Challenges & Opportunities in Quantum Applications

Sonika Johri

Coherent Computing

Next Steps in Quantum Computing, May 18

Overview

Where are we?

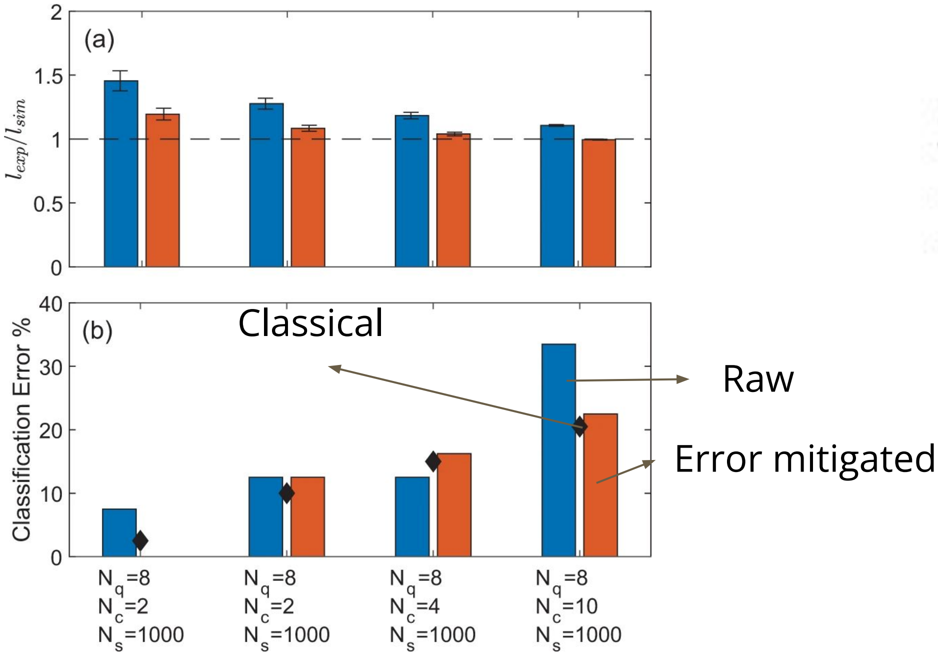
- Recent end-to-end demonstrations of quantum applications

What do we need to utilize 50-500 qubits?

- Benchmarking
- Identifying practical quantum advantage
- Software stack
- High-level quantum programming framework

Some recent demos in quantum machine learning

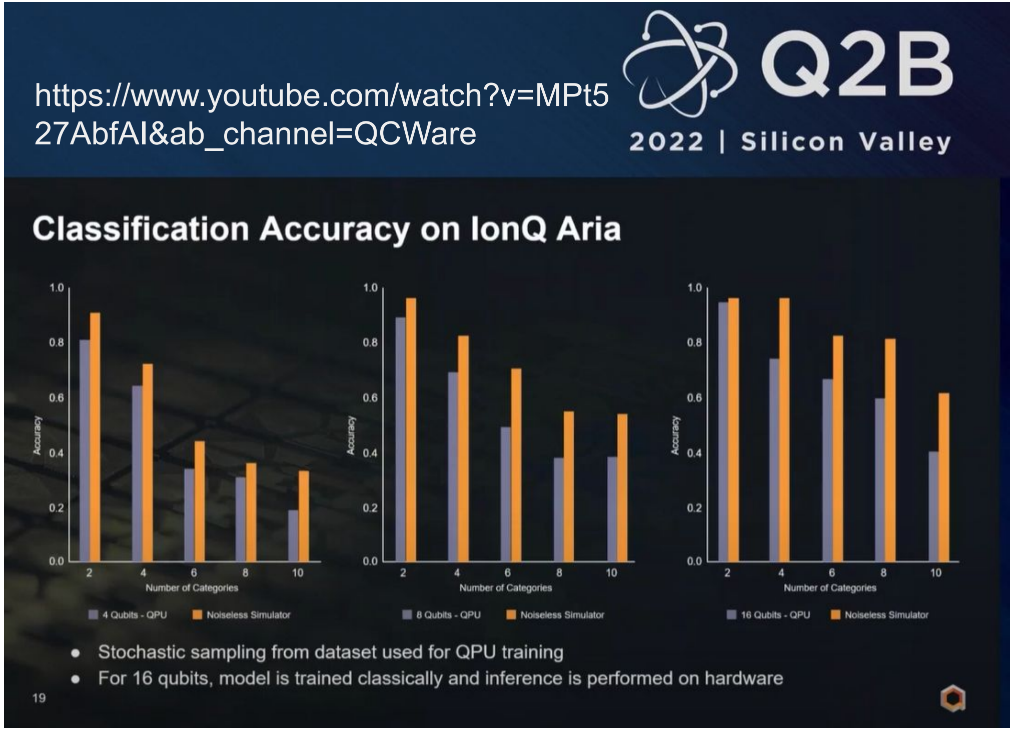
Dataset: MNIST,
Algorithm: Quantum Nearest Centroid
Hardware: IonQ Harmony

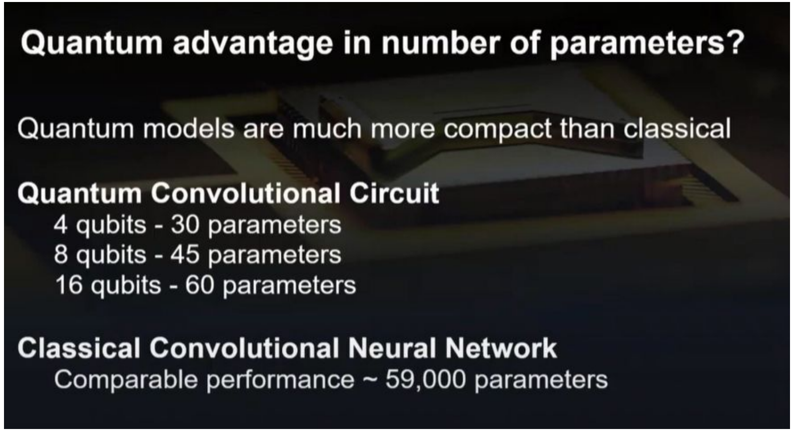


Classical runtime: O(nkd)
Quantum runtime: O(kd + nd + kn log(d)/eps)

npj Quantum Information volume 7, Article number: 122 (2021)

Dataset: German Road Sign
Algorithm: Quantum CNN
Hardware: IonQ Aria





Dataset: Stock Prices
Algorithm: Quantum Copula-based GAN
Hardware: IonQ Harmony

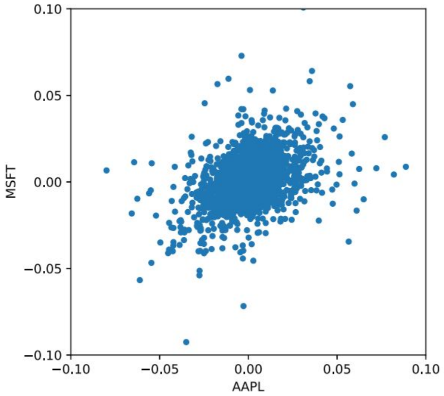


TABLE I. KS statistics and p-value of KS test across multiple models. The quantum models use $N_q = 6$ qubits.

Model	D_{KS} (the smaller the better)	p-value (threshold 0.05)
Parametric model	0.0449	0.117
Classical GAN	0.0363–0.0508	0.0530–0.309
QGAN simulation	0.0320–0.0396	0.226–0.473
QGAN experiment, QPU cloud	0.0352	0.3570
QCBM simulation	0.0425–0.0520	0.0511–0.1717
QCBM experiment, QPU cloud	0.0373–0.0515	0.0548–0.3030
QCBM experiment, QPU Next Gen	0.0330–0.0510	0.0578–0.4465

Physical Review Research 4, 043092 (2022)

Dataset: Stock Prices
Algorithm: Quantum Copula-based GAN
Hardware: IonQ Harmony

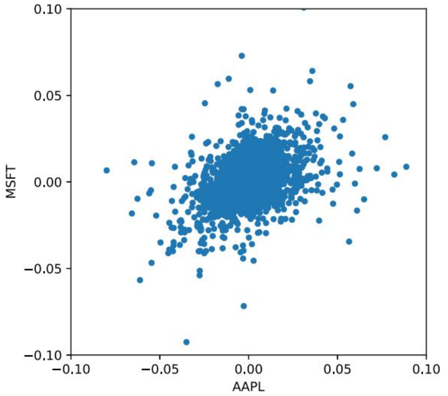


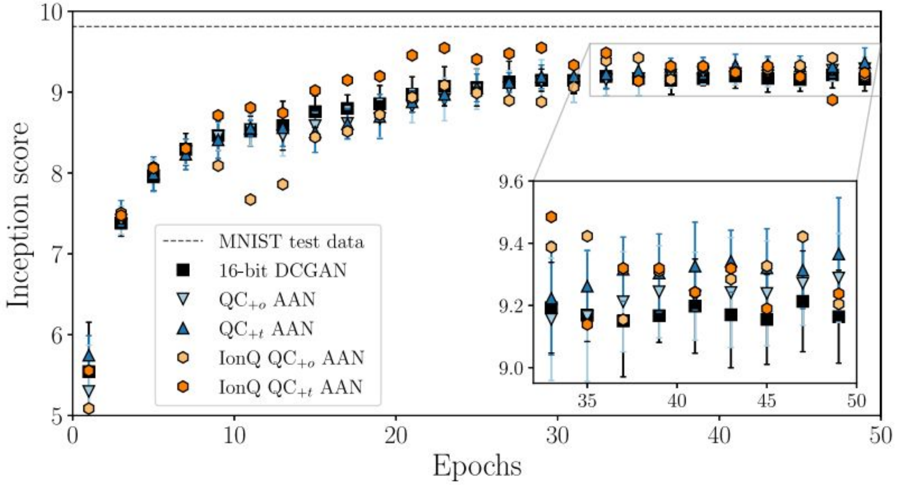
TABLE I. KS statistics and p-value of KS test across multiple models. The quantum models use $N_q = 6$ qubits.

Model	D_{KS} (the smaller the better)	p-value (threshold 0.05)
Parametric model		
Classical GAN		
QGAN simulation		
QGAN experiment, QPU cloud		
QCBM simulation		
QCBM experiment, QPU cloud		
QCBM experiment, QPU Next Gen		

We also note that we are able to train QGAN/QCBM at a much faster learning rate and therefore conclude the training with much fewer iterations than classical GAN. In classical GAN, the learning rate used is 0.0001 and model training concludes after 20 000 iterations. Attempts to increase the learning rate failed due to nonconvergence in model training. In QGAN, model training concludes after 1000 iterations. In QCBM for 6 qubits, the training converges to a good value for as little as 20 iterations.

Physical Review Research 4, 043092 (2022)

Dataset: MNIST
Algorithm: QC-AAN
Hardware: IonQ Harmony



8-qubit experiment on IonQ

QC_{+o} AAN
$IS = 9.43 \pm 0.02$

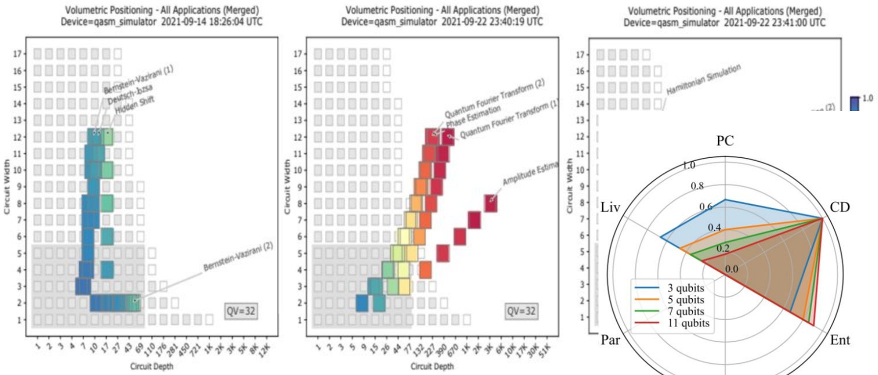
QC_{+t} AAN
$IS = 9.54 \pm 0.02$

Physical Review X 12, 031010 (2022)

Next Steps

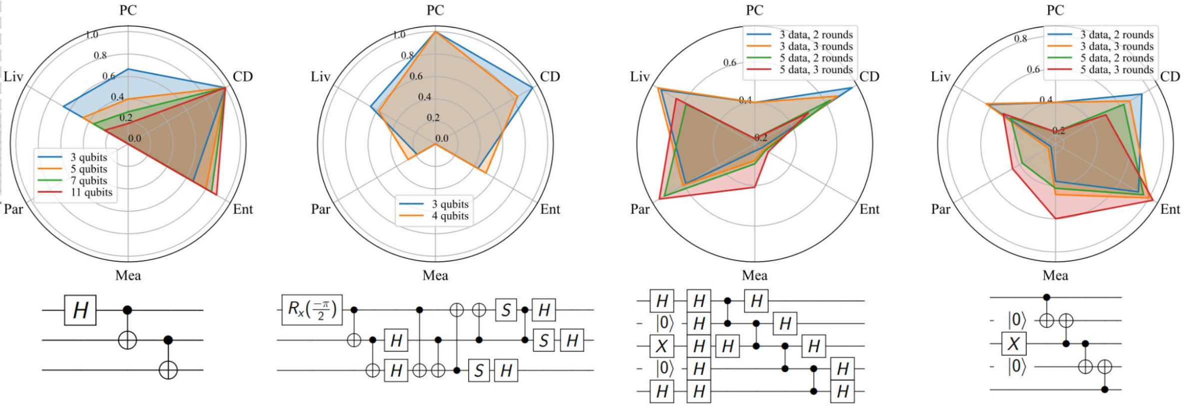
Benchmarking

Benchmarking framework & theory needed for rapidly evolving applications and hardware



QEDC benchmarking suite, arXiv:2110.03137

SupermarQ, arXiv: 2202.11045



(a) GHZ (b) Mermin-Bell (c) Phase Code (d) Bit Code

Identifying Practical Quantum Advantage

- Quantum advantage in practice, vs just pure math
- Heuristic algorithms harder to prove but will be the main hope for quantum advantage with near-term hardware
- For instance, in QML, variational ansatz can capture correlations that are hard for classical, leading to
  - ❖ Fewer iterations to train
  - ❖ Faster inference
  - ❖ Better accuracy in the tails
  - ❖ Use fewer parameters in the model
  - ❖ Better at generalizing to unseen data
  - ❖ Better at predicting outliers

Software stack that is efficient and robust

- Hardware architecture is evolving fast (new devices every ~6 months) + different platforms have different underlying physical operations -> want to avoid user having to customize and update programs constantly -> need Intermediate Representations
- Hardware errors can be complex and even time-dependent -> how far can software stack compensate for that?
- Hybrid quantum-classical -> Lots of classical tools needed: Ex. optimizers that use few evaluations & are robust to fluctuations; circuit knitting tools; computing on quantum networks; asynchronous execution; mid-circuit measurement handling

High-level quantum programming framework

- Right now, quantum compilation refers to circuit transformations + mapping and scheduling to hardware
- Do we need to move away from circuit model at the user level to program 50-500 qubits?
- Instead:
  - ❖ Quantum data structures, Ex. Hilbert spaces, Bell states, reduced density matrices with specified entanglement spectrum, matrix product states, tensor networks, ground states
  - ❖ Quantum operations: Ex. Hamiltonian evolution, oracle operations, amplitude amplification, quantum signal processing
  - ❖ User has access to these abstraction in the programming framework
  - ❖ Compiler synthesizes efficient circuits to realize these data structures and operations