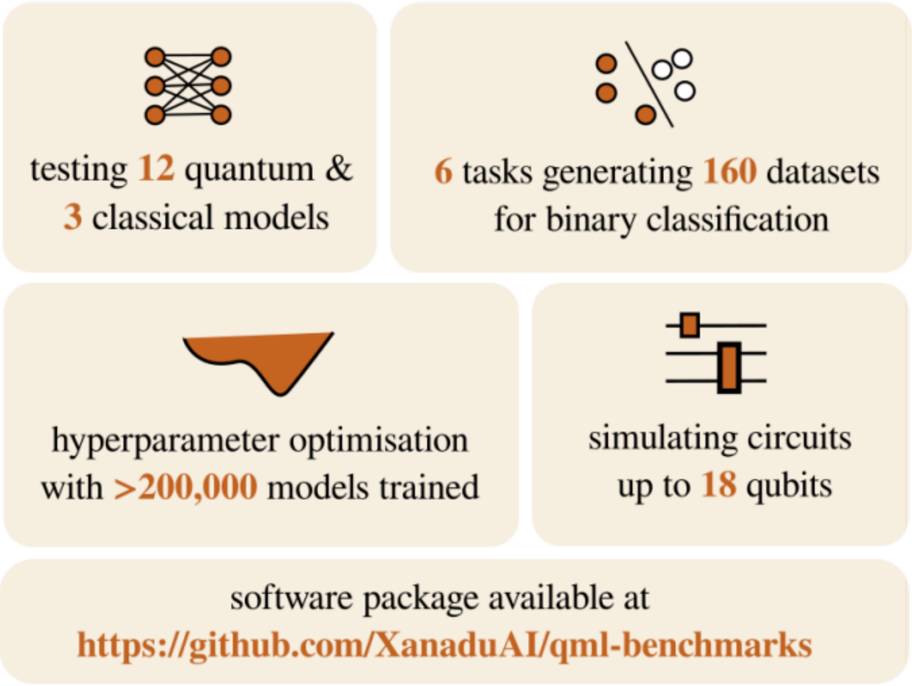
- Certainly many QML studies claim a quantum advantage.
  Yet it can be hard to achieve in practice.



Source: https://arxiv.org/pdf/2403.07059

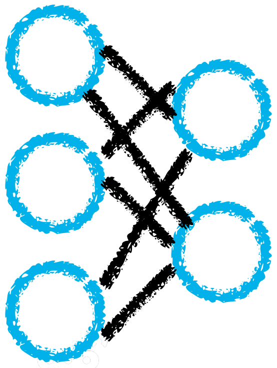
Classical ML is hard to beat

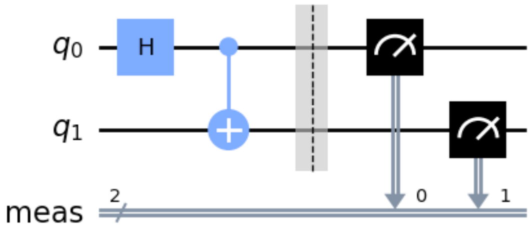
- Pennylane **meta**-study was disappointing for QML proponents.
  - Reproduced representative assortment of QML models from literature.
  - „overall, out-of-the-box classical machine learning models outperform the quantum classifiers…
  - removing entanglement from a quantum model often results in as good or better performance, suggesting that “quantumness” may not be the crucial ingredient.“



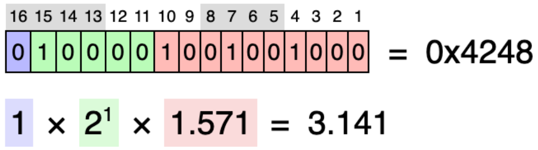
“Quantum Data“

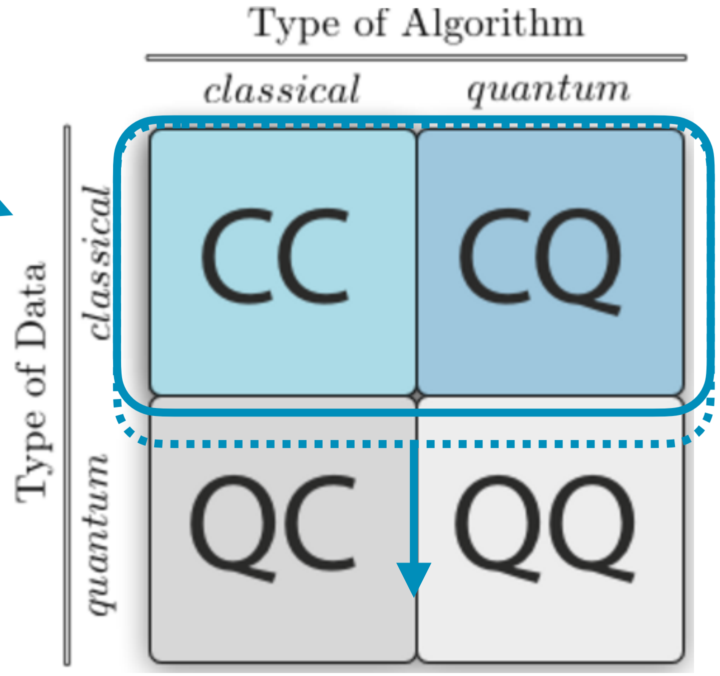
- Try to explore more broadly than swapping classical models for quantum equivalents.

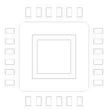




16-bit (half)







$$|0\rangle, |1\rangle$$
$$|00\rangle, |01\rangle, |10\rangle, |11\rangle$$
$$|000\rangle, |100\rangle, |010\rangle, |001\rangle, |110\rangle, |011\rangle, |101\rangle, |111\rangle$$
$$\ldots$$

Sources: https://en.wikipedia.org/wiki/Quantum_machine_learning + https://evanw.github.io/float-toy/

Power of data in quantum machine learning

Paper Ideas

- Classically hard problems can still be competitive with quantum models when one considers the affect of the available data.
- Projected Quantum Kernel (PQK) can provide a small advantage.
- Methodology for constructing artificial quantum advantages.

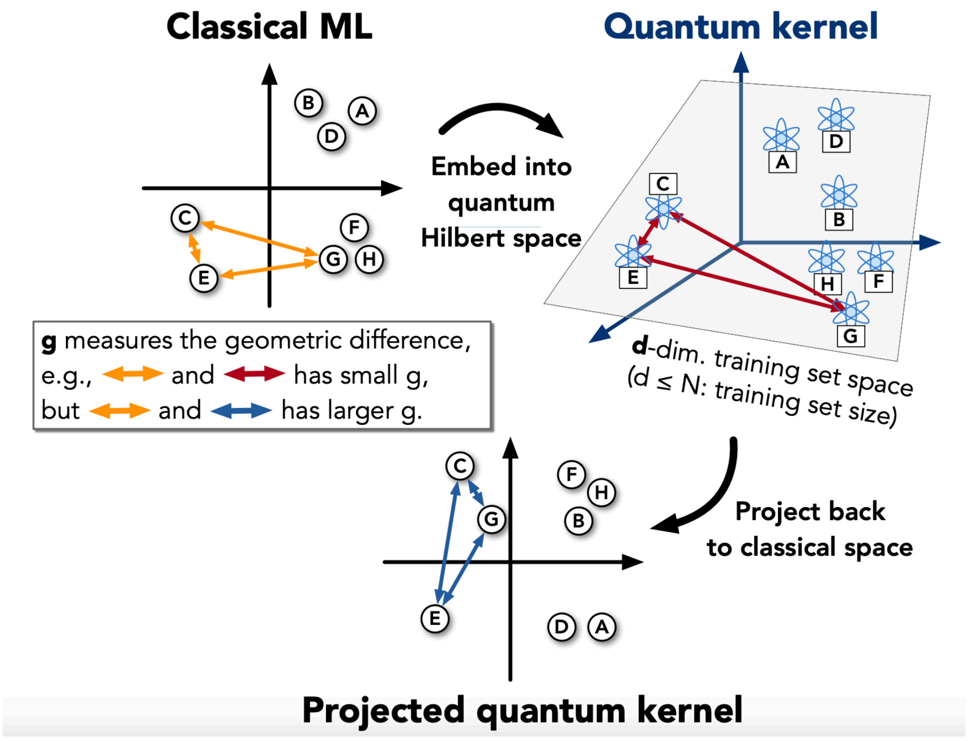
$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

$$k^{\mathrm{Q}}(x_i, x_j) = |\langle x_i | x_j \rangle|^2.$$

$$k^{\mathrm{PQ}}(x_i, x_j) = \exp\left(-\gamma \sum_k \sum_{P \in \{X,Y,Z\}} \left(\mathrm{Tr}(P\rho(x_i)_k) - \mathrm{Tr}(P\rho(x_j)_k)\right)^2\right),$$

Projected Quantum Kernel/Quantum Shadow

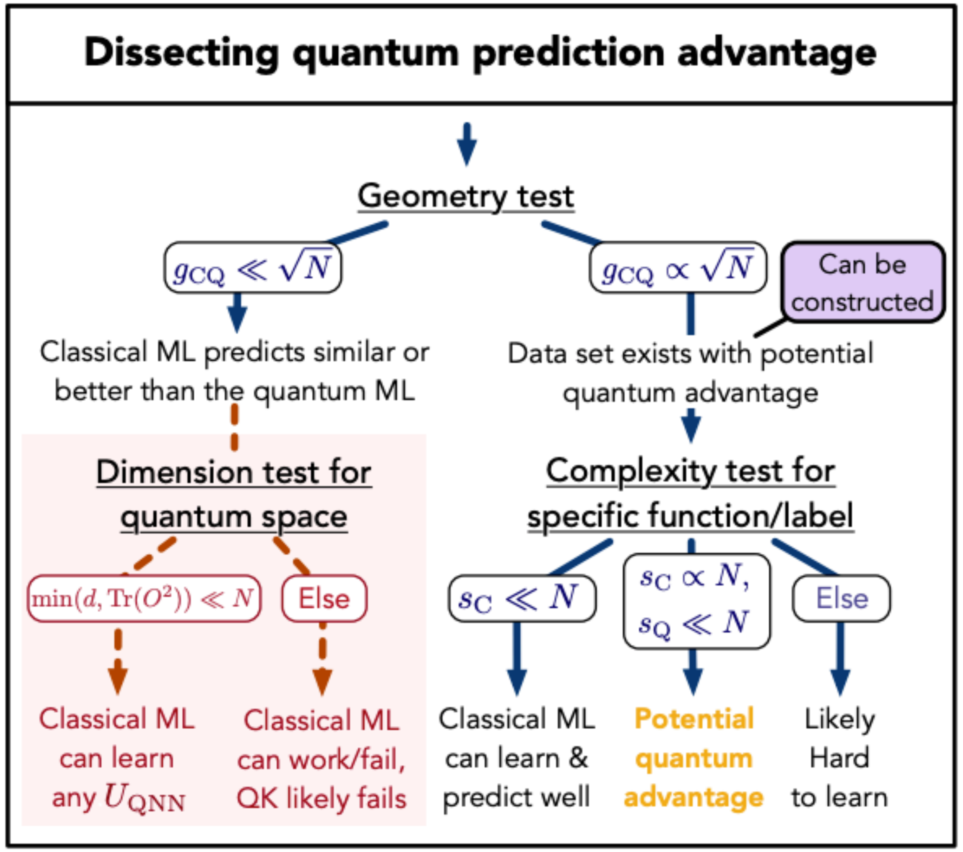
- **PQK Intuition** “Best of both worlds“:
  - Quantum feature space captures richer representations.
  - Alleviates problems associated with large Hilbert spaces e.g. all the inner products vanishing because the space is too big.



Source:
Power of data in quantum machine learning https://arxiv.org/pdf/2011.01938

Geometric Difference

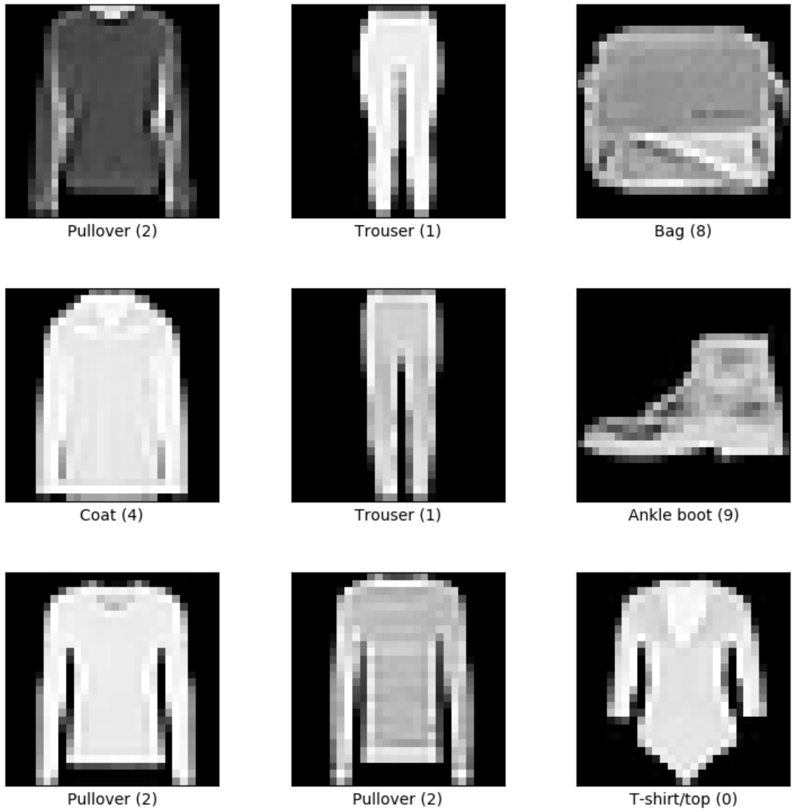
- Can construct a quantity g_{cq} which expresses alignment between two kernel-induced feature spaces (original dataset and PQK features).
- Embeddings with higher g_{cq} tend to show signs of quantum model outperforming.
- Can also artificially maximise g_{cq} by a relabelling procedure.
    - Tool for screening embeddings.



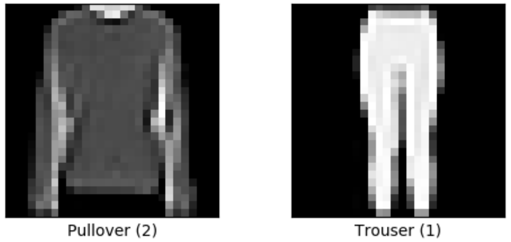
Source:
Power of data in quantum machine learning https://arxiv.org/pdf/2011.01938

Partial paper reproduction

Dataset



Filter



Truncate

[1.234, 2.345 ……]

Vector of length:
DATASET_DIM = N_QUBITS - 1

N_TRAIN:int =100, N_TEST: int=20
(Small but observed trends remain same for
The larger scale tests which were run)

Sources: https://github.com/zalandoresearch/fashion-mnist

Problem setup

Common QML Setup

Classical:

x y **Classical NN**

Quantum:

x y **Quantum NN**

Whats changes:

NN

Setup here

Classical:

x y Classical NN

Quantum:

x_pqk y Classical NN

Whats changes: data encoding.

Problem setup

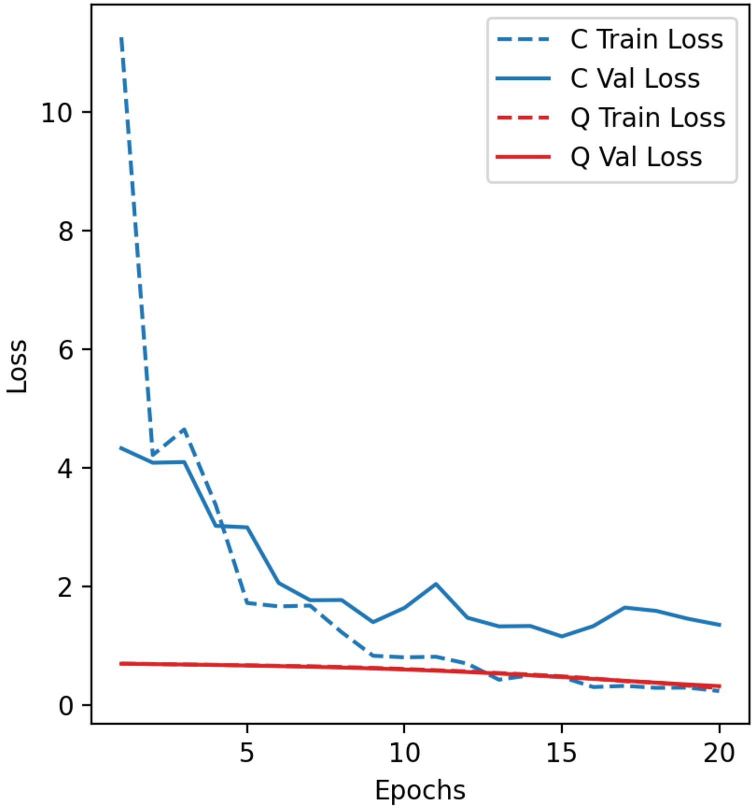
```
def __init__(self, DATASET_DIM):
    super().__init__()
    self.fc1 = torch.nn.Linear(DATASET_DIM, 128)
    self.fc2 = torch.nn.Linear(128, 64)
    self.fc3 = torch.nn.Linear(64, 16)
    self.fc4 = torch.nn.Linear(16, 2)
    self.relu = torch.nn.ReLU()
    self.dataset_dim = DATASET_DIM
```

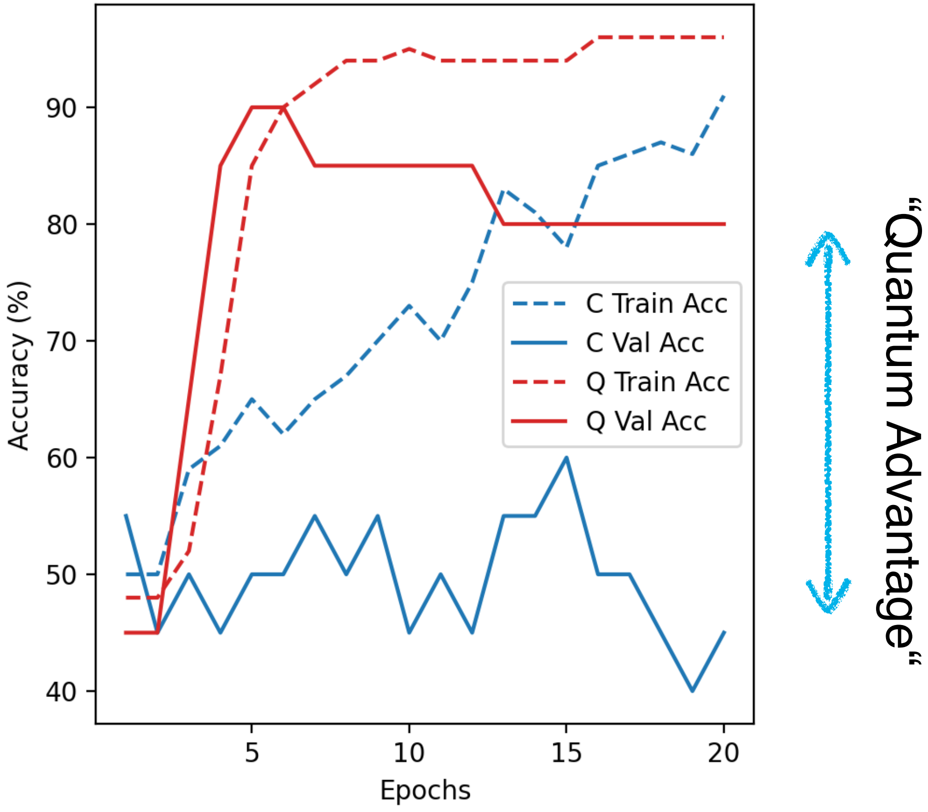
```
def __init__(self, N_QUBITS):
    super().__init__()
    self.fc1 = torch.nn.Linear(3*N_QUBITS, 128)
    self.fc2 = torch.nn.Linear(128, 64)
    self.fc3 = torch.nn.Linear(64, 16)
    self.fc4 = torch.nn.Linear(16, 2)
    self.relu = torch.nn.ReLU()
    self.n_qubits = N_QUBITS
```

Partial reproduction of power of data paper

HLRS

- TensorFlow tutorial from Google uses PQK circuits rather than full kernels. https://www.tensorflow.org/quantum/tutorials/quantum_data
- We reproduced their tutorial in PennyLane.
- “Quantum advantage“ is artificially constructed.



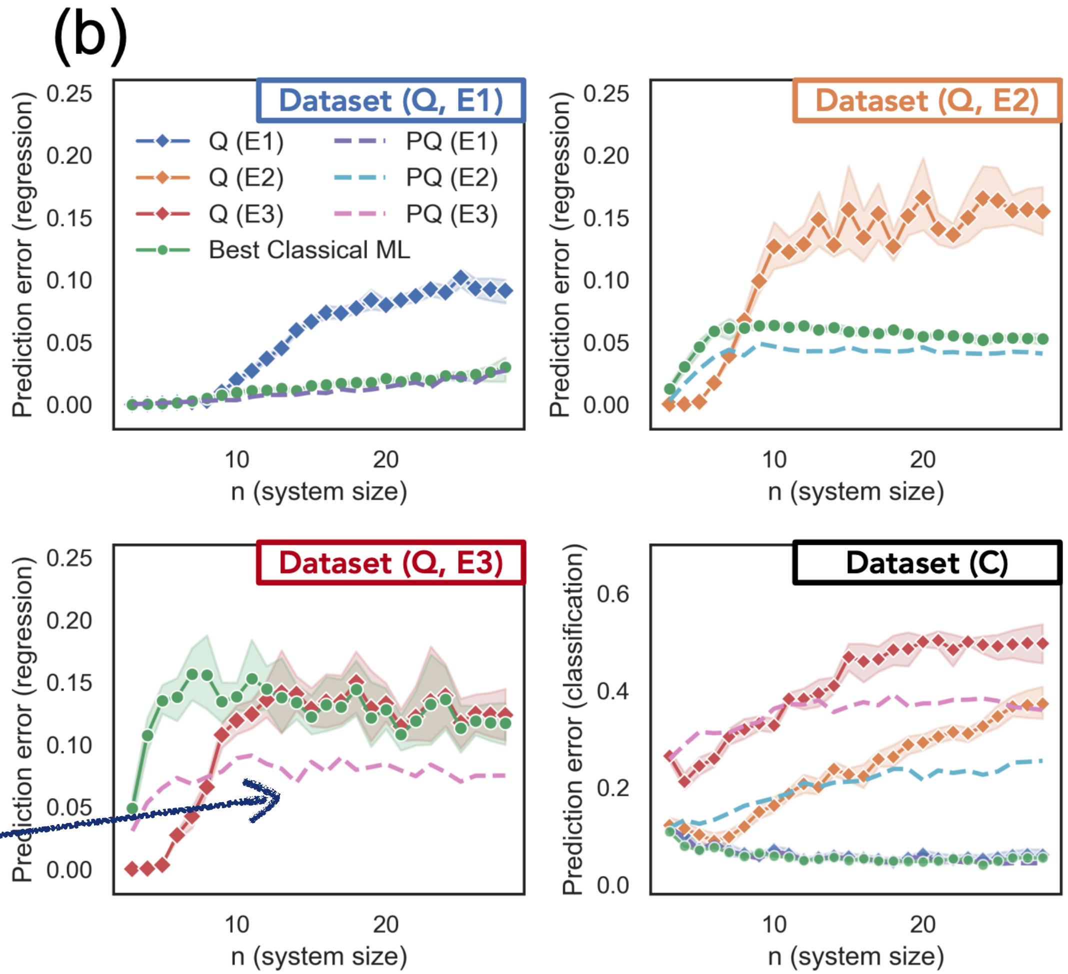


Quantum Advantage is hard to find

- Hard to get a real quantum advantage using quantum shadows and pure classification.

Regression (KRR) as they used kernels.

Can't reproduce for standard classification



Source:
Power of data in quantum machine learning https://arxiv.org/pdf/2011.01938

Useful vs useless advantage

Useful vs useless advantage

„The recent success… [showing] that quantum computers can sample from probability distributions that are exponentially difficult to sample from classically…

If these distributions were to coincide with real-world distributions…“

Pragmatic definition: Performs better

- on a useful dataset.
- On a NISQ device
- At a large problem size.

How useful is proving you can sample XEB circuits better?

Source:
Power of data in quantum machine learning https://arxiv.org/pdf/2011.01938

Relabelling OR rigging the data

Before

Classical:

x   y   Classical NN

Quantum:

x_pqk   y   Classical NN

Quantum perspective:

We found a dataset that is exponentially difficult to sample from classically.
Quantum oracle.

After relabelling

Classical:

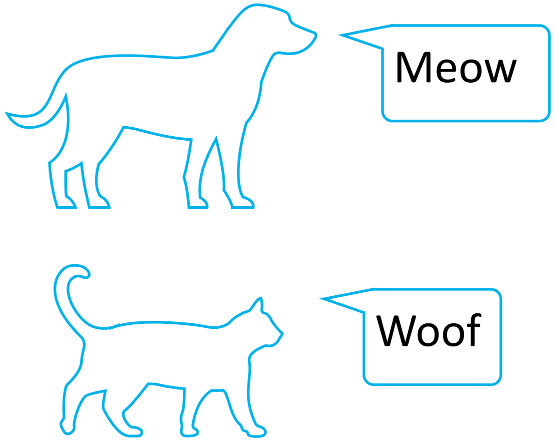
x   **y_new**   Classical NN

Quantum:

x_pqk   **y_new**   Classical NN

Classical perspective:

You changed the labels.

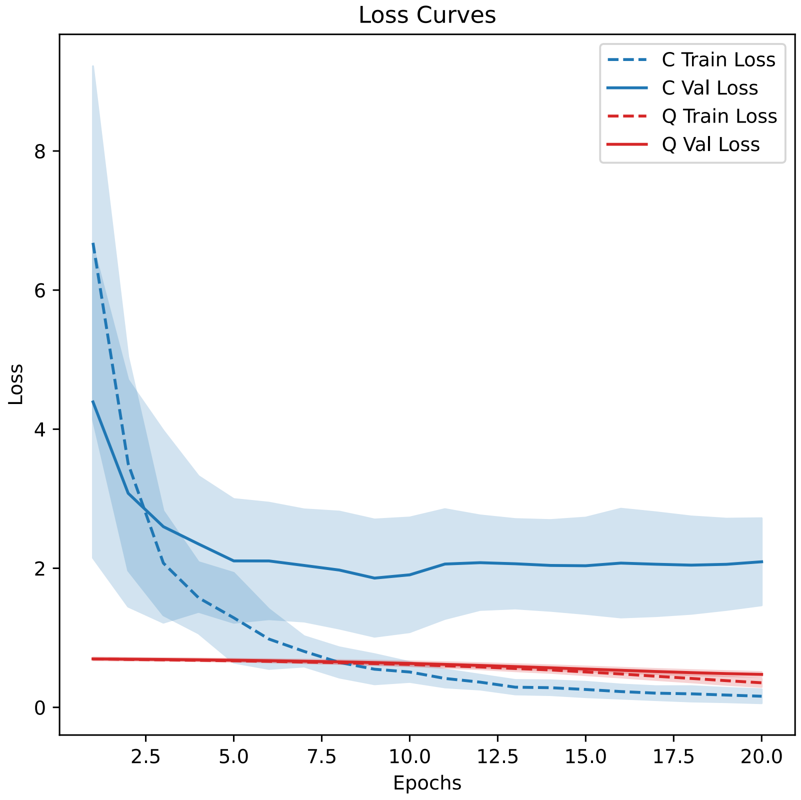


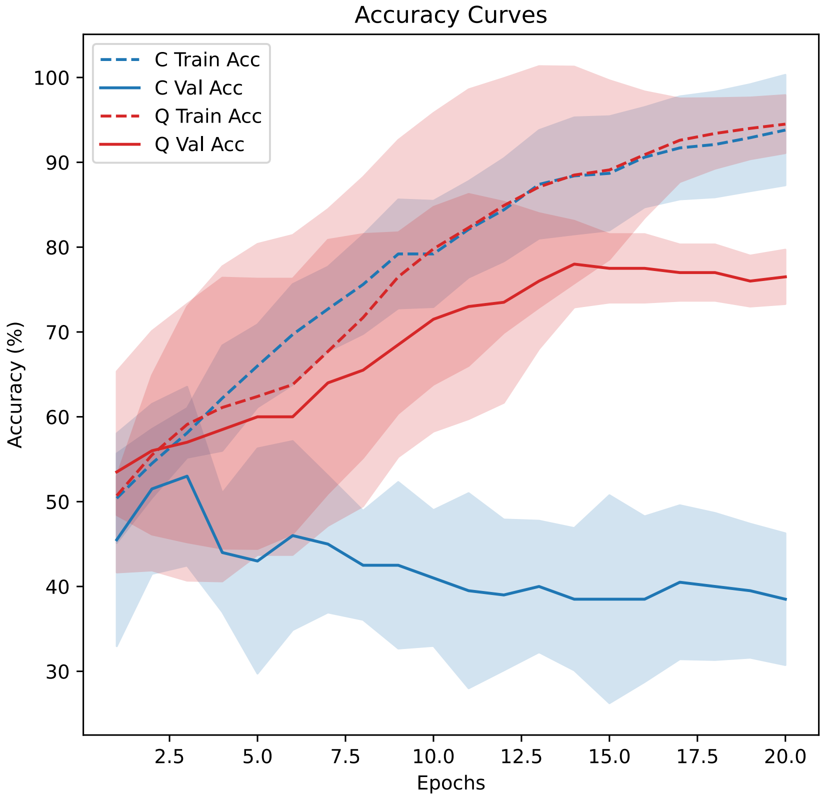
The artificial advantage is very hard to beat

100% rigging

n_qubits = 10
num_runs = 10

- Results show a lot of variance between runs but overall trends remain same.
- Full relabelling is very hard to beat even use more sophisticated classical model.



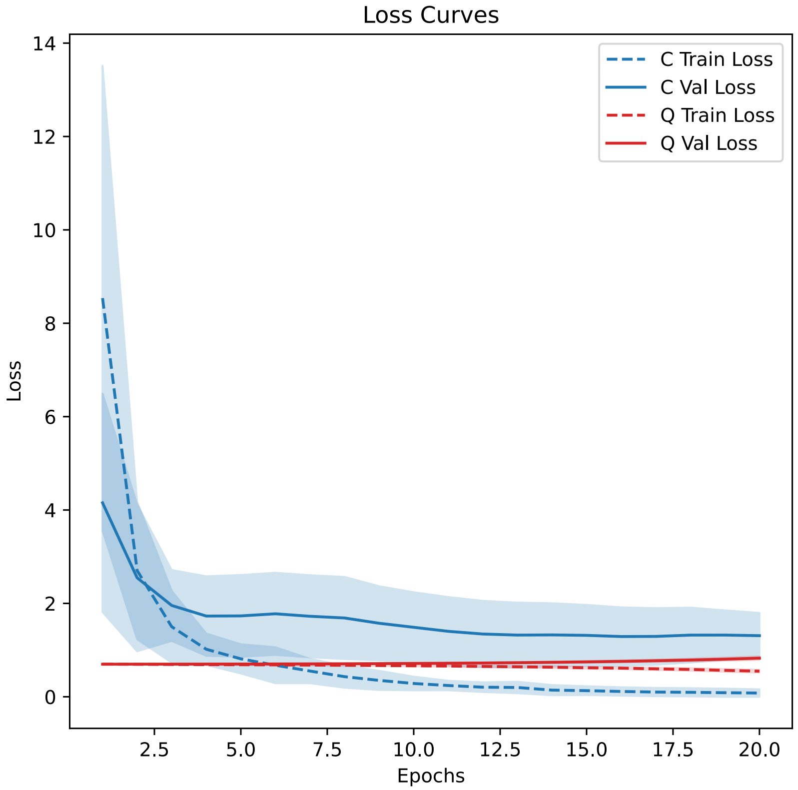


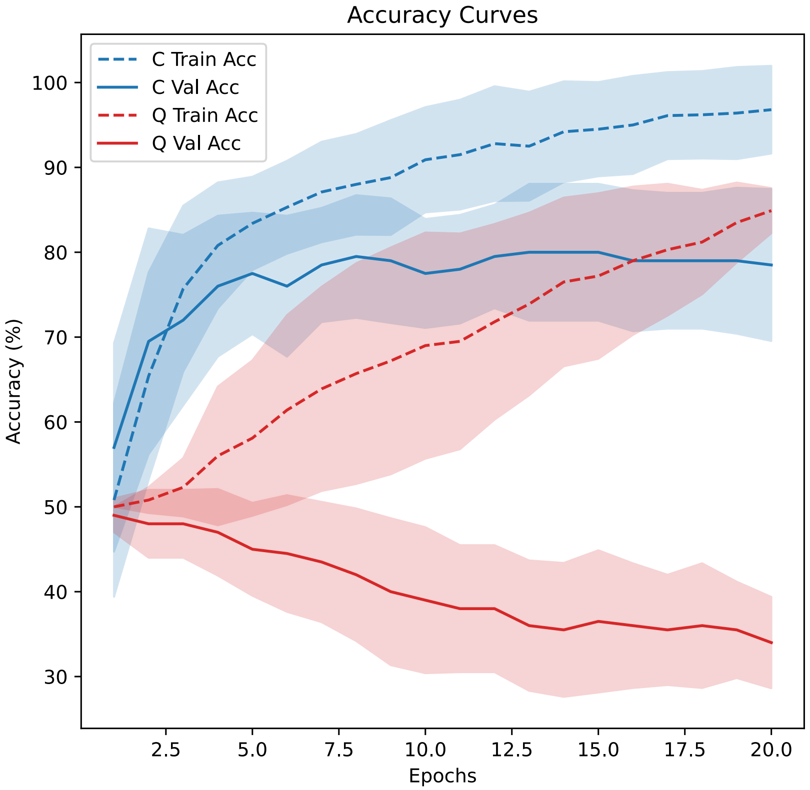
Reduce the data rigging and classical wins

20% rigging

n_qubits = 10
num_runs = 10

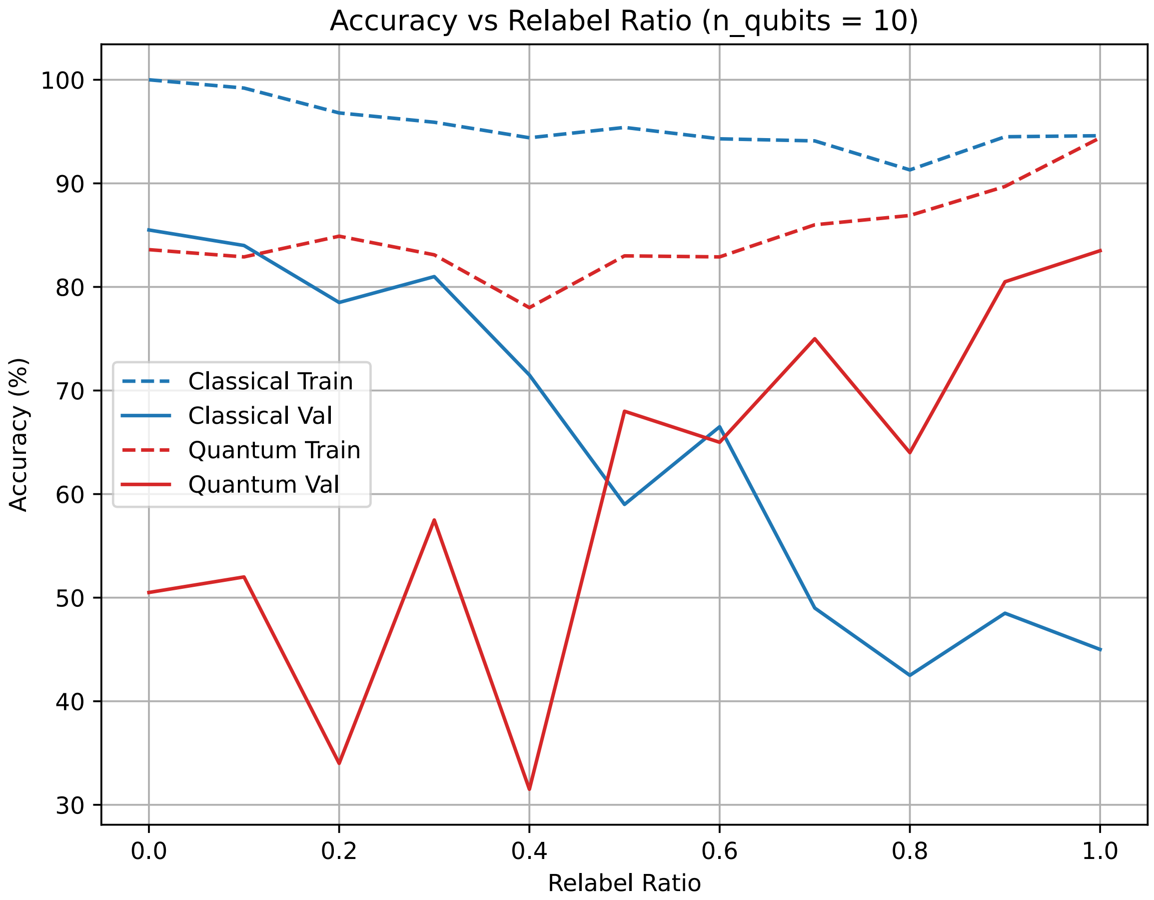
- Flip some of the labels back and advantage disappears. No straightforward relationship for a „partial quantum advantage“.





Data rigging effects

- Mainly seems to degrade classical performance rather than improve quantum performance.
- Quantum also has the advantage that it is already fed the PQK features.



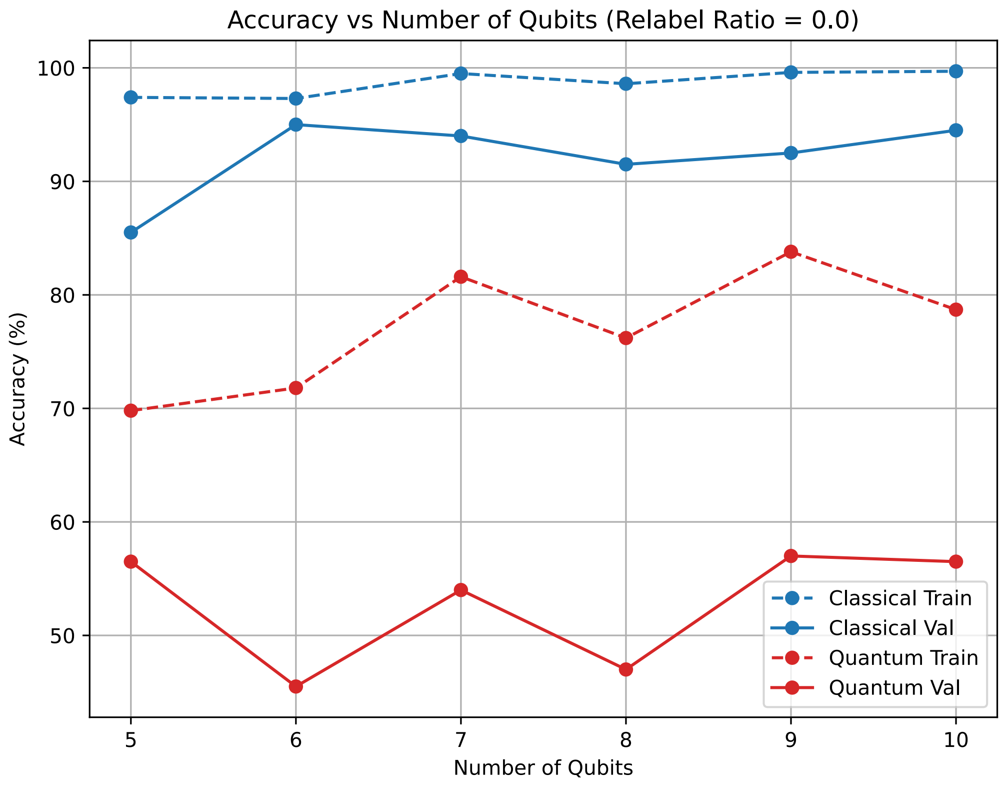
Conclusion

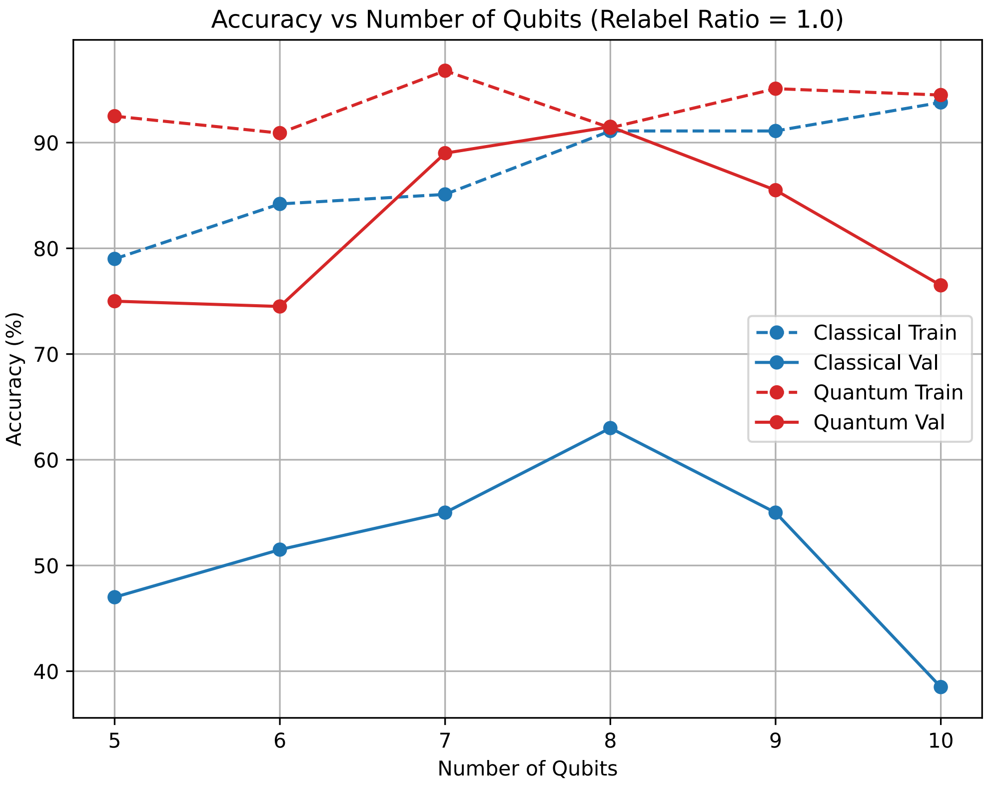
- Can get a mild quantum advantage with kernel methods.
- Relabelling to study quantum advantage rigs the game rather than finding a better learning method.

Future work

- Use of kernel methods instead of PQK circuits.
- Larger scale simulations. N_TRAIN:int =100 is rather small.

Vielen Dank!


Accuracy vs Number of Qubits (Relabel Ratio = 0.0)
Accuracy (%)
Number of Qubits
100
90
80
70
60
50
5
6
7
8
9
10
Classical Train
Classical Val
Quantum Train
Quantum Val


Accuracy vs Number of Qubits (Relabel Ratio = 1.0)
Accuracy (%)
Number of Qubits
90
80
70
60
50
40
5
6
7
8
9
10
Classical Train
Classical Val
Quantum Train
Quantum Val