Analysis of Quantum Circuits for Classification Tasks

Master's Thesis

Author: Prasham Bhuta
Supervisor: Assoc. Prof. Dr. Linas Petkevičius
Reviewer: MIF/DMSTI Research Assistant Marco Marcozzi

Vilnius University
Faculty of Mathematics and Informatics

2024-January-18

Goal

To describe and showcase quantum algorithms, and compare with classical machine learning methods by setting a classification task under a similar construct.

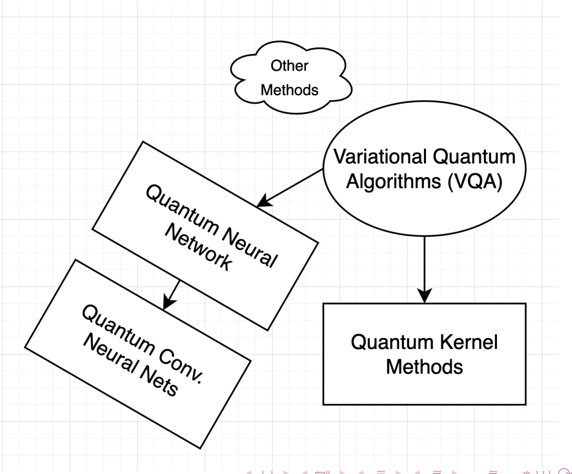
Objectives

1. Systematic exploration of the development and the underlying principles governing quantum circuits.
2. Showcase the use of current technologies to initialize and train quantum architecture.
3. Investigation on quantum performance with varying quantum circuits, and problem complexities.
4. Classification with quantum circuits, and compare performance with classical methods.

VQA at the core of quantum algorithms.

Variational Quantum Algorithms (VQA) have become one of the integral methods to construct Quantum Circuits. Some peculiarities of quantum computing, focused on, are:

- The performance for problems compared to classical computing.
- The time complexity of quantum systems when run on a classical-quantum simulated system.



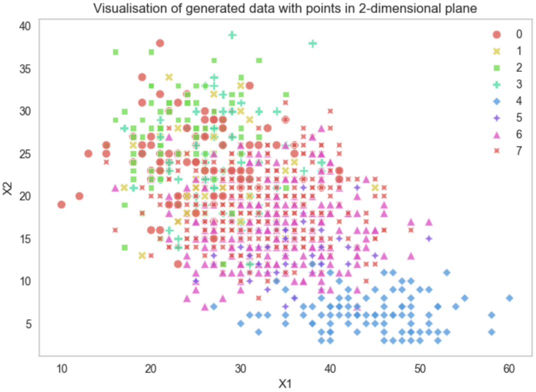
To perform classification tasks with quantum circuits, and analyze their performance with classical methods, the models used are:

Quantum Circuit Methods	Classical Methods
Quantum Support Vector Classifier (QSVC)	Support Vector Machine Classifier
Variational Quantum Classifier (VQC)	Light Gradient Boost Machine
	Quadratic Discriminant Analysis
	Random Forest Classifier
	K Neighbors Classifier

Dataset

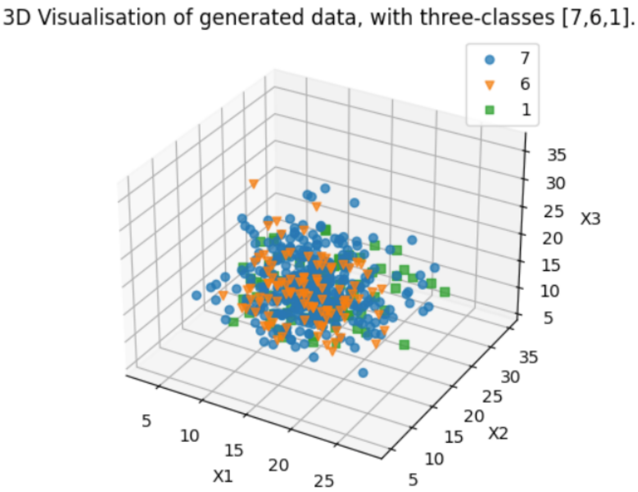
X1	X2	X3	X4	Y1	Y2	Y3	YT
15.0	11.0	18.0	7.0	1	1	1	7
14.0	12.0	16.0	5.0	1	1	1	7
14.0	17.0	16.0	6.0	0	0	1	1
14.0	13.0	17.0	6.0	1	1	1	7
25.0	3.0	17.0	5.0	1	0	1	5

1 lentelė. Table represents the generated data (X, Y) and transformed labels YT.



1 pav. Visualisation of data in 2D space

Dataset



2 pav. Visualisation of few classes.

Approach

Parameters	Values
No of samples	1024
No of features (X_i)	[2, 4, 8, 12, 16]
No of classes	3
No of labels	3
No of feature map reps (Quantum)	[2,3,4,5,6,8,12]
No of ansatz reps (Quantum)	[2,3,4,5,6,8,12]

2 lentelė. Table shows the parameters used for quantum circuit complexity.

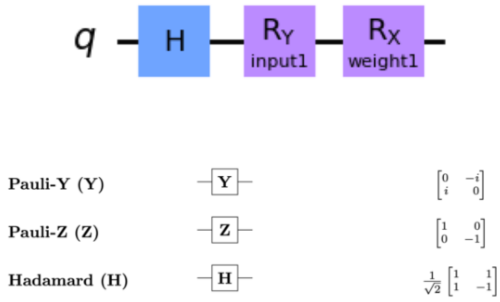
Approach

Parameters	Values
No of samples	1024
No of features (X_i)	[2, 4, 8, 12, 16]
No of classes	[3,4,5,6,8,12]
No of labels	[3,4,5,6,8,12]
No of feature map reps (Quantum)	3
No of ansatz reps (Quantum)	3

3 lentelė. Table shows the parameters used for classification problem complexity.

Notations

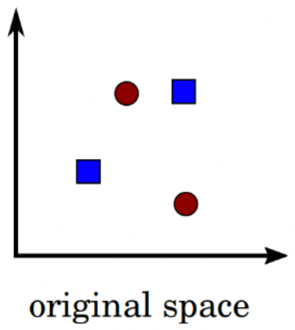
- **Qubit:** Fundamental units behind quantum computing. Represented $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α, β are the coefficients of the basic states of the qubit.
- **Quantum Gate:** Analogical to classical gates applied to qubit states, eg. applying a state matrix U such that $U|\psi_1\rangle = |\psi_2\rangle$, where $|\psi_2\rangle$ is the new quantum state.
- **Quantum Circuits:** A collection of quantum gates connected and working together.



3 pav. Quantum circuit with Hadamard Gate.

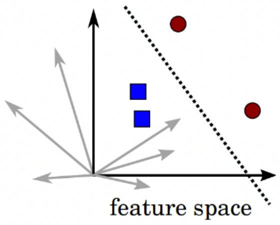
Definitions

- **FeatureMap:** Formally, if χ is an input set, then a feature map is a map $\phi : \chi \to F$ from input to the vector in Hilbert space. The vector $\phi(x)$ for all $x \in \chi$ are called feature vectors. F is a vector space which is also called the Hilbert space.
- Feature map creates the data loader state to feed the input data into a quantum state, *ZZFeatureMap* in study experiments.



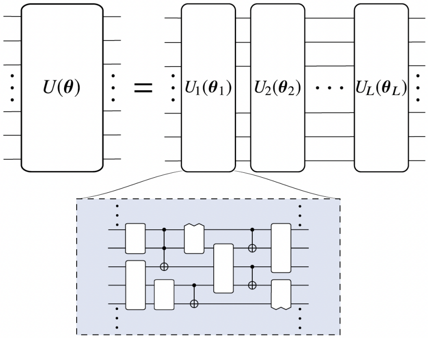
4 pav. Representation in original space.

- **FeatureMap:** Formally, if χ is an input set, then a feature map is a map $\phi : \chi \rightarrow F$ from input to the vector in Hilbert space. The vector $\phi(x)$ for all $x \in \chi$ are called feature vectors. F is a vector space which is also called the Hilbert space.
- Feature map creates the data loader state to feed the input data into a quantum state, *ZZFeatureMap* in study experiments.

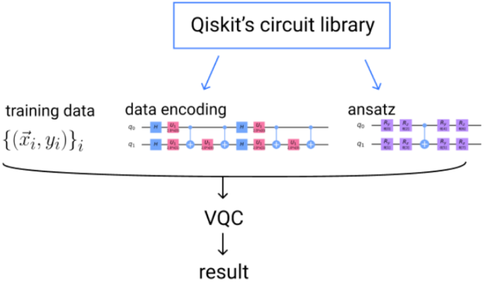


5 pav. Representation in higher dimensional space.

- **Ansatz:** "Initial placement of tool for a workpiece." It is the starting assumption of the parameter θ and how they can be trained to minimize the cost, the unitary $U(\theta)$ is generally expressed as a product of L sequentially applied unitaries $U(\theta) = U_L(\theta_L)...U_2(\theta_2)U_1(\theta_1)$.
- *RealAmplitude* with alternating rotations of Y and CX Gates.



6 pav. Ansatz basic diagram



7 pav. VQC classifier with *ZZFeatureMap*, and *RealAmplitudes* ansatz.

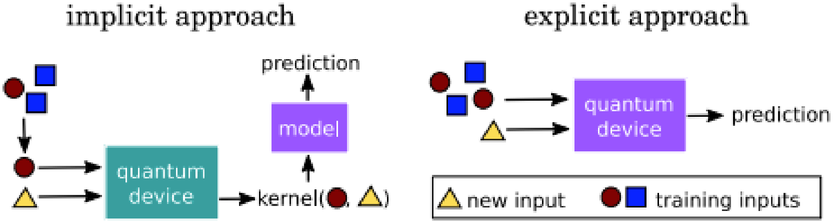
- **VQC:** Classifier of Variational Quantum Algorithm (VQA) which uses a classical co-part to train the parameters. The cost function is defined as
$$C(\theta) = f(\{\rho_k\}, \{O_k\}, U(\theta)),$$
  - f is Constrained Optimization BY Linear Approximation (COBYLA),
  - ρ_k is the input states,
  - O_k is a set of observables,
  - $U(\theta)$ is unitary function

- **QSVC:** Quantum Support Vector Classifier (QSVC) leverages quantum kernel methods [a] to map $\phi(\vec{x})$ the input vector to a Hilbert space. Mathematically, the kernel matrix is represented as,

$$K_{ij} = \left|\langle \phi(\vec{x}_i) \middle| \phi(\vec{x}_i) \rangle\right|^2$$

  - K_{ij} is the kernel matrix,
  - $(\vec{x}_i, \vec{x}_j)$ are n dimensional inputs,
  - $\phi(\vec{x})$ is the quantum feature map,
  - $|\langle a|b\rangle|^2$ denotes the overlap of two quantum states a and b.

[a] [1] Havlíček, V., Córcoles, A. D., Temme, K., Harrow, A. W., Kandala, A., Chow, J. M., & Gambetta, J. M. (2019). Supervised learning with quantum-enhanced feature spaces. Nature, 567(7747), 209-212. arXiv:1804.11326v2 [quant-ph]



8 pav. Two approaches of quantum algorithms, VQC, and QSVC with Kernel.

Experimental Approach

Data Labels Transformation

To transform the data into a single dimension, we use the trick of converting the n-array of 0s, and 1s into integers. The approach is how we convert binary to integers of base 10, for example [1, 0, 1] will convert to $1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 5$.

Loss

For multiclass problems, i.e., the number of classes is >2, cross-entropy is calculated for each class label and the result is the sum of all losses. The formula behind it is,

$$Loss = -\sum y_{o,c} log(p_{o,c})$$

where y is presence of label in observation o, p is predicted probability observation o is of class c.

Quantum Training Algorithm

Algorithm 2 Quantum training

Load data from sklearn.datasets.make_multilabel_classification
Setup Feature map, Ansatz & Quantum Kernel
Setup QSVC or VQC:
  # Specify target variable(Y) and features (X)
  # Transform target variable(Y) to (YT)
Compare and Train Models:
  `QSVC.fit(X,YT) or VQC.fit(X,YT)`
Make Predictions:
  # Make predictions on new data
  `predictions = QSVC.predict(X) or VQC.predict(X)`
  # Transform predictions to original shape
  # Calculate F1 score

---

9 pav. Quantum circuit training using QVC, and QSVC with Kernel.

Approach

Parameters	Values
No of samples	1024
No of features (X_i)	[2, 4, 8, 12, 16]
No of classes	3
No of labels	3
No of feature map reps (Quantum)	[2,3,4,5,6,8,12]
No of ansatz reps (Quantum)	[2,3,4,5,6,8,12]

4 lentelė. Table shows the parameters used for quantum circuit complexity.

Parameters	Values
No of samples	1024
No of features (X_i)	[2, 4, 8, 12, 16]
No of classes	[3,4,5,6,8,12]
No of labels	[3,4,5,6,8,12]
No of feature map reps (Quantum)	3
No of ansatz reps (Quantum)	3

5 lentelė. Table shows the parameters used for classification problem complexity.

Results

Performance metric - F1 Score

To measure the quality of the classification tasks, the metric used is the F1 score. The F1 score is a harmonic mean of precision and recall. It is given by the formula,

$$F_1 = 2\frac{precision\ .\ recall}{precision + recall}$$

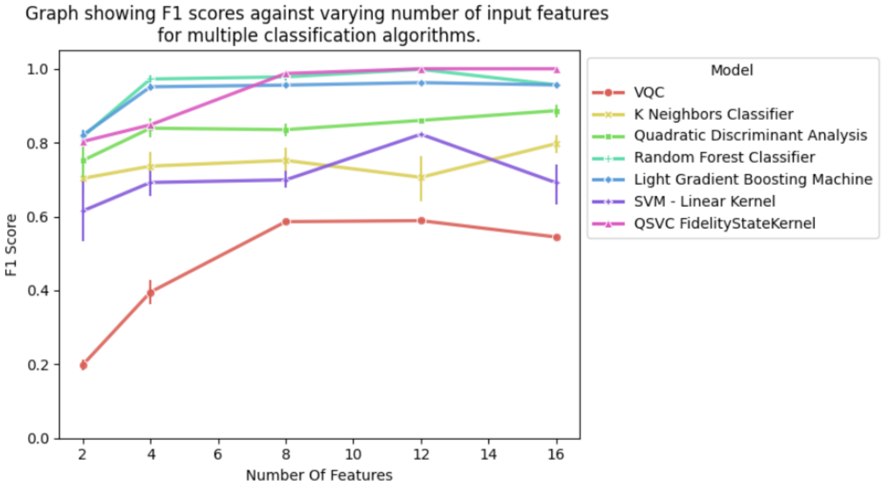
The final score is calculated based on the weighted average of the class occurrence frequency, taking in account for class imbalance.

Results

Model Name	No of features (M) 2	4	8	12	16
VQC	0.199	0.395	0.586	0.589	0.545
QSVC FidelityStateKernel	0.803	0.848	**0.987**	**1.000**	**1.000**
K Neighbors Classifier	0.703	0.736	0.751	0.706	0.797
Light Gradient Boosting Machine	0.820	0.951	0.955	0.962	0.956
Quadratic Discriminant Analysis	0.751	0.839	0.835	0.860	0.886
Random Forest Classifier	**0.821**	**0.972**	0.978	0.998	0.956
SVM - Linear Kernel	0.615	0.692	0.699	0.823	0.6961

10 pav. F1 score for classical and quantum models with varying number of input features.

Results



11 pav. F1 score for classical and quantum models with varying number of input features.

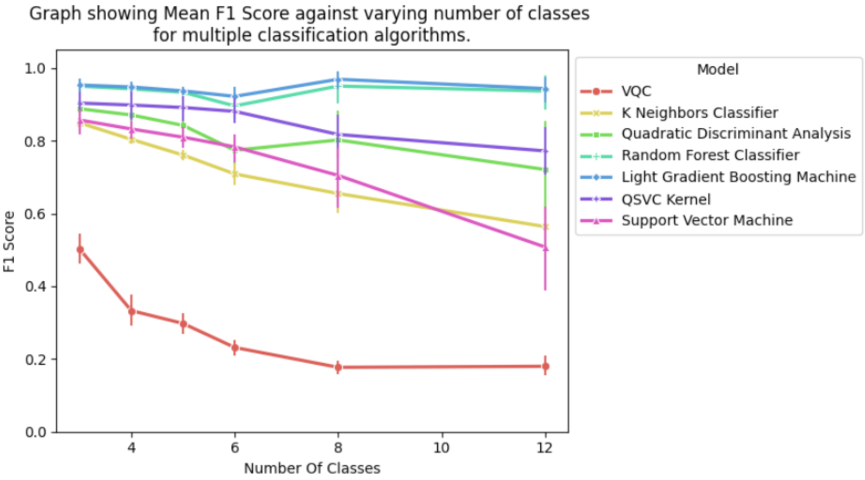
Results

Model Name	No of classes (N) 3	4	5	6	8	12
VQC	0.502	0.333	0.298	0.232	0.178	0.179
QSVC FidelityStateKernel	0.903	0.899	0.891	0.880	0.817	0.772
K Neighbors Classifier	0.849	0.803	0.760	0.705	0.656	0.564
Light Gradient Boosting Machine	**0.953**	**0.947**	**0.936**	**0.921**	**0.968**	**0.943**
Quadratic Discriminant Analysis	0.888	0.870	0.842	0.773	0.802	0.720
Random Forest Classifier	0.950	0.942	0.933	0.895	0.950	0.936
SVM - Linear Kernel	0.856	0.832	0.810	0.783	0.704	0.507

12 pav. F1 score for classical and quantum models with varying classes.

Model Name	No of feature map and ansatz repetitions (R) 2	3	4	5	6	8	12
VQC	0.296	0.294	0.194	0.213	0.182	0.201	0.191

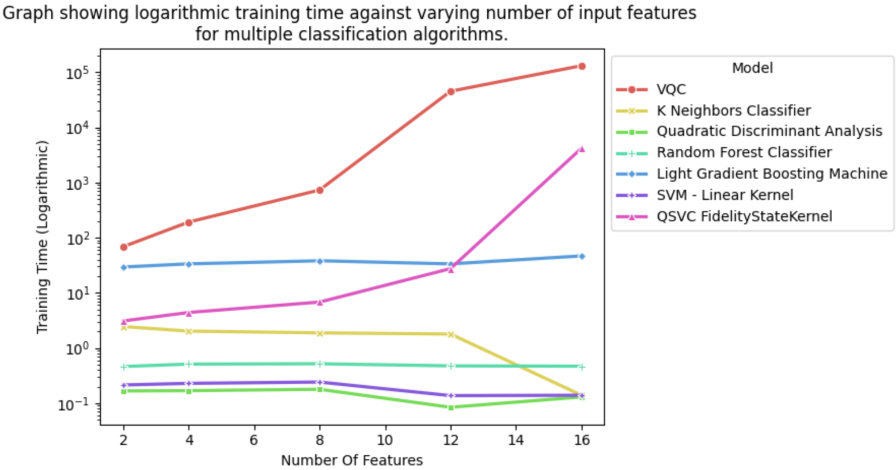
13 pav. F1 score for VQC with varying degrees of feature map, and ansatz repetitions.



14 pav. F1 score for classical and quantum models with varying classes.

Model Name	No of features (M) 2	4	8	12	16
VQC	68.729	192.284	732.892	45223.159	131429.982
QSVC FidelityStateKernel	3.096	4.406	6.810	27.492	4174.854
K Neighbors Classifier	2.449	2.025	1.882	1.794	0.139
Light Gradient Boosting Machine	29.447	33.578	38.154	33.527	46.722
Quadratic Discriminant Analysis	0.167	0.169	0.179	0.084	0.130
Random Forest Classifier	0.462	0.511	0.519	0.475	0.468
SVM - Linear Kernel	0.214	0.229	0.241	0.137	0.139

15 pav. Computational time for classical and quantum models with varying features.



16 pav. Logarithimic computational time for classical and quantum models with varying features.

Results

1. The best-performing models for varying numbers of input features (M) are Random forest (for a lower number of features), and QSVC (for a higher number of features).
2. The performance of the quantum QSVC model for varying numbers of classes is comparable to the top-performing (Light Gradient Boost Machine) model.
3. The quantum model training time on the quantum-classical simulator for the number of input features (M) for QVC, and QSVC increases exponentially with an increase in complexity.
4. The performance, F1 score, for QVC with varying numbers of feature map and ansatz repetitions doesn't show improvement.

Conclusions

1. Quantum support vector classifier (QSVC) based on a linear kernel outperforms the classical machine learning classifiers.
2. The performance of the quantum SVC classifier drops by a small margin with increasing the number of classes, this is in contrast with the good classical models whose performance remains stable.
3. Increasing the complexity of the quantum circuit algorithms, exponentially increases computation time on the quantum-classical simulator, but importantly on a quantum-classical simulator.
4. Varying the number of feature map repetitions, or ansatz repetitions to vary the variational quantum classifier (QVC) complexity does not lead to improved performance of the classifier.
5. The performance, F1 score, & training time of quantum models would change if the experiments were repeated on a physical quantum computer.

Future Scope

Further research within the scope of this study and overall in the field of quantum computing can be stated as,

Possible extensions in current study

- Use of different methods of entanglement, such as 'linear', 'rotational', 'reverse-linear', and others to check for performance.
- Checking for performance on real quantum systems, especially the time complexity task.
- Experiment with the same approach but a different quantum technology platform, and compare performances.

Future Scope

Further research within the scope of this study and overall in the field of quantum computing can be stated as,

In the field of quantum computing

- Construction of complex architecture such as Quantum Convolutional Neural Nets (QCNN), Quantum Autoencoders will widen the scope of machine learning tasks with quantum computers.
- Since limited scope to perform multi-class, multi-label classification, construction of circuits that facilitate that.

Thank you for your attention! Labai Ačiū!

Classical Training Algorithm

Algorithm 1 Classical training

Load data from sklearn.datasets.make_multilabel_classification

Setup PyCaret:

 # Specify target variable(Y) and features (X)

 # Transform target variable(Y)

```
exp1 = setup(data, target='YT')
```

Create and Train Models:

 # Create different classification models

```
create_model(models=['knn', 'qda', 'rfc', 'svm', 'lgbm'])
```

Make Predictions:

 # Make predictions on new data

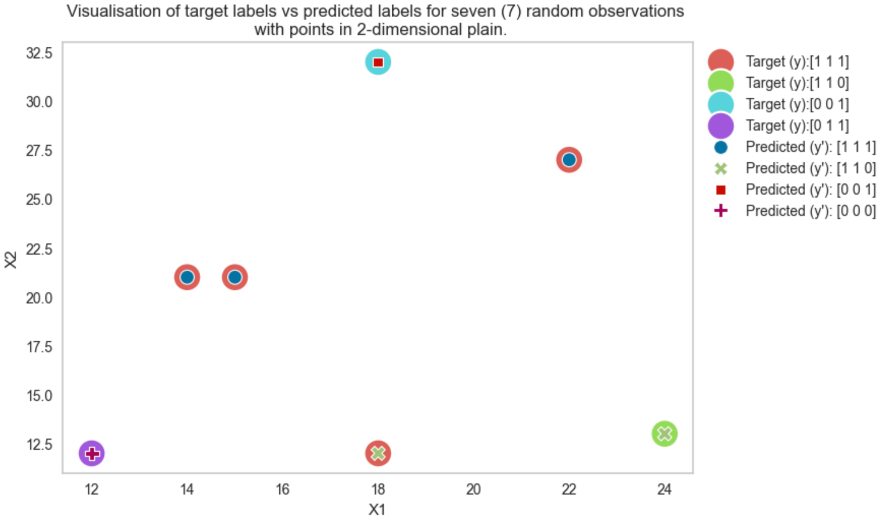
```
predictions = predict_model(best_model, X)
```

 # Transform predictions to original shape

 # Calculate F1 score

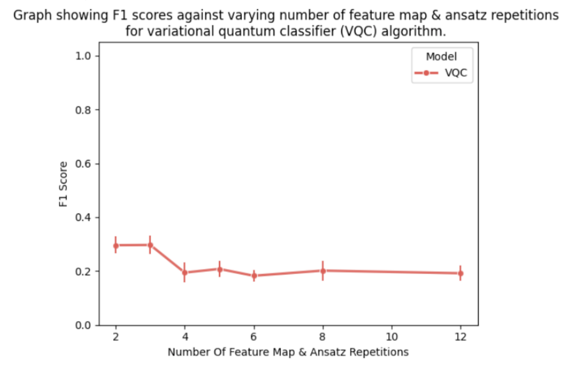
17 pav. Classical circuit training with pycaret.

+Results



Visualisation of target labels vs predicted labels for seven (7) random observations with points in 2-dimensional plain.
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
12.5
X2
12
14
16
18
20
22
24
X1
Target (y):[1 1 1]
Target (y):[1 1 0]
Target (y):[0 0 1]
Target (y):[0 1 1]
Predicted (y'): [1 1 1]
Predicted (y'): [1 1 0]
Predicted (y'): [0 0 1]
Predicted (y'): [0 0 0]

Performance with varying number of reps.



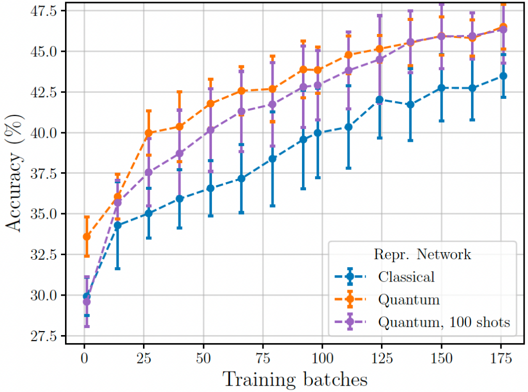
19 pav. Performance of VQC with varying degrees of feature map and ansatz reps.

[

Why quantum technology for classification?] Classification is a fundamental task to recognize patterns in data, and quantum circuits are currently in the pubescent stage.

- Makes a good match!

In their study[a], the researchers conducted training with the first five classes of CIFAR-10 and implemented a projection head [JAX+22]. They explored three distinct representation networks: a classical MLP with bias and Leaky ReLU activation functions, a quantum network trained on a state vector simulator, and a quantum network trained on a sampling-based simulator.



[a] B Jaderberg et al 2022 Quantum Sci. Technol. 7 035005

In their study[a], the highest accuracy is obtained at the end of training, where the quantum model achieves an accuracy of (46:51 ± 1:37)% compared to (43:49 ± 1:31)% for the classical model.[JAX+22]

- Fairly comparing Quantum & Classical Methods remains an open question.
- Control over how much task is classical and how much is quantum based.

[a] B Jaderberg et al 2022 Quantum Sci. Technol. 7 035005

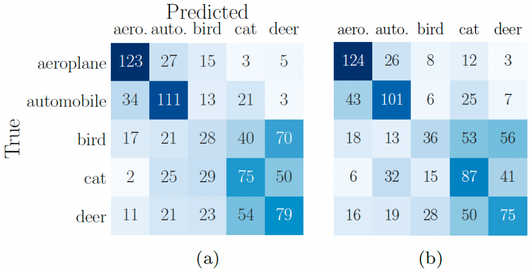


FIG. 6. Confusion matrix from classifying 900 images using the best performing (a) classical model evaluated on a classical computer (b) quantum model evaluated on a real quantum computer with 100 shots per circuit. For a given true label (rows) and predicted label (columns), the number in each box shows the total number of times that prediction was made.