Background

Quantum Machine Learning (QML) is a relatively recent field that aims to study the possible ways to combine Machine Learning methods with Quantum Computing. One such way with immediate applications consists in using quantum circuits as parts of ML algorithms for classical (non-quantum) data. In this context, the goal is to use the higher expressivity of quantum circuits, encoding the data in high-dimensionality Hilbert spaces, to explore more efficiently the parameter space and improve the quality of the final model. Although promising for some applications, QML is hindered by current Quantum Processing Units (QPUs) being very rare and costly to use, as well as having low throughput and no usable RAM equivalent. In practice, this means that Variational Quantum Algorithms (VQA) [1], which optimize the parameters of a quantum circuits in a similar way as their classical counterparts, require too many quantum circuit evaluations to be trainable on real hardware, even for very small datasets. In fact, to the best of our knowledge, no current QML method can be trained on real hardware for non-trivial tasks. As such, these methods remain confined to simulators, which limits their ability to generalize.

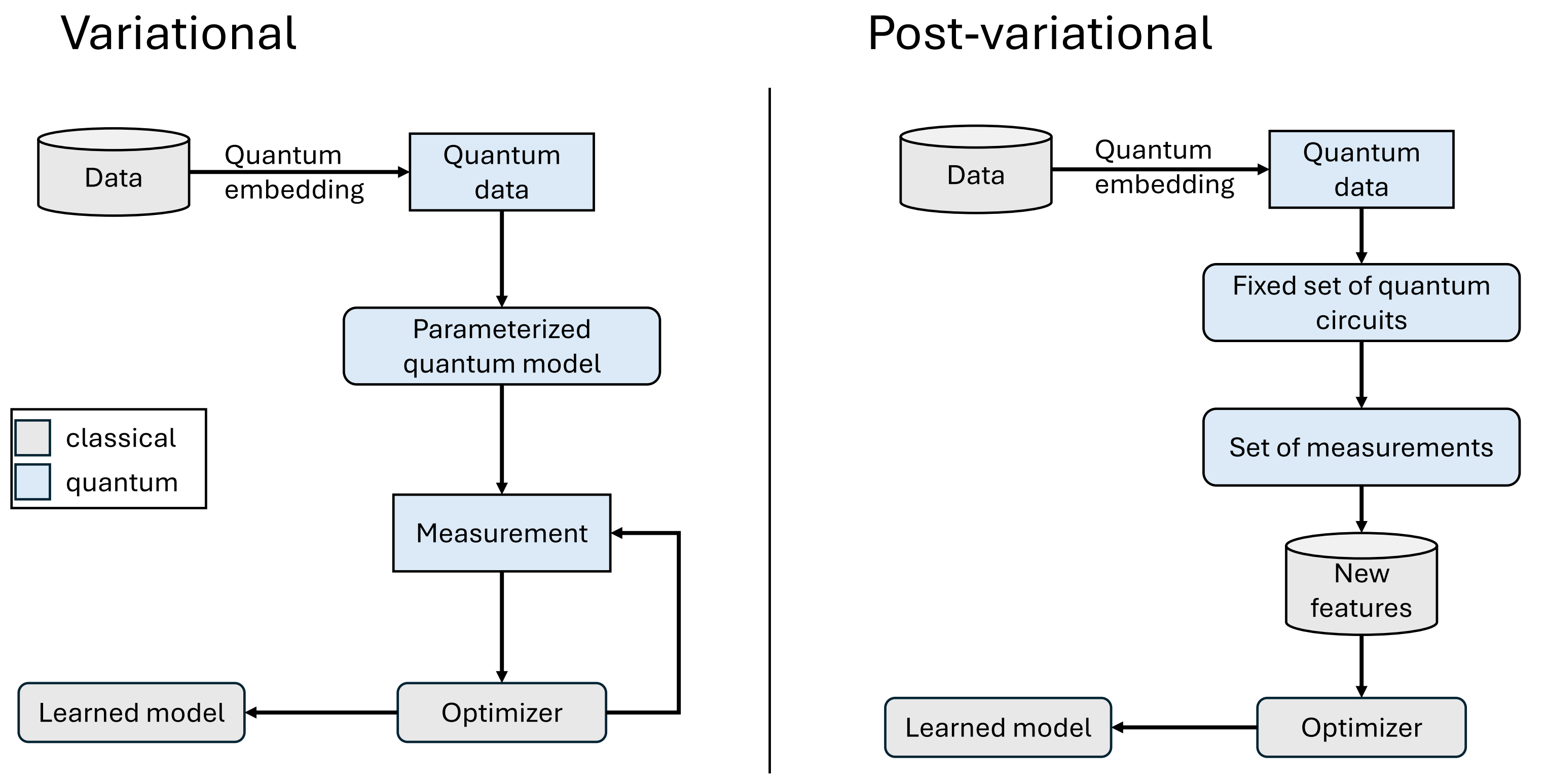
Goal

Find a Quantum Machine Learning algorithm which can realistically be fully trained and evaluated on real quantum hardware

Methods – algorithm

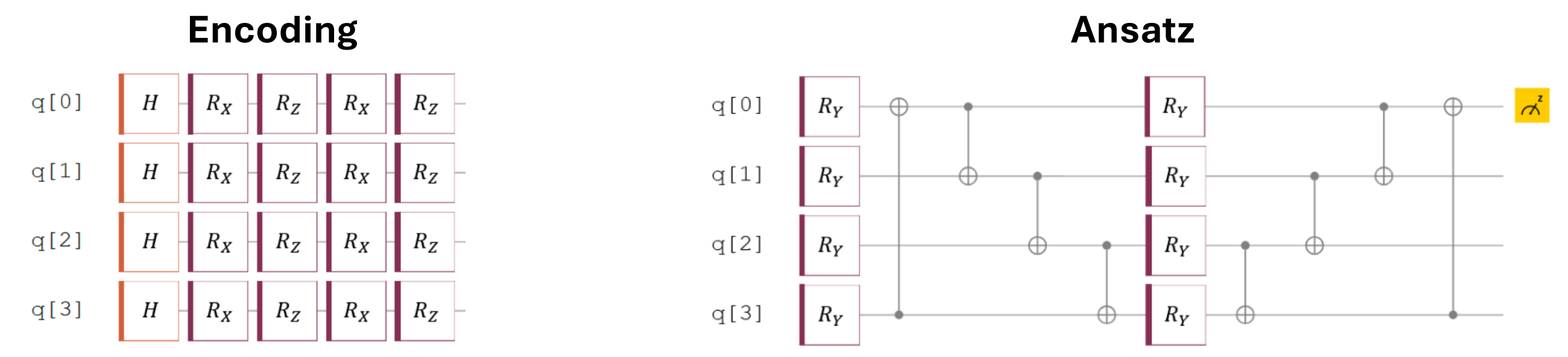
Post-Variational Quantum Neural Networks

Post-Variational Quantum Neural Networks (QNN) are a recent QML algorithm [2]. The core idea is to predefine a set of quantum circuits, apply them to each sample of the dataset, and use the measurements from these circuits as features of the new transformed dataset. The final model (a Multi-Layer Perceptron) is then trained classically on this dataset. Although this method was originally developed to solve the issue of barren plateaus that occur when training VQA, it also has the benefit of requiring fewer circuits evaluations.


Variational

Post-variational

Selecting the quantum circuits for feature mapping


Encoding

Ansatz

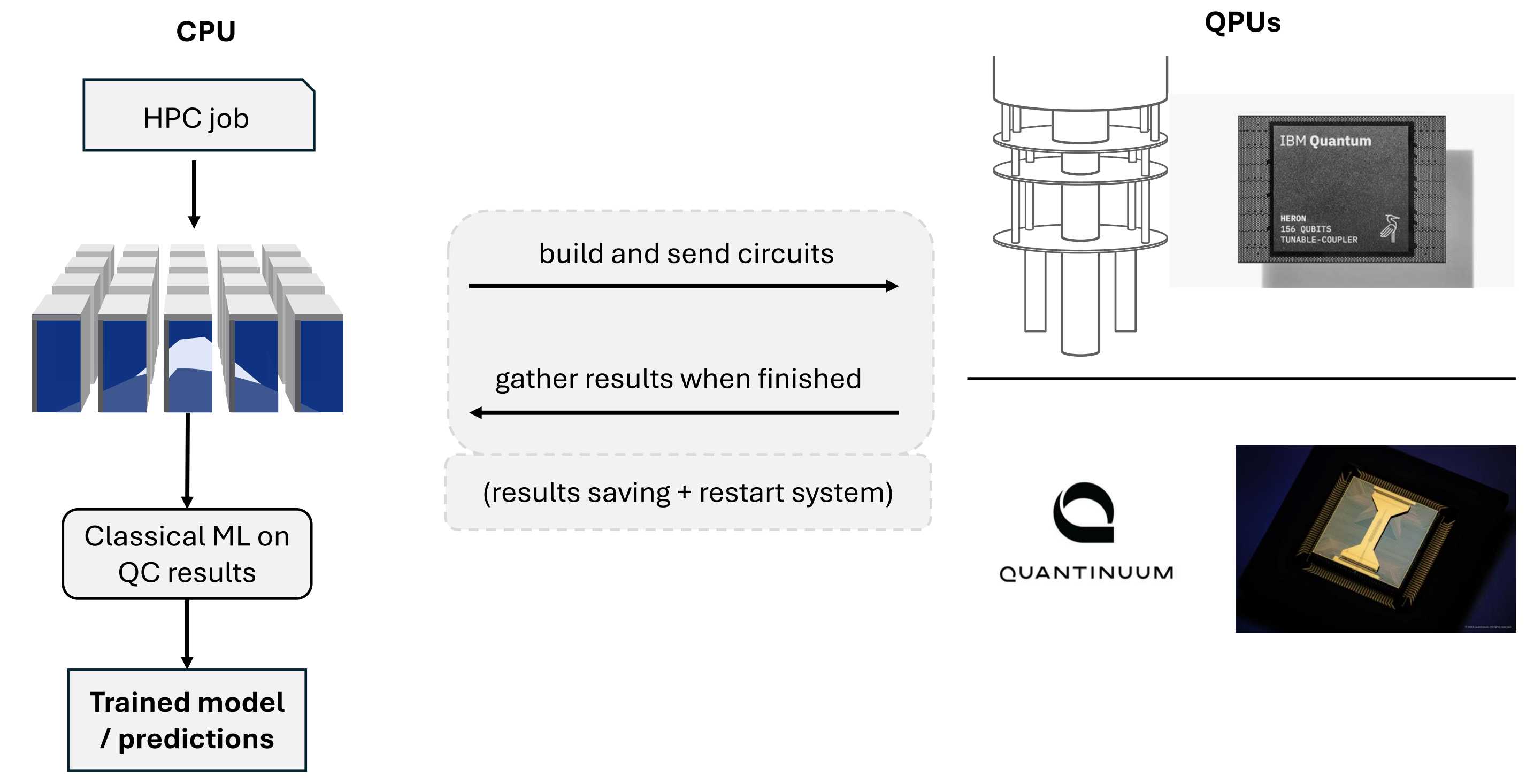
Two variants:

- *Observables*: use the encoding circuit only + measure subsets of qubits in different Pauli bases (X, Y, Z)
- *Ansatze*: use the encoding+ansätz circuit, add a subset of Ry rotations (fixed value), measure top qubit

Methods – workflow and data

Hardware used:

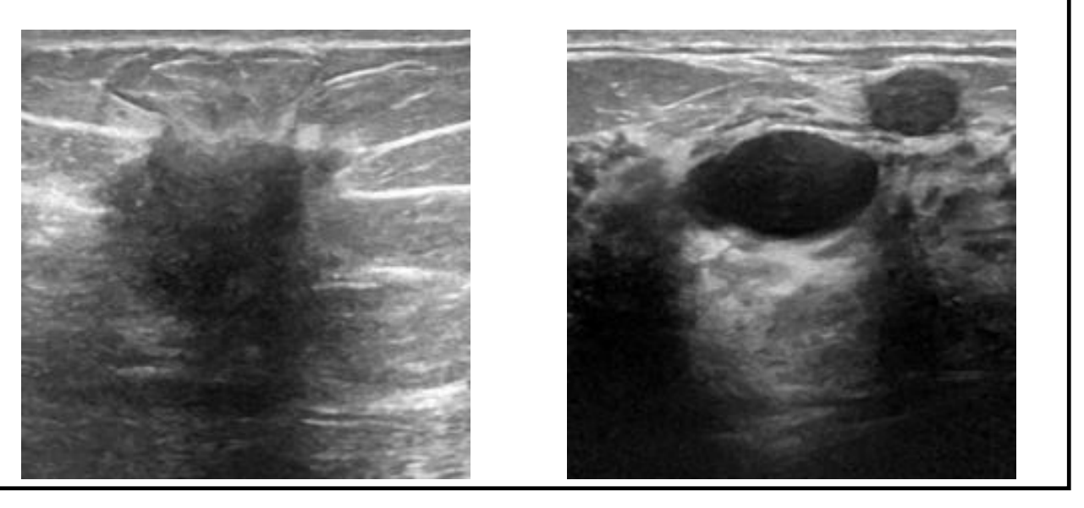
- CPU: supercomputer Fugaku
- QPU: IBM Heron r2 156-qubit superconducting device “IBM-Kobe”
- QPU: Quantinuum H1 20-qubit ion-trap device “Reimei”


CPU
QPUs
HPC job
build and send circuits
gather results when finished
(results saving + restart system)
Classical ML on QC results
Trained model / predictions
IBM Quantum
QUANTINUUM

Dataset and experimental setup

We used the BreastMNIST dataset from the MedMNIST [3] collection. It is a binary classification task on a set of 780 grey-scale breast tumors images with 28x28 pixels. The class distribution is 73/27. We experimented with different localities, and with versions of the images downscaled to 8x8 and 16x16 pixels

Two images from the training set of BreastMNIST (left: benign; right, malignant)



Results – runtime

Number of quantum circuit evaluations for Post-Variational QNNs

Observables variant

$$c_O = n \cdot \sum_{l=1}^{L} \binom{q}{l} \cdot 3^l$$

Ansatze variant

$$c_a = n \cdot \left(\binom{2q}{L} \cdot 2^L + 1 \right)$$

where n is the number of training samples, L is the locality, and q is the number of qubits (= image pixels). Locality is a parameter which is defined differently for each variant:

- For the observables variant, it is the size of each observed qubit subset
- For the ansatze variant, it is the number of parameterized gates set to a non-zero rotation

This parameter controls the complexity of the model, but more complex models require more computations, since all combinations are used as features.

As an illustration, the table here shows the number of quantum circuit evaluations for each training sample for the two variants with varying locality, with q = 8 qubits.

locality	observables	ansatze
1	24	33
2	276	481
3	1788	4481

Runtime in practice – IBM-Kobe

Since the quantum circuits use from 8 to 28 qubits, and there are 156 qubits available, it is possible to use circuit stacking in order to reduce the QPU load by computing fewer but wider circuits. Since QPU time is a scarce resource and a lot of time is spent waiting in queue or in pre-/post-processing, this helps reduce significantly the total runtime.

Median/maximum/minimum total runtime (in seconds) for circuits used in different configurations and using various numbers of qubits

	n qubits	Median	Minimum	Maximum
ans-8	88	21.2	18.7	95.3
obs-8	24	14.6	13.2	4432
obs-16	48	19.1	17.3	86.3
obs-28	84	24.6	22.6	33.6
obs-8-loc-2	106	20.5	16.7	1240

Using circuit stacking (with a maximum of 120 qubits), the total model training time ranged from **3.5 hours to 20 hours**, which makes it practically feasible on real quantum hardware. As a comparison, fully training a VQA in similar conditions would require an estimate of **over 10 days** of pure quantum computation time (excluding time in queue, circuit preparation, results processing).

Results – model quality

Although the main goal was to show that Post-Variational QNNs could be fully trained on real QPUs, it is also interesting to look at the quality of the models that are learned. To that intent, we present below the average (over 10 runs with different initial states) and standard deviation for the accuracy and area under ROC curve, on both the training and the test sets (random split). The configuration names contain the parameters values. For example, “obs-8-loc-2” refers to the *observables* variant with images downscaled to 8x8 pixels and locality set to 2. All models are run on IBM-Kobe, except “variational” which is the baseline VQA run on a noiseless simulator, and “obs-8-reimei” which is run on the ion-trap QPU Reimei.

	train	test
variational	0.54 ± 0.03	0.521 ± 0.065
ans-8	0.71 ± 0.02	0.74 ± 0.034
ans-16	0.721 ± 0.005	0.755 ± 0.002
obs-8	0.725 ± 0.004	0.756 ± 0.005
obs-8-reimei	0.724 ± 0.001	0.754 ± 0.013
obs-16	0.739 ± 0.017	0.765 ± 0.014
obs-28	0.73 ± 0.031	0.737 ± 0.046
obs-8-loc-2	0.756 ± 0.011	0.768 ± 0.019
obs-16-loc-2	0.788 ± 0.018	**0.787 ± 0.012**
obs-28-loc-2	**0.877 ± 0.072**	0.758 ± 0.011
obs-8-loc-3	0.756 ± 0.017	0.772 ± 0.022

	train	test
variational	0.495 ± 0.016	0.542 ± 0.031
ans-8	0.505 ± 0.01	0.539 ± 0.023
ans-16	0.485 ± 0.01	0.522 ± 0.026
obs-8	0.544 ± 0.102	0.559 ± 0.111
obs-8-reimei	0.525 ± 0.075	0.532 ± 0.106
obs-16	0.678 ± 0.068	0.679 ± 0.054
obs-28	0.627 ± 0.119	0.572 ± 0.099
obs-8-loc-2	0.718 ± 0.025	0.689 ± 0.014
obs-16-loc-2	0.777 ± 0.035	**0.71 ± 0.017**
obs-28-loc-2	**0.909 ± 0.075**	0.681 ± 0.021
obs-8-loc-3	0.721 ± 0.034	0.679 ± 0.015

Main observations

- We could only run one configuration on Reimei due to availability constraints, but the results are similar to those obtained on IBM-Kobe (see “obs-8” and “obs-8-reimei”), with the difference being well within the standard deviation
- We only ran two configurations of the ansatze variant since the results were not promising after “ans-16” and it is more costly to run than the other variant
- Overall, the observables variant with locality 2 running on 16x16 or 28x28 pixels images gave the best results, with the second one showing signs of overfitting.
- Going from locality 1 to 2 led to significantly better models, but using locality 3 does not seem to improve the results further despite the much larger number of features it uses

Perspectives

- ➢ Try feature selection to further reduce the number of quantum circuits evaluations required, aiming to only compute the useful features if possible
- ➢ Run experiments on larger and different datasets
- ➢ Compare with “state-of-the-art” QML methods on the standard train/test split

Acknowledgement

This poster is based on results obtained from a project, JPNP20017, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. 2021. Variational quantum algorithms. Nature Reviews Physics 3, 9 (2021), 625–644.

[2] Po-Wei Huang and Patrick Rebentrost. 2023. Post-variational quantum neural networks. arXiv preprint arXiv:2307.10560 (2023).

[3] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and BingbingNi.2023. MedMNISTv2-A large-scale lightweight benchmark for 2D and 3D biomedical image classification. Scientific Data 10, 1 (2023), 41.