QOC: Quantum On-Chip Training with Parameter Shift and Gradient Pruning

Hanrui Wang[1*], Zirui Li[2*], Jiaqi Gu[3], Yongshan Ding[4], David Z. Pan[3], Song Han[1]

[1]Massachusetts Institute of Technology, [2]Shanghai Jiao Tong University, [3]University of Texas at Austin, [4]Yale University

Outline

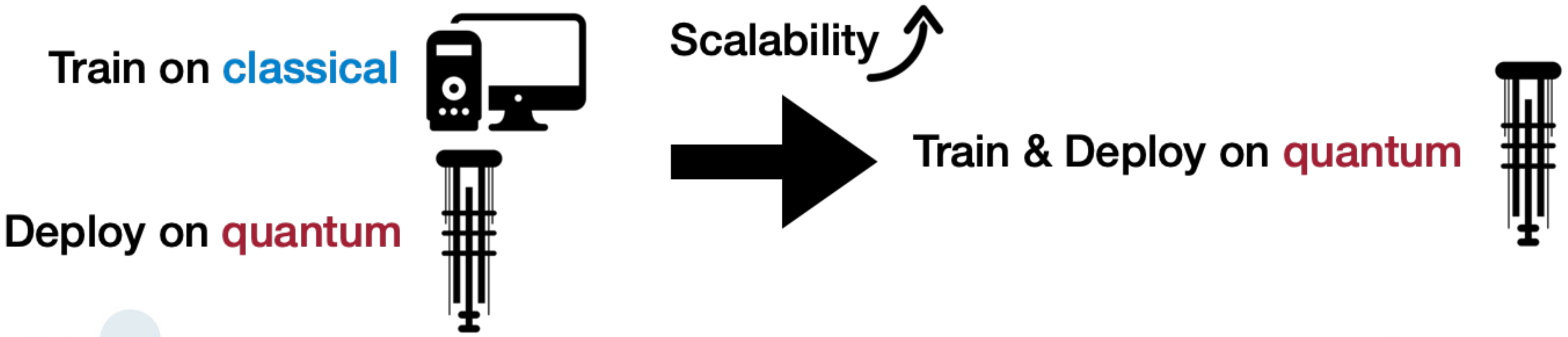
- Overview
- Background
- QOC Methodology
- Evaluation
- TorchQuantum Library
- Conclusion

Outline

- Overview
- Background
- QOC Methodology
- Evaluation
- TorchQuantum Library
- Conclusion

QOC Overview

- Conventional: train on **classical** simulator
  - Unscalable
- QOC: train on **quantum** machine
  - Calculates gradients on **quantum** machines
  - **Scalable**

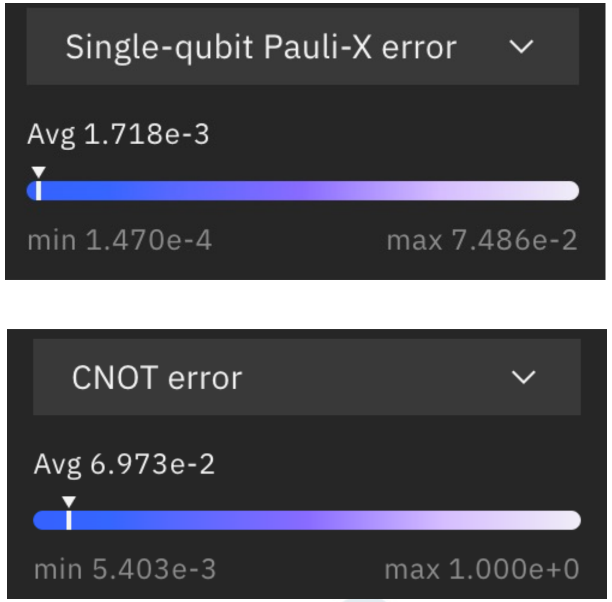


Outline

- Overview
- Background
- QOC Methodology
- Evaluation
- TorchQuantum Library
- Conclusion

NISQ Era

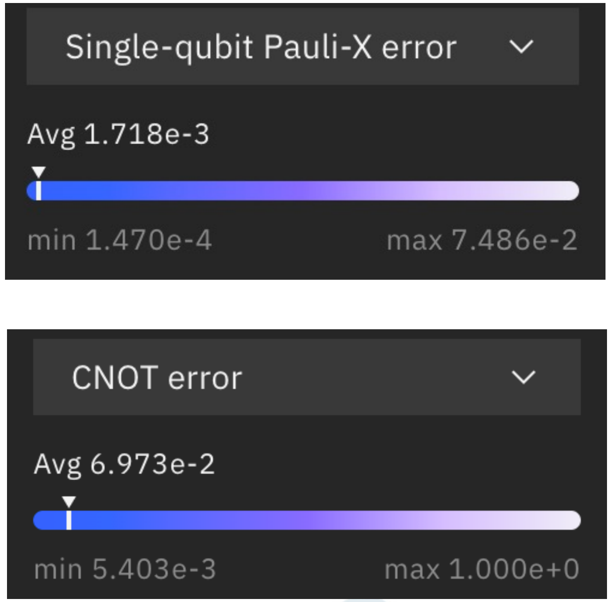
- Noisy Intermediate-Scale Quantum (NISQ)
  - **Noisy**: qubits are sensitive to environment; quantum gates are unreliable



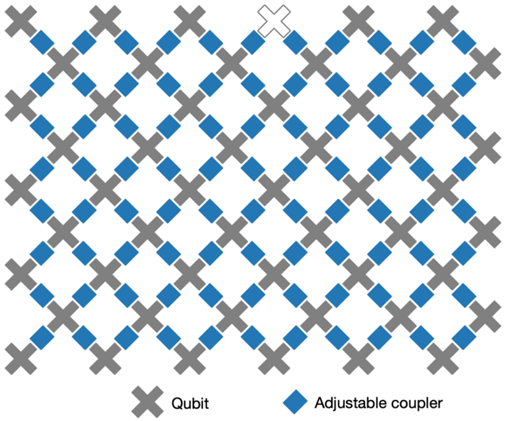
Gate Error Rate
https://quantum-computing.ibm.com/

NISQ Era

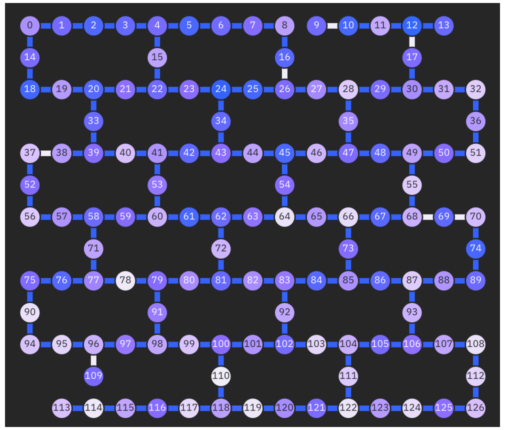
- Noisy Intermediate-Scale Quantum (NISQ)
  - **Noisy**: qubits are sensitive to environment; quantum gates are unreliable
  - **Limited number** of qubits: tens to hundreds of qubits



Gate Error Rate
https://quantum-computing.ibm.com/



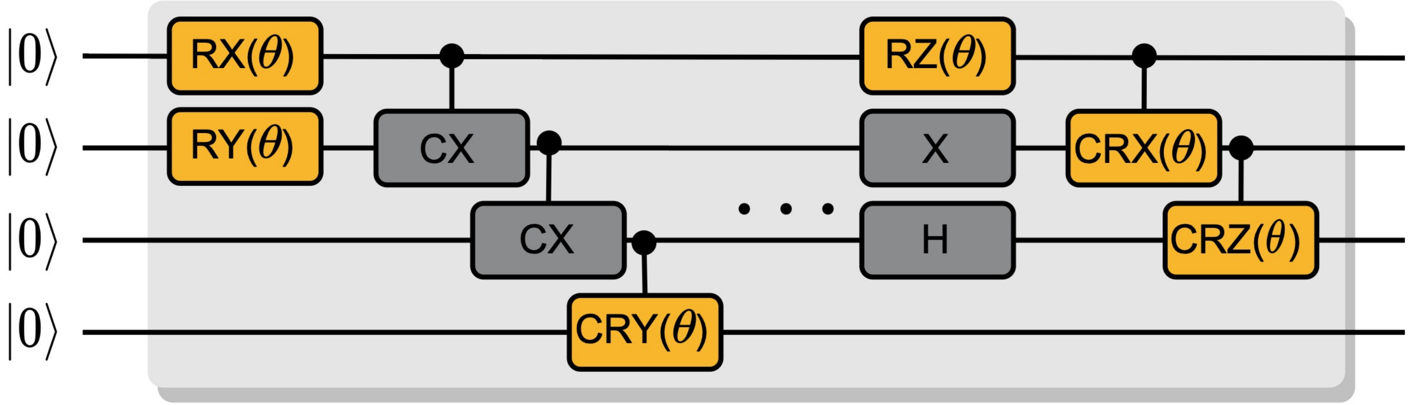
Google Sycamore
https://www.nature.com/articles/s41586-019-1666-5



IBM Washington
https://quantum-computing.ibm.com/

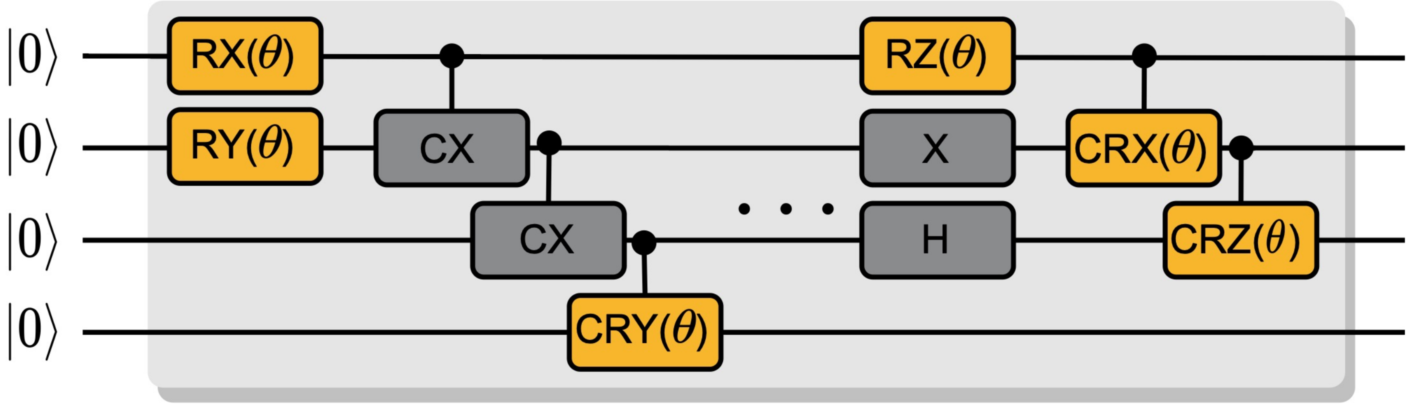
Parameterized Quantum Circuits (PQC)

- Parameterized Quantum Circuits (PQC)
- Quantum circuit with fixed gates and parameterized gates



Parameterized Quantum Circuits (PQC)

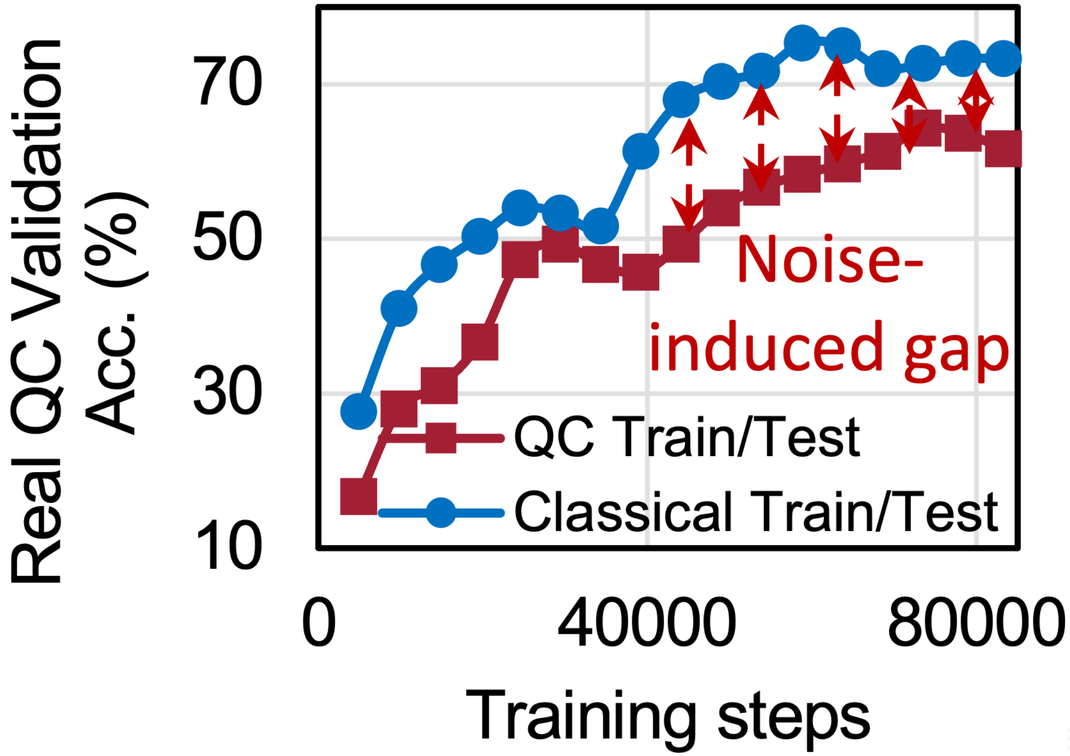
- Parameterized Quantum Circuits (PQC)
- Quantum circuit with fixed gates and parameterized gates



- PQCs are commonly used in **hybrid classical-quantum** models and show promises to achieve quantum advantage
  - Variational Quantum Eigensolver (VQE)
  - Quantum Neural Networks (QNN)
  - Quantum Approximate Optimization Algorithm (QAOA)

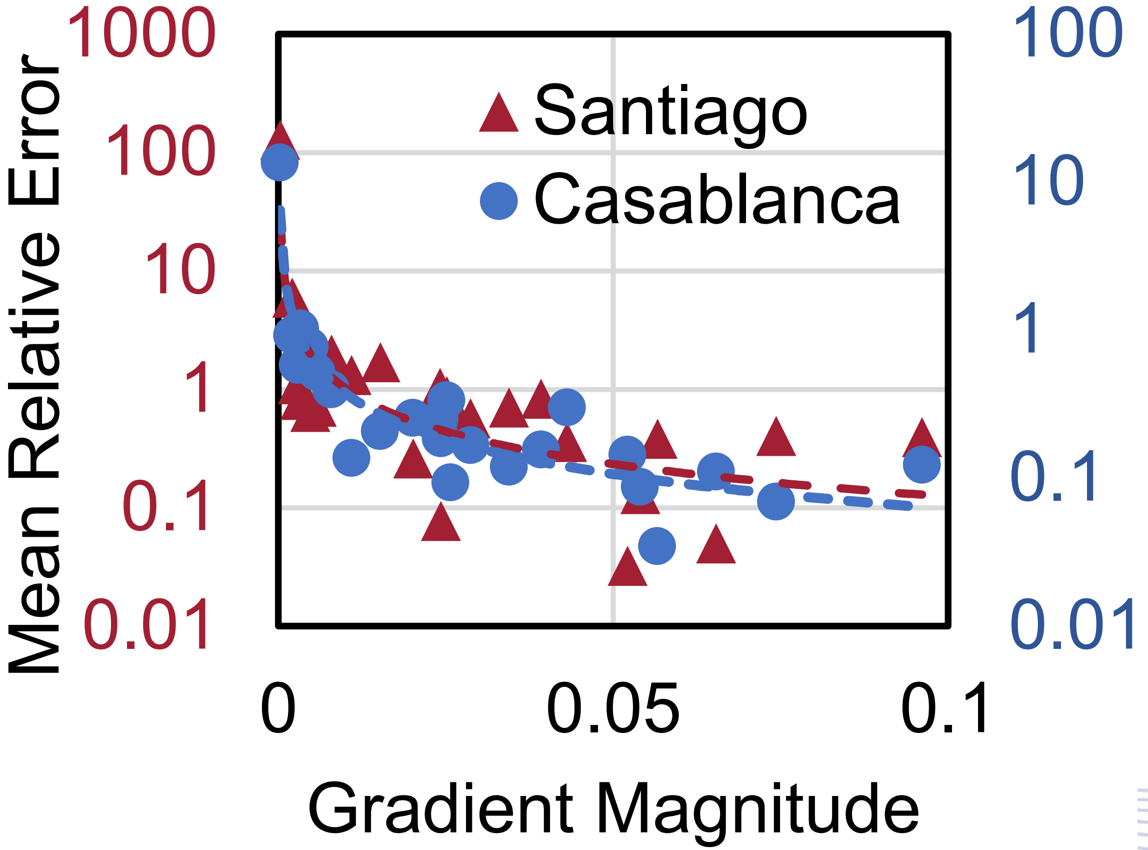
Challenge of On-chip Training: noise

- Noise **reduces reliability** of on-chip computed gradients



Challenge of On-chip Training: noise

- Noise **reduces reliability** of on-chip computed gradients
- **Small** magnitude gradients have **large** relative errors

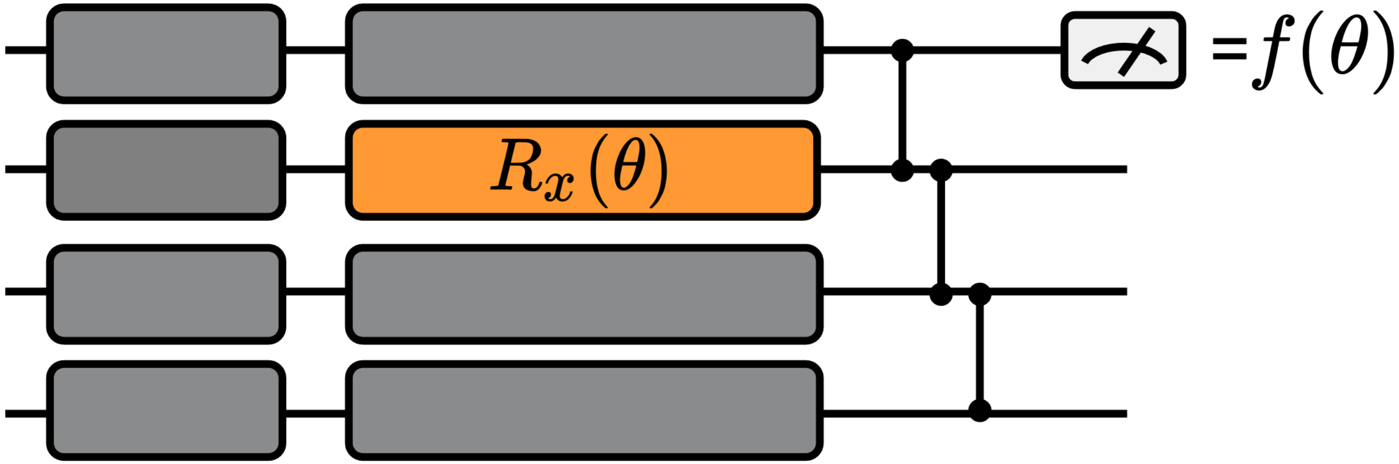


Outline

- Overview
- Background
- QOC Methodology
- Evaluation
- TorchQuantum Library
- Conclusion

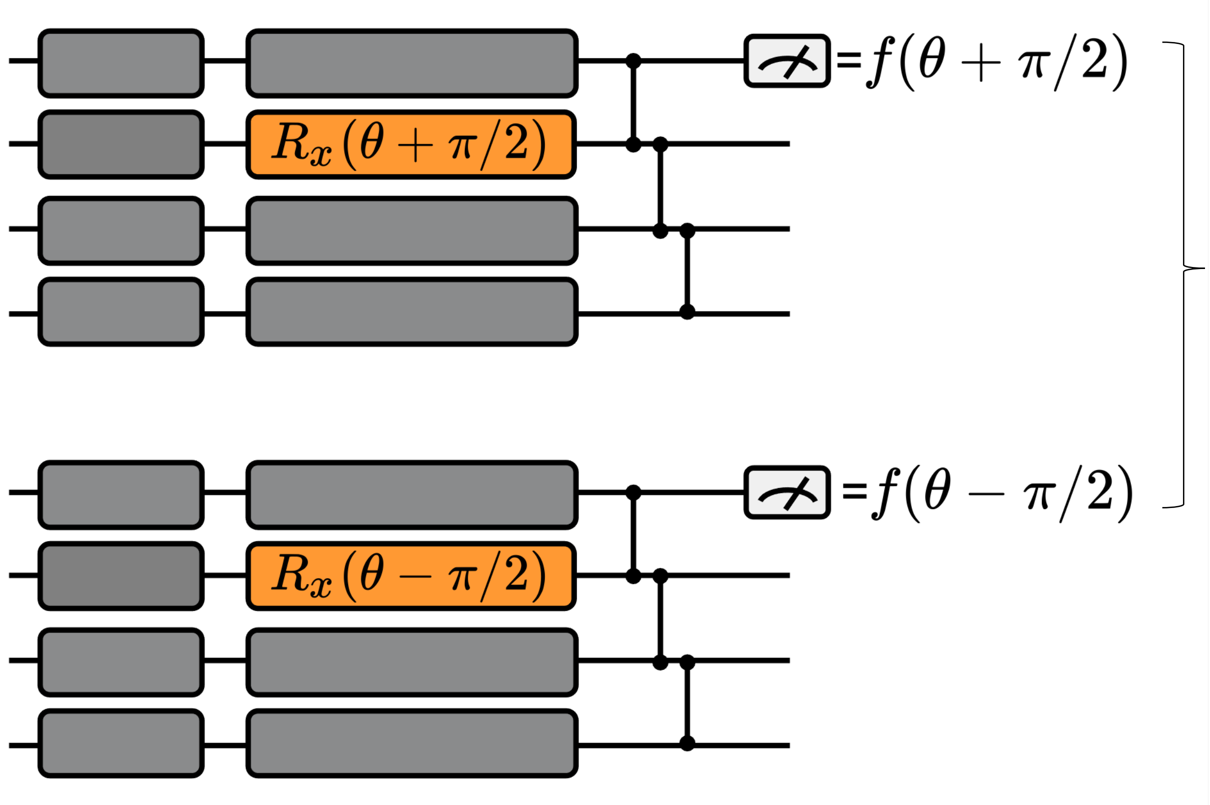
Parameter Shift Rules

- Calculate the gradient of θ w.r.t. $f(\theta)$.



Parameter Shift Rules

- Shift θ twice



$$\frac{\partial}{\partial\theta}f(\theta)=\frac{1}{2}\left(f\left(\theta+\frac{\pi}{2}\right)-f\left(\theta-\frac{\pi}{2}\right)\right)$$

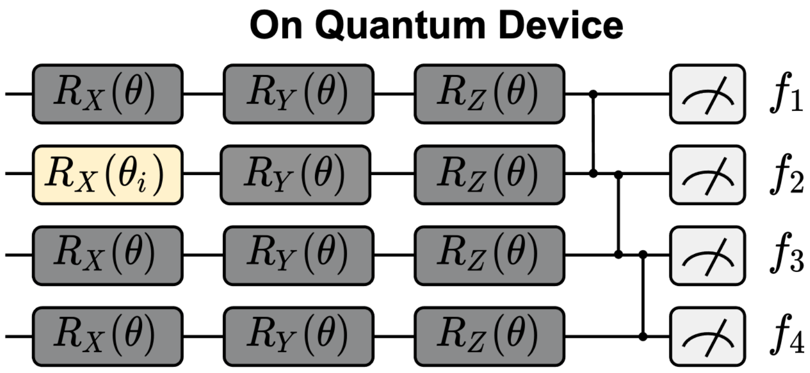
Parameter Shift Rules

- This formula is valid to all rotation gates
  - RZ, RY, RX, RXX, RZZ
- One gradient requires two runs on real quantum machine

$$\frac{\partial}{\partial\theta} f(\theta) = \frac{1}{2}\left(f\left(\theta + \frac{\pi}{2}\right) - f\left(\theta - \frac{\pi}{2}\right)\right)$$

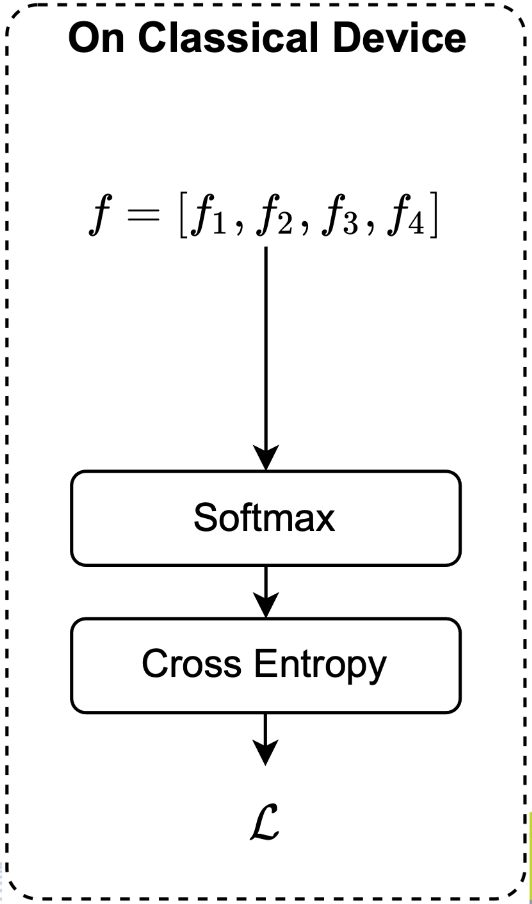
Calculate Gradients of PQC

- Step 1: Run on QC without shift to obtain f



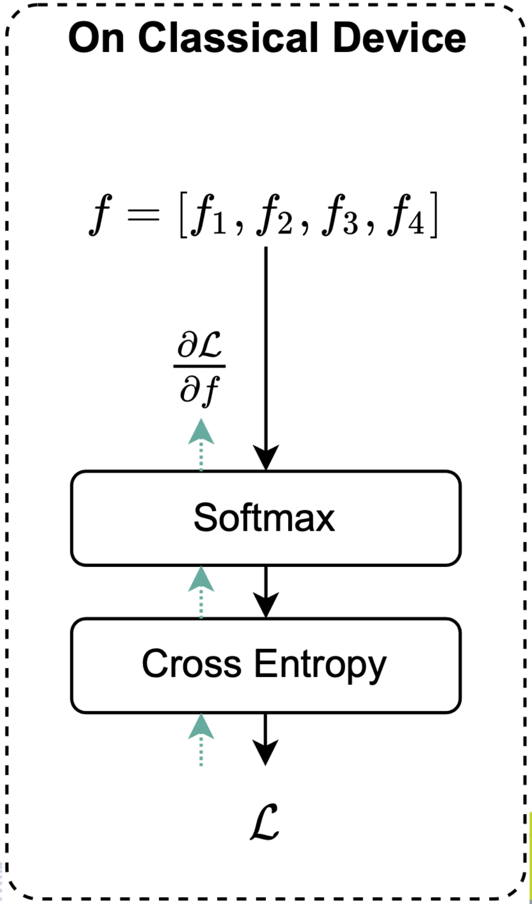
Calculate Gradients of PQC

- Step 2: Further forward to get $Loss$



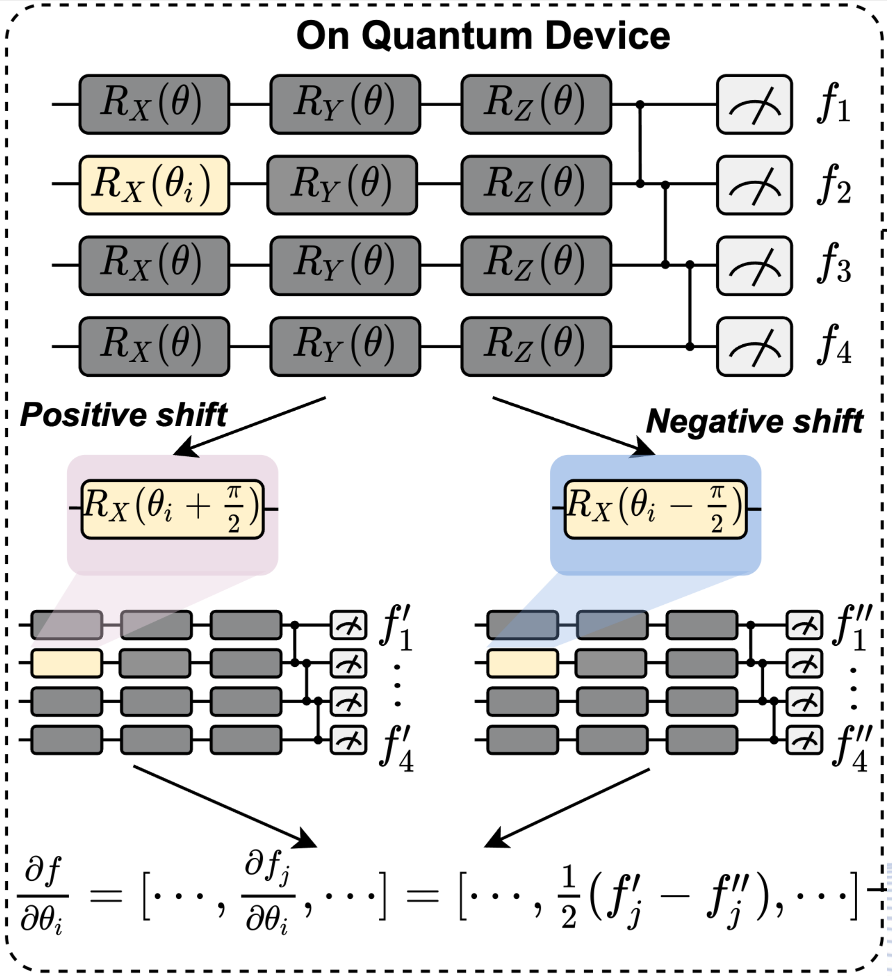
Calculate Gradients of PQC

- Step 3: Backpropagation to calculate $\frac{\partial Loss}{\partial f(\theta)}$



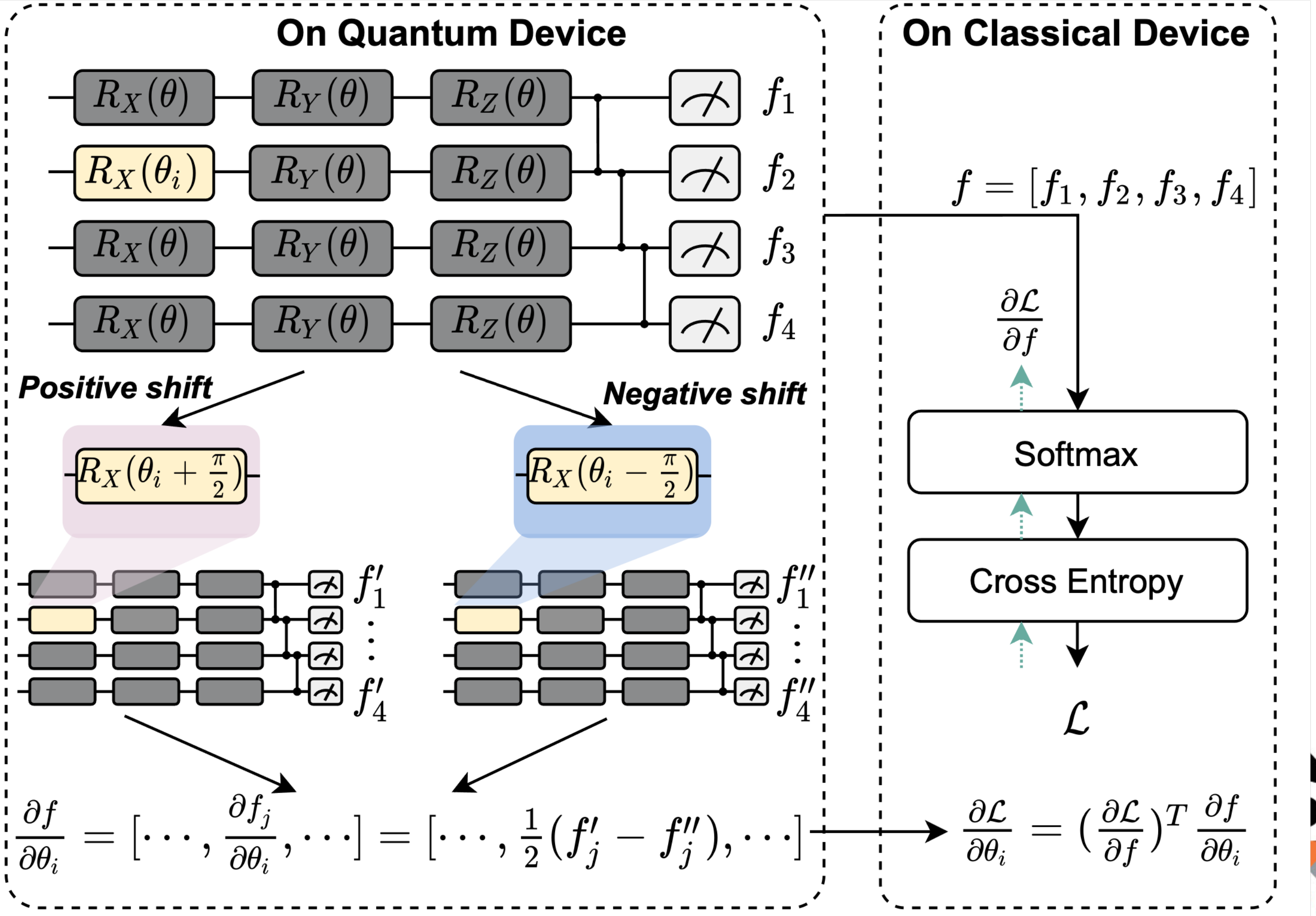
Calculate Gradients of PQC

- Step 4: Shift twice and run on QC to calculate $\frac{\partial f(\theta)}{\partial \theta_i}$



Calculate Gradients of PQC

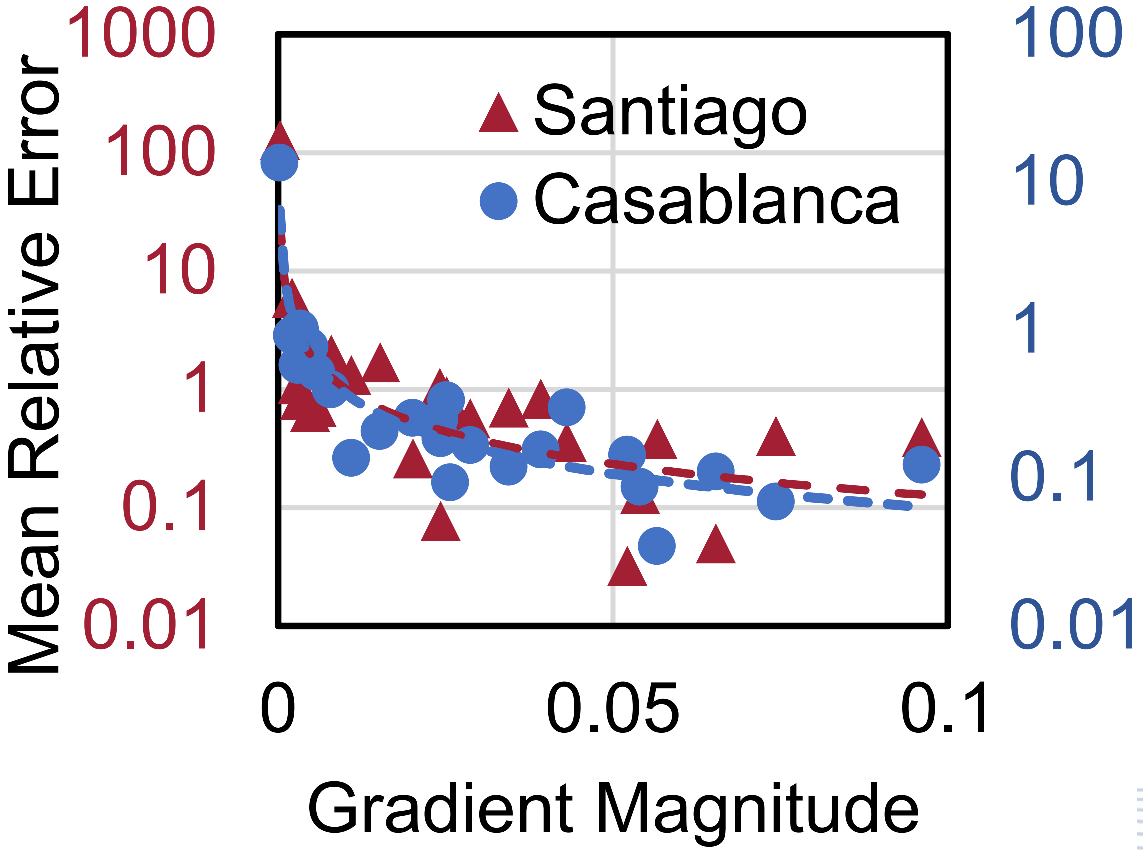
- Step 5: By Chain Rule: $\frac{\partial Loss}{\partial f(\theta)} \frac{\partial f(\theta)}{\partial \theta_i} = \frac{\partial Loss}{\partial \theta_i}$, sum over 4 passes (4 qubits)



DESIGN AUTOMATION CONFERENCE

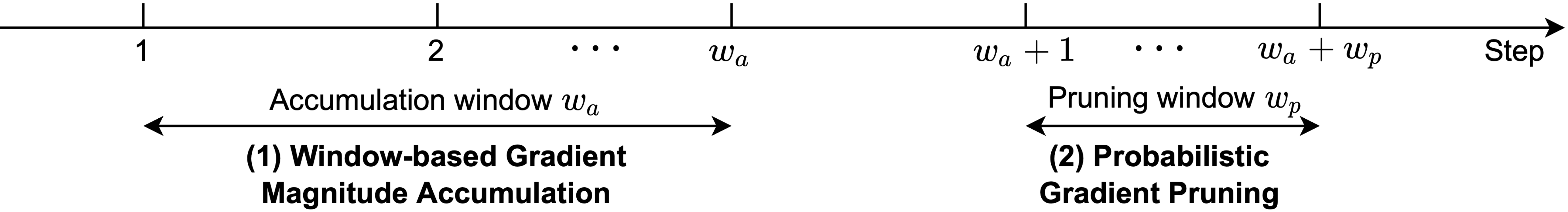
Probabilistic Gradient Pruning

- **Small** magnitude gradients have **large** relative errors



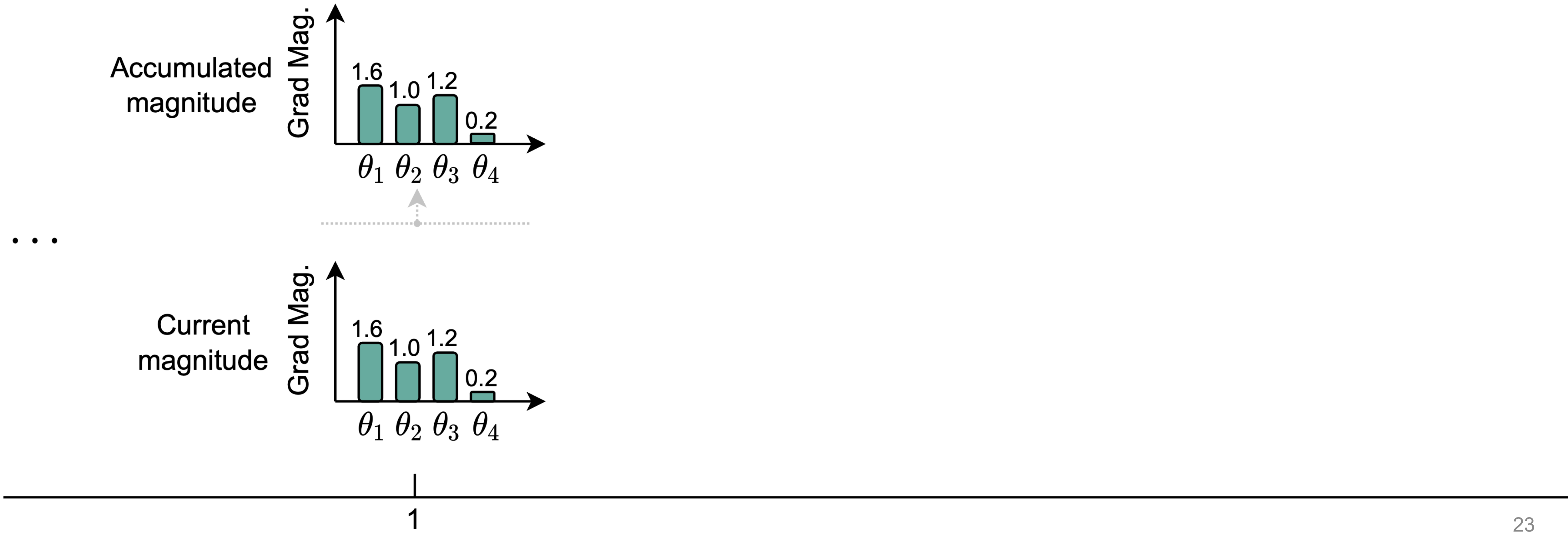
Probabilistic Gradient Pruning

- Accumulation Window followed by Pruning Window repeatedly



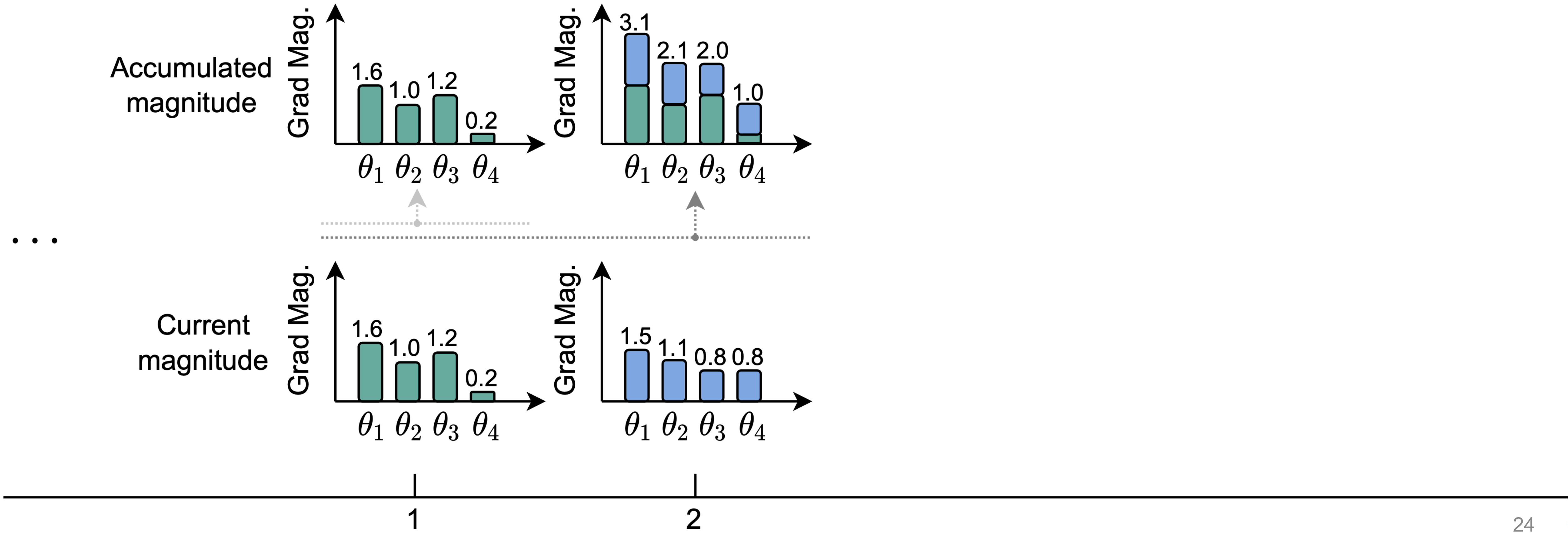
Accumulation Window

- Keep a record of accumulated gradient magnitude



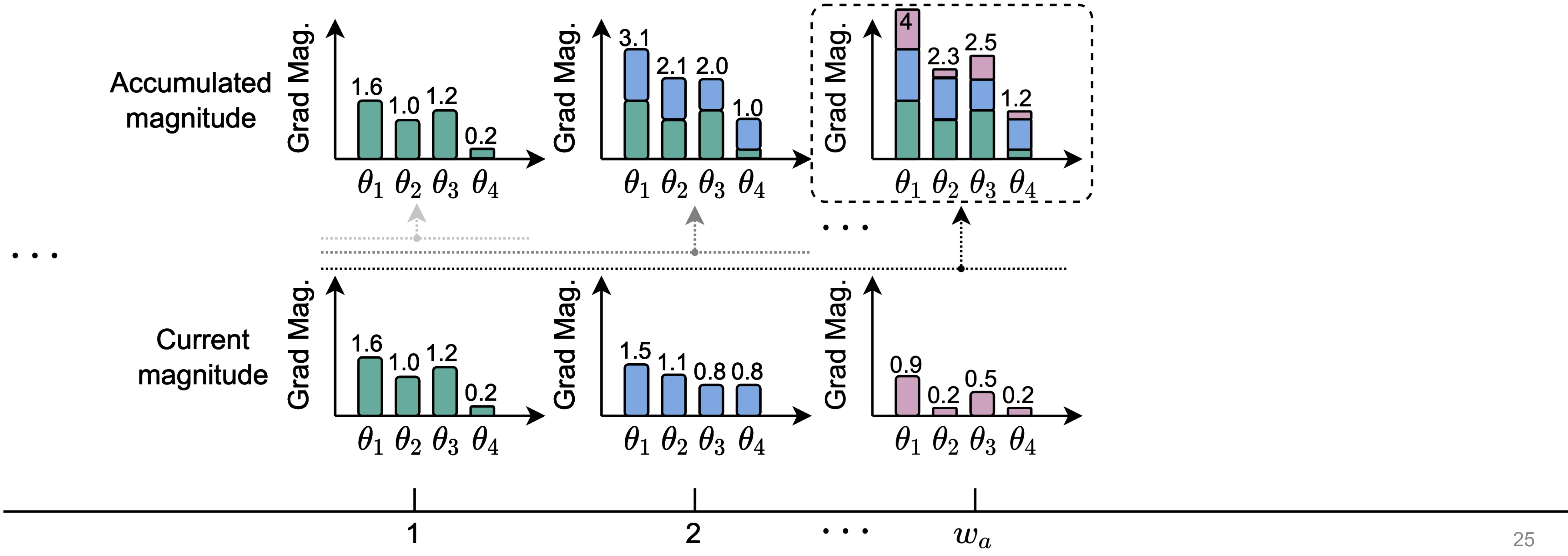
Accumulation Window

- Keep a record of accumulated gradient magnitude.



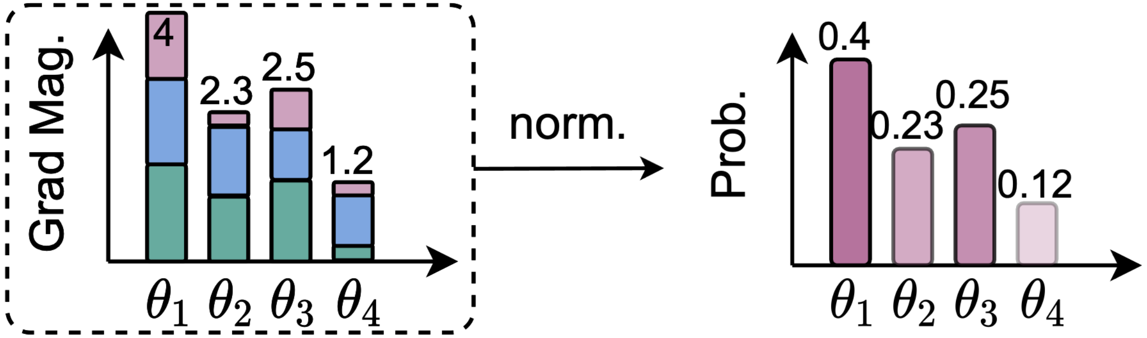
Accumulation Window

- Keep a record of accumulated gradient magnitude



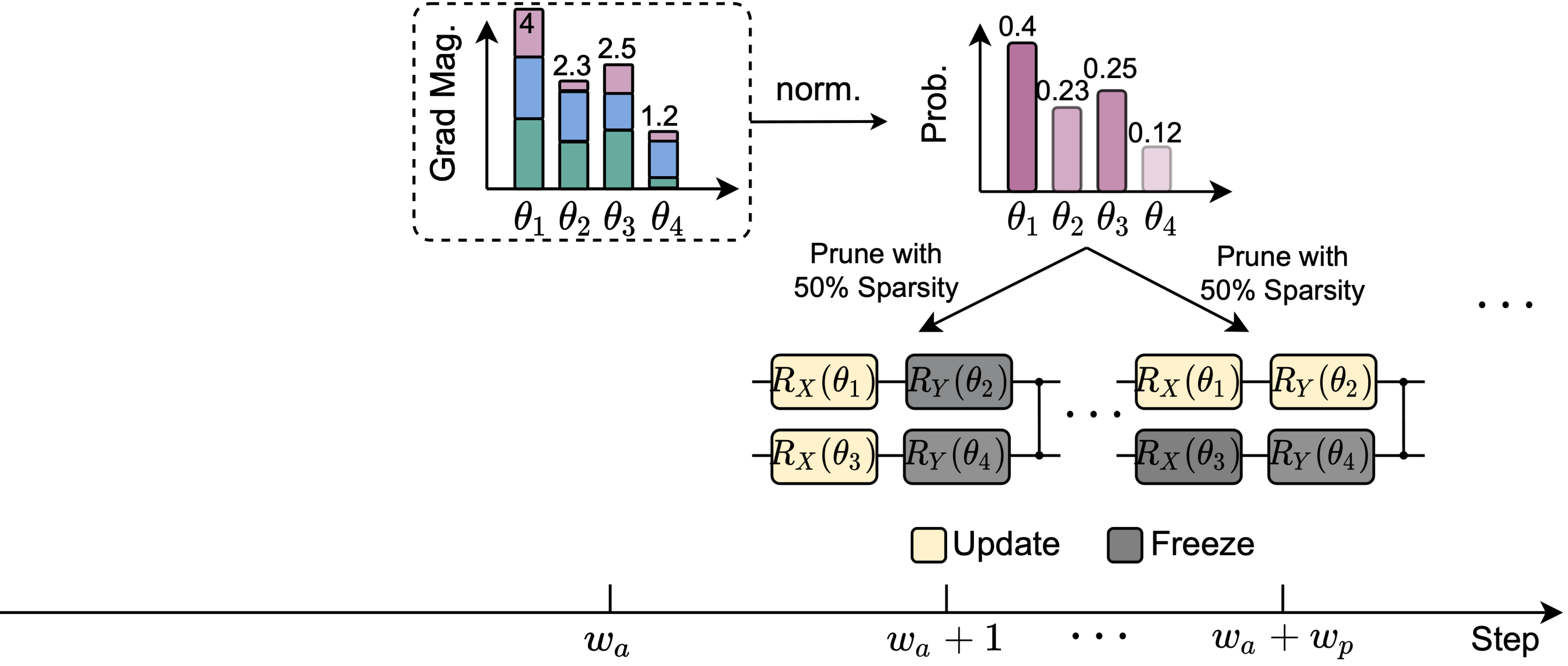
Pruning Window

- Normalize the accumulated gradient magnitude to a probability distribution



Pruning Window

- Prune the calculation of some gradients according to the probability distribution



Outline

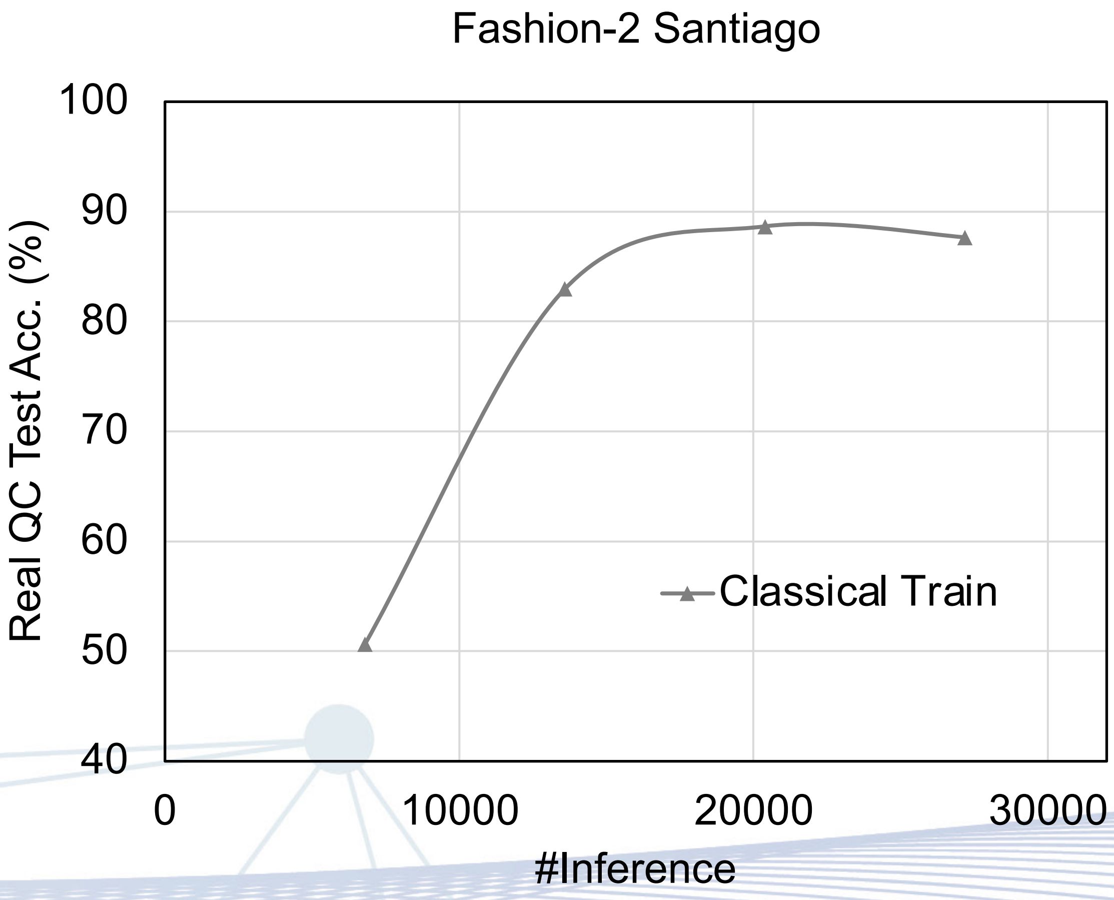
- Overview
- Background
- QOC
- Evaluation
- TorchQuantum Library
- Conclusion

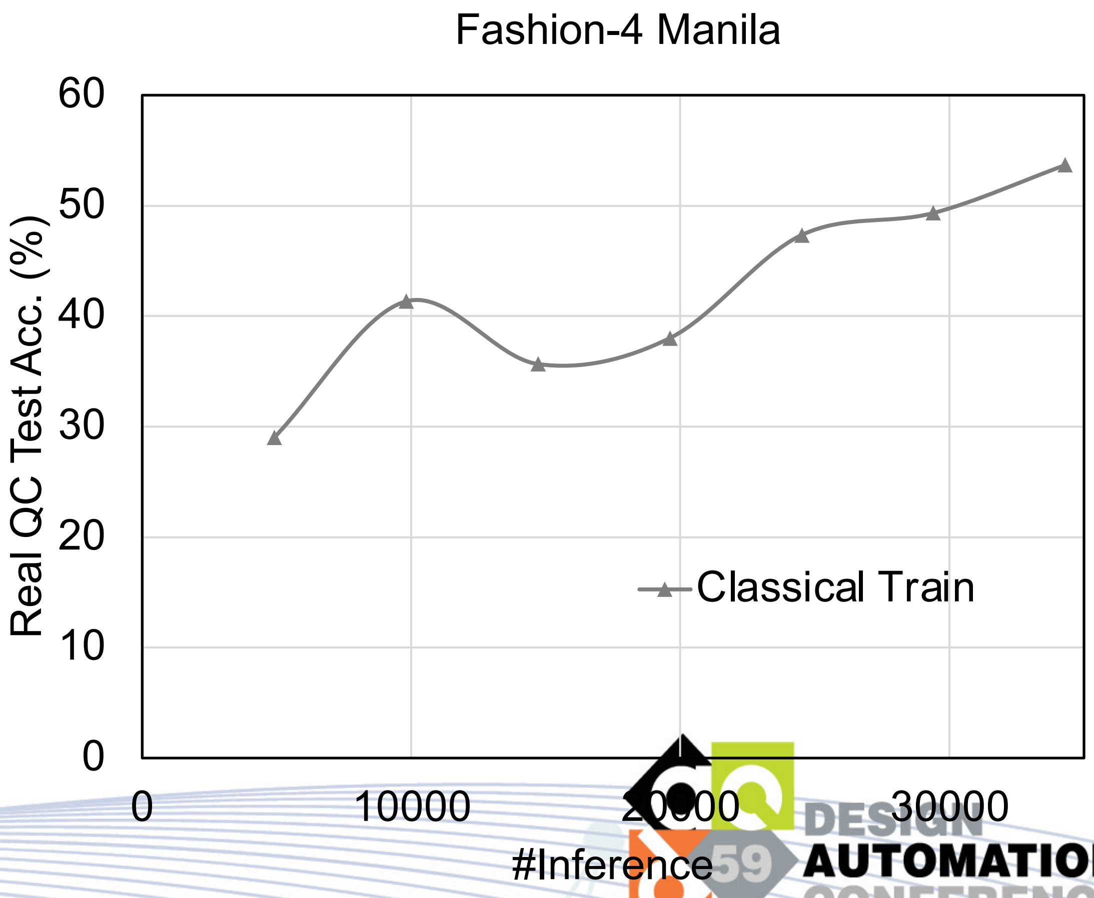
Evaluation

- Benchmarks
  - Quantum Machine Learning task: MNIST 4-class, 2-class, Fashion MNIST 4-class, 2-class, Vowel 4-class
  - Variational Quantum Eigensolver task: H2 molecule
- Quantum Devices
  - IBMQ
  - #Qubits: 5 to 7
  - Quantum Volume: 8 to 32
- Circuit architecture
  - RZZ+RY, RXYZ+CZ, RZX+RXX

QNN Training Curves

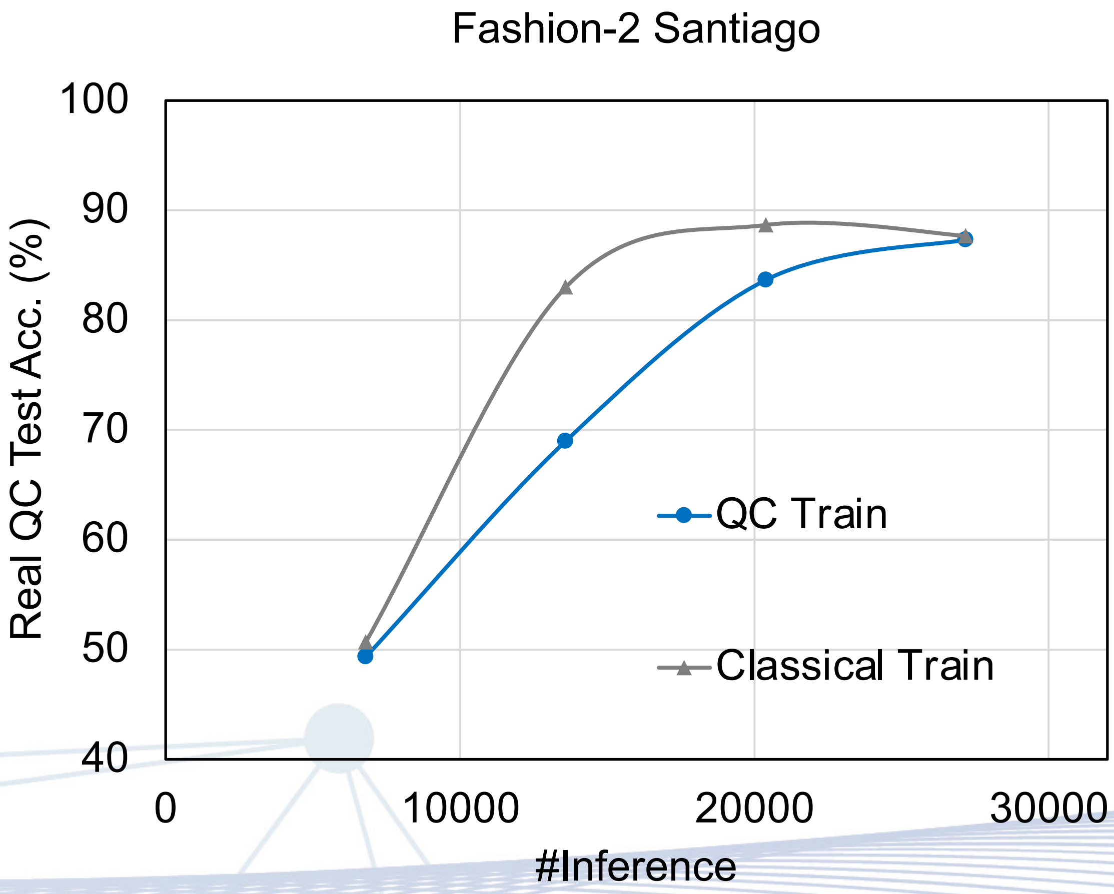
- Classical Train:
  - Train on classical simulator and test on real QC

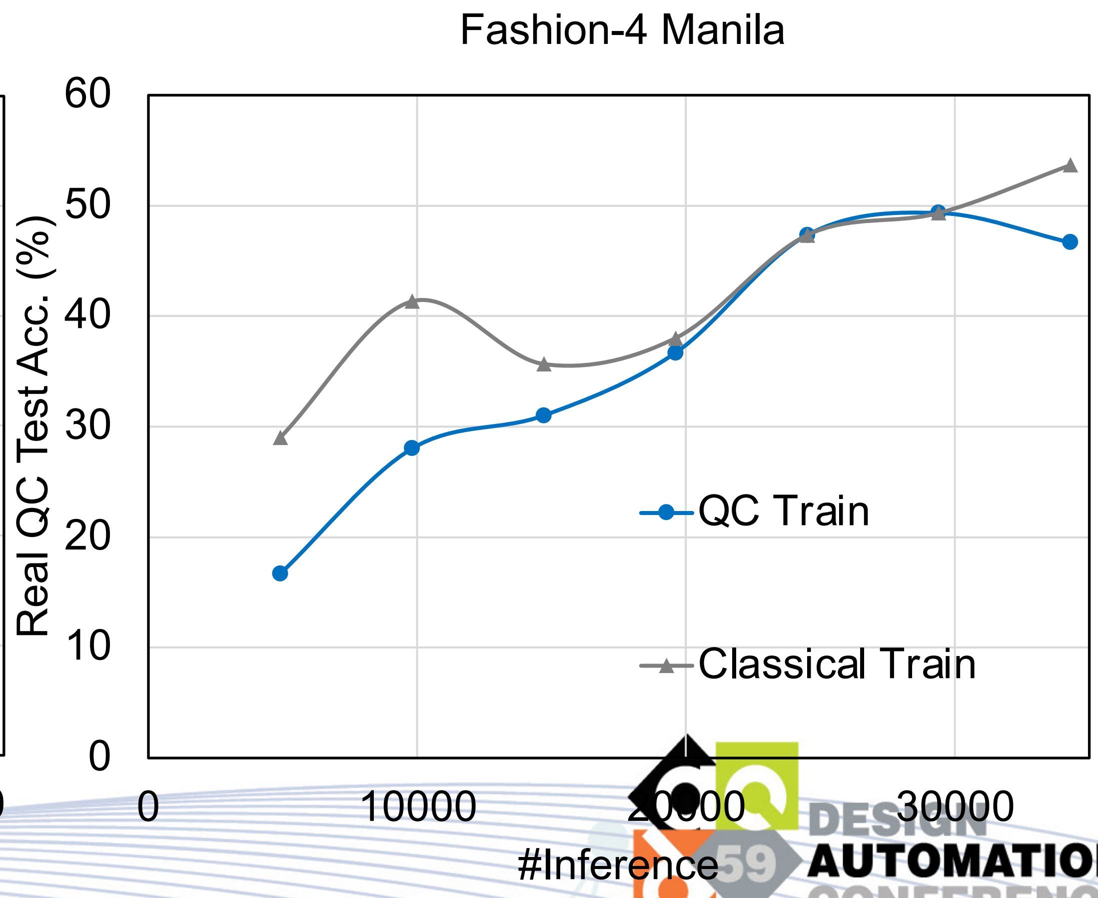




QNN Training Curves

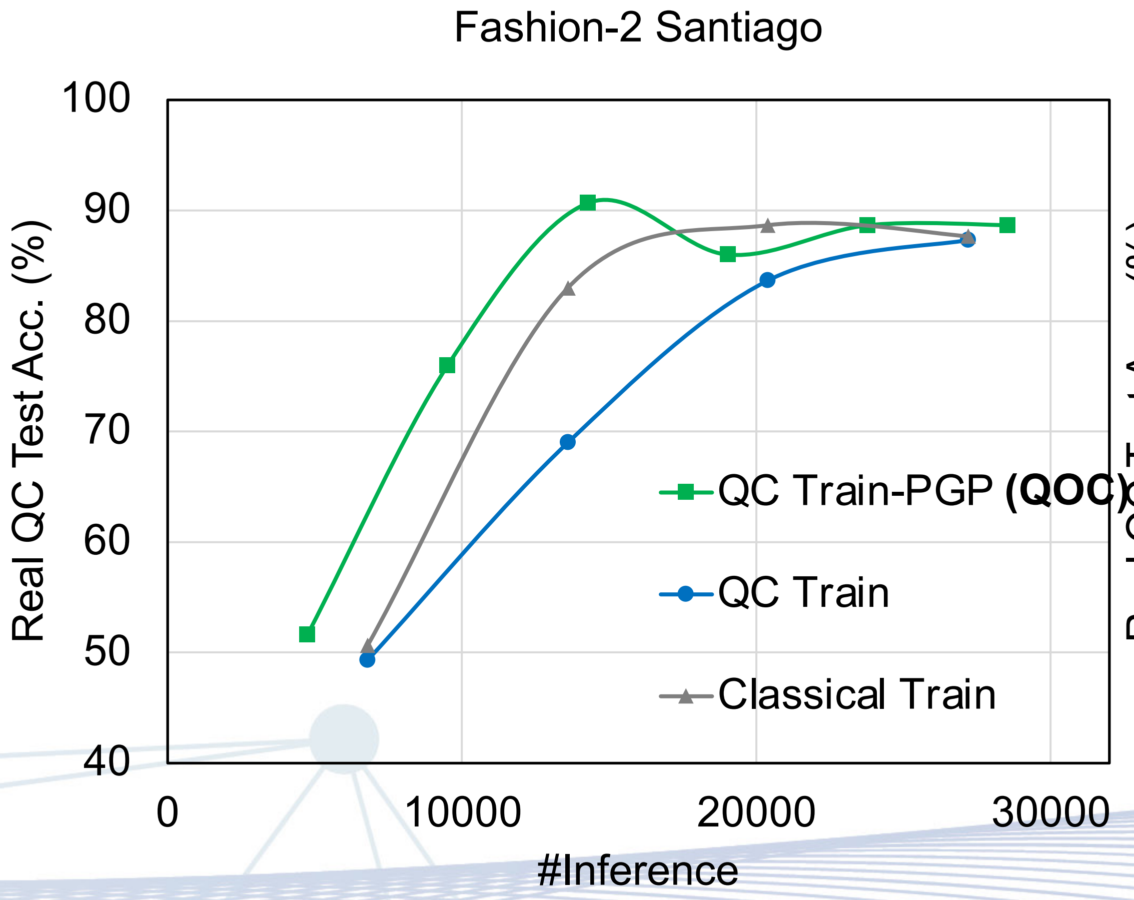
- QC Train:
  - Train and test the model on real QC

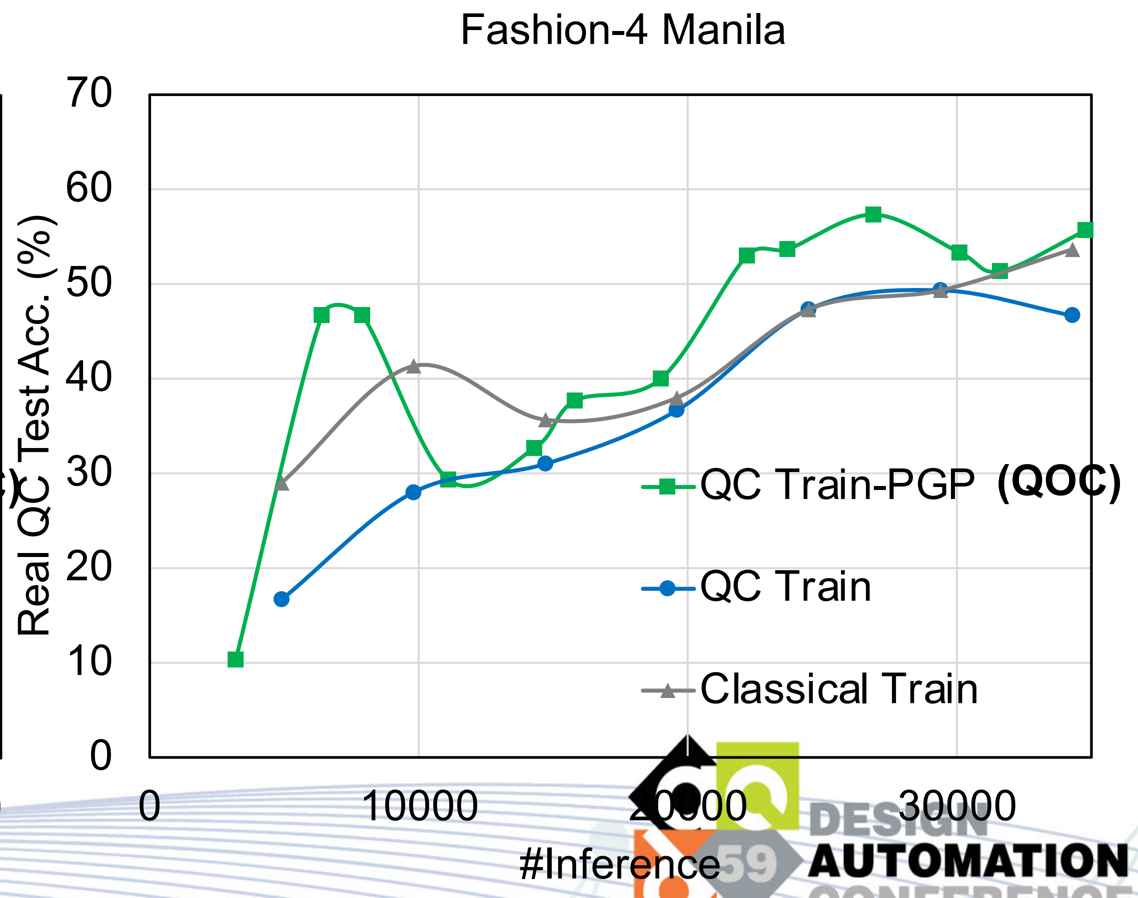




QNN Training Curves

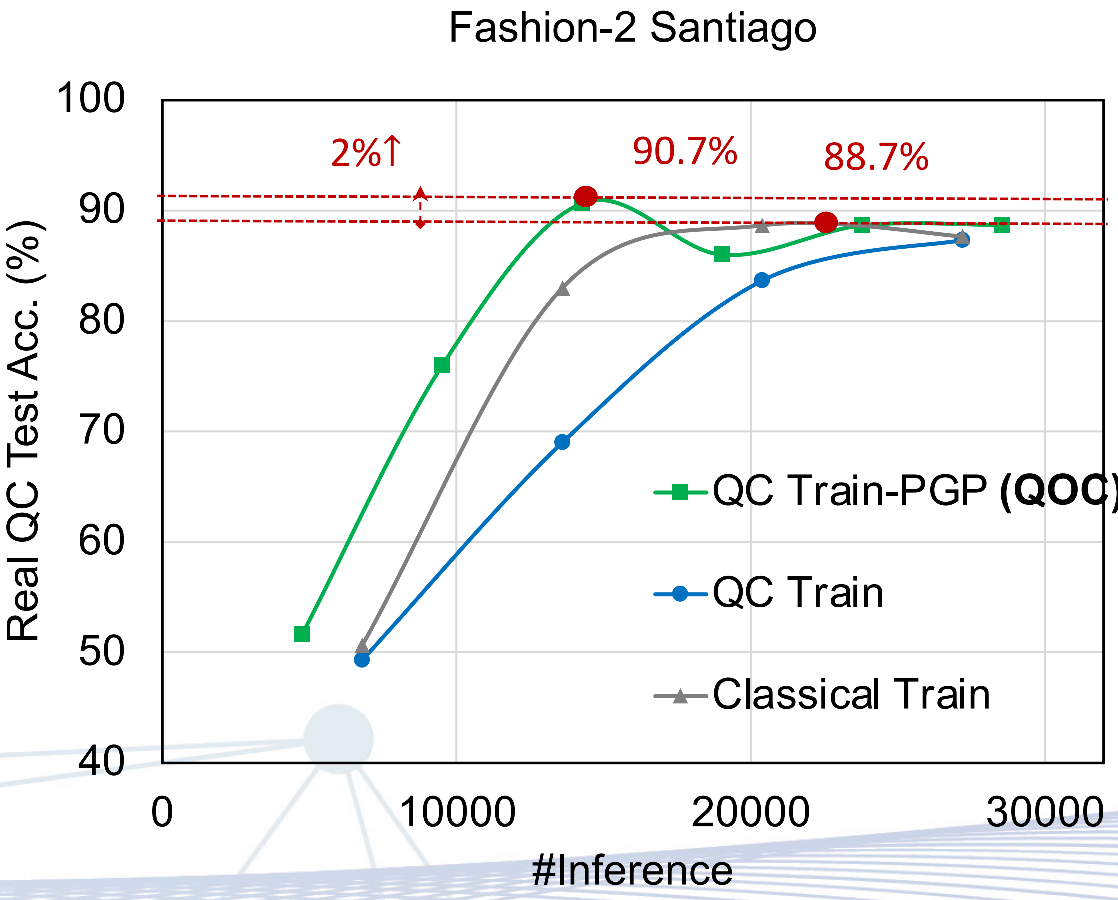
- QC Train-PGP (QOC):
  - Train and test on real QC with gradient pruning

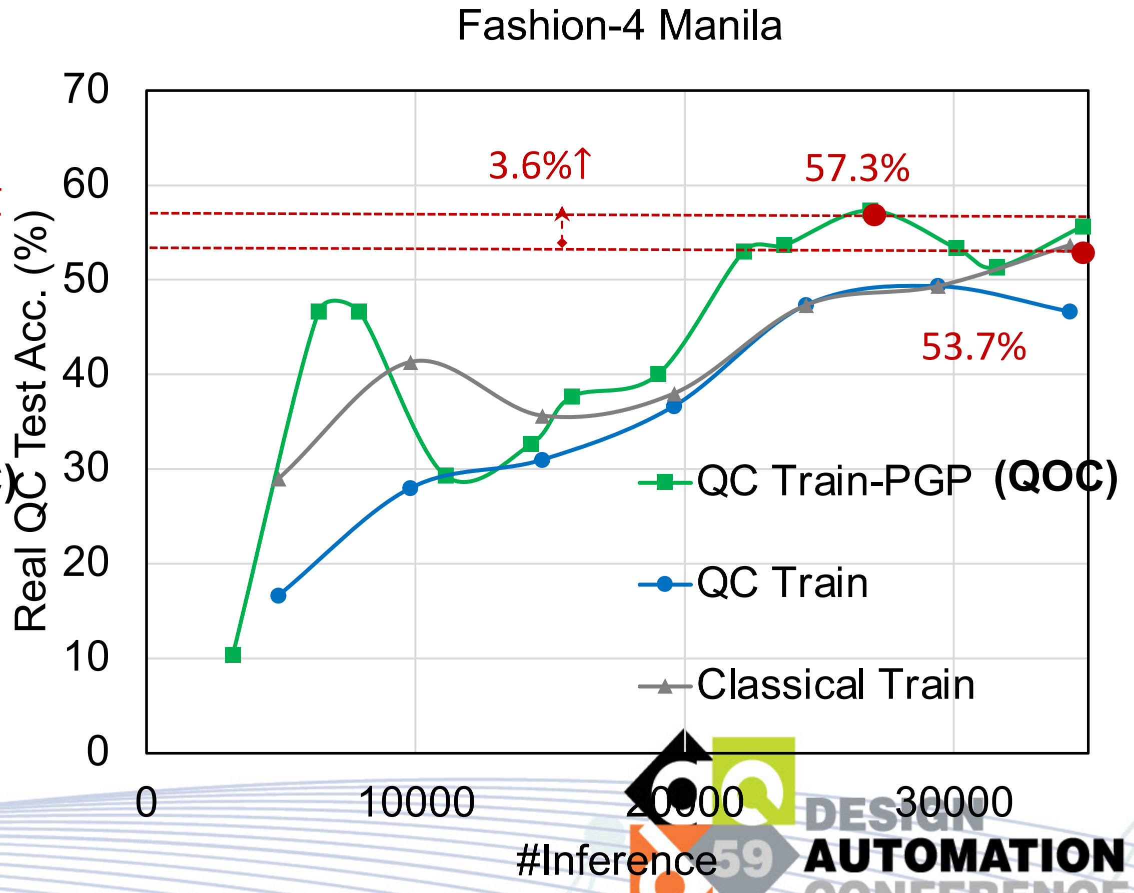




QNN Training Curves

- Gradient pruning can brings **2%~4%** accuracy improvements
- Pruning **accelerate convergence**

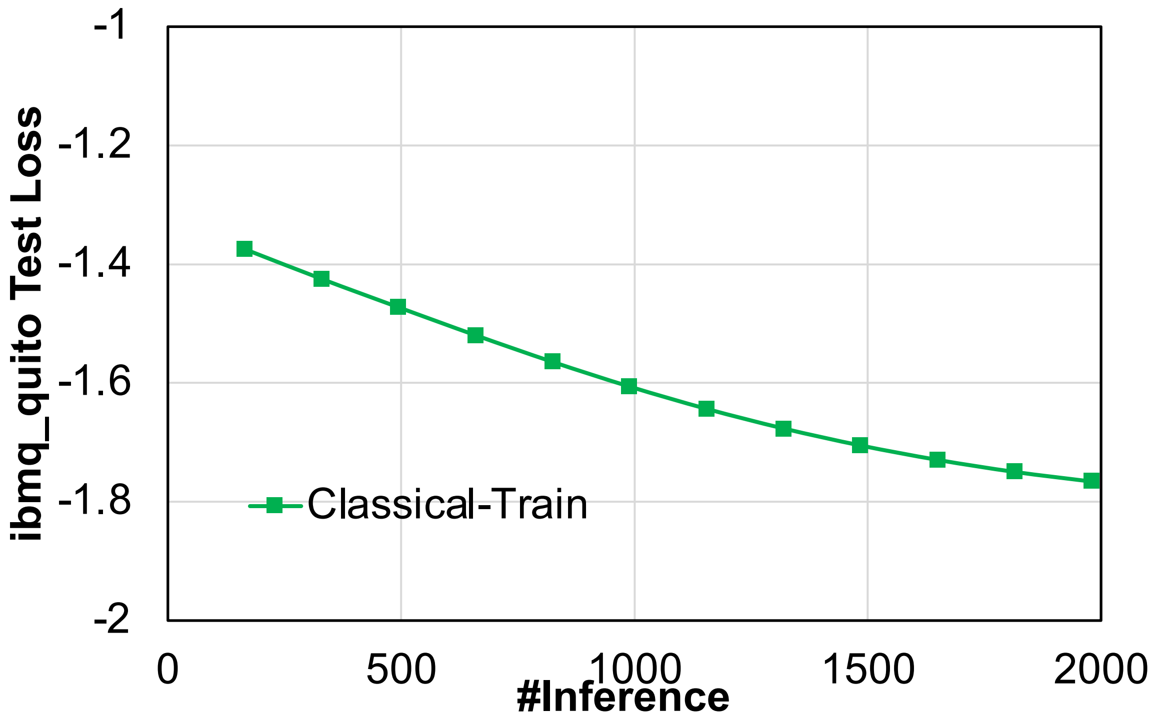




DESIGN AUTOMATION CONFERENCE

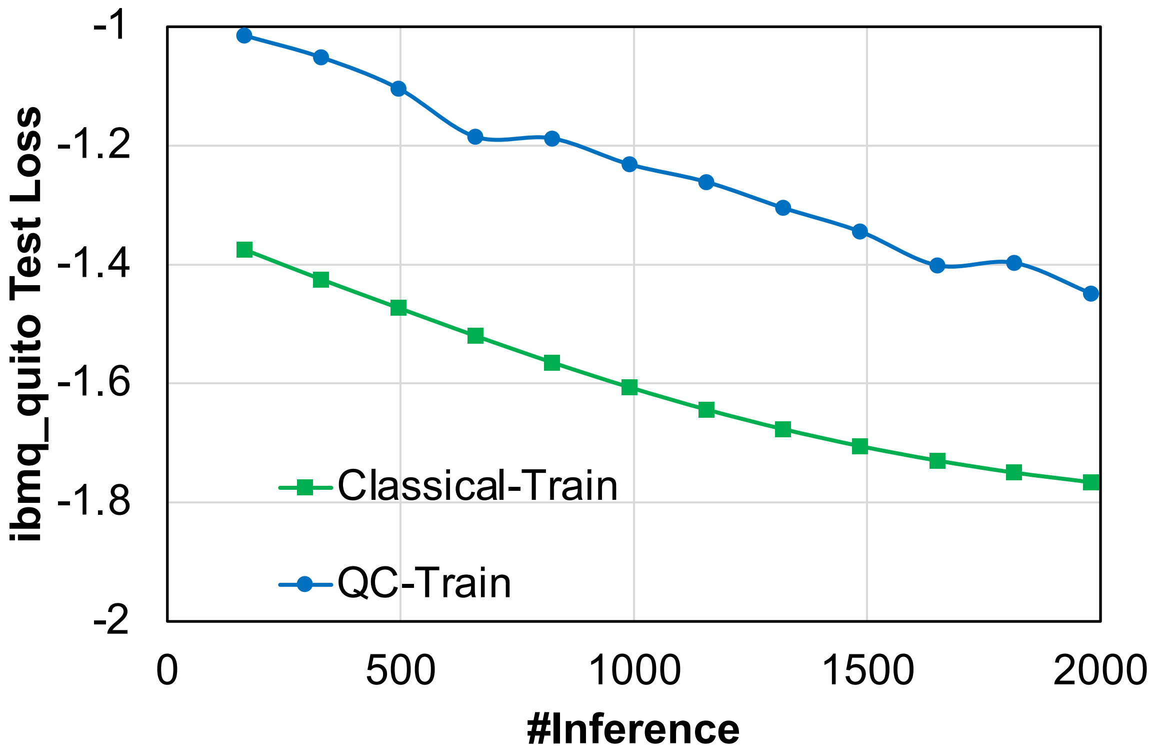
VQE H2 Training Curves

- Estimated ground state energy (y-axis) is the lower the better



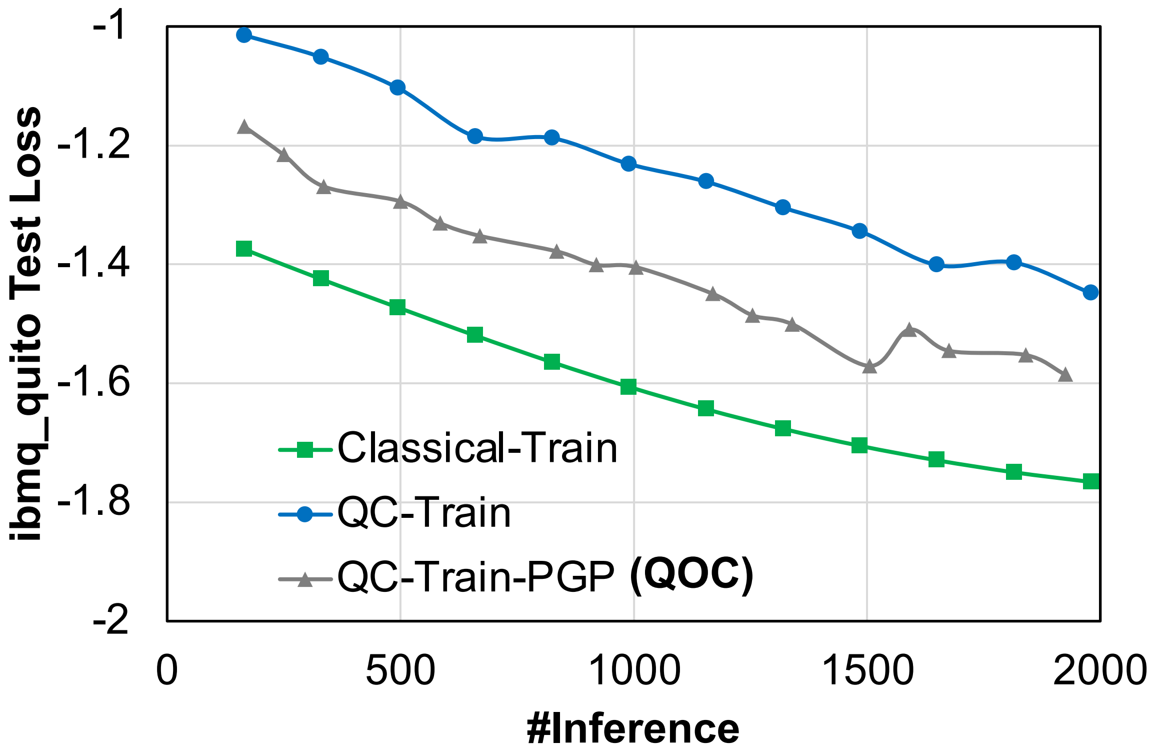
VQE H2 Training Curves

- The loss on real QC is **higher** than that on classical computer



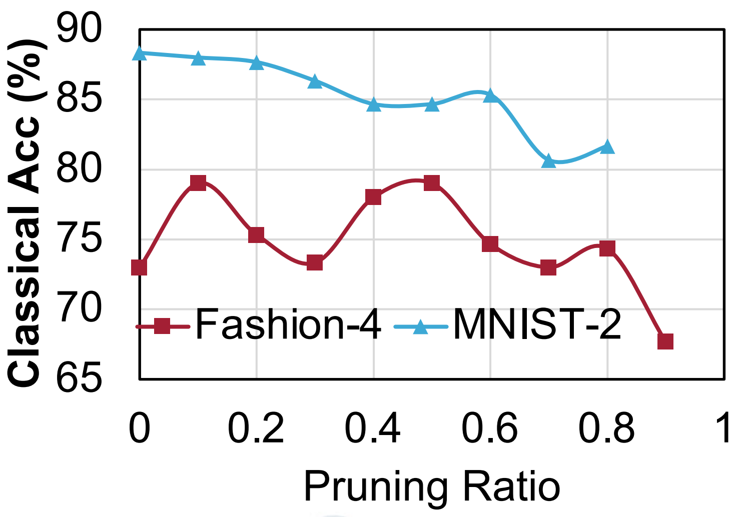
VQE H2 Training Curves

- Gradient pruning can **reduce the gap** between quantum and classical



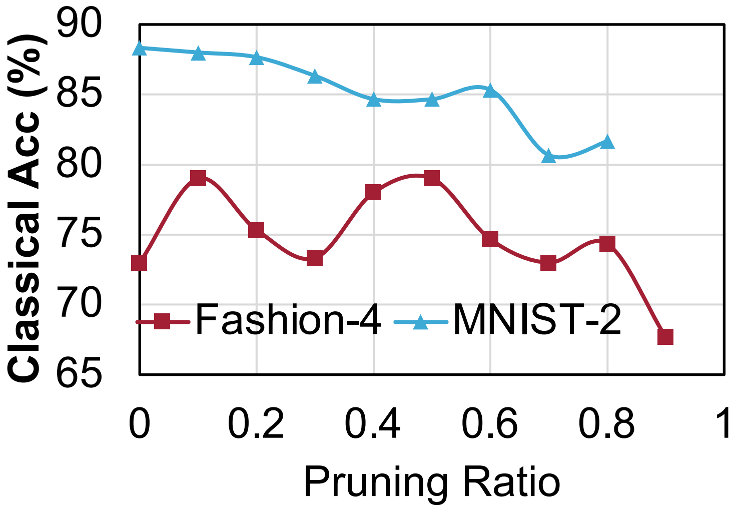
Hyperparameters

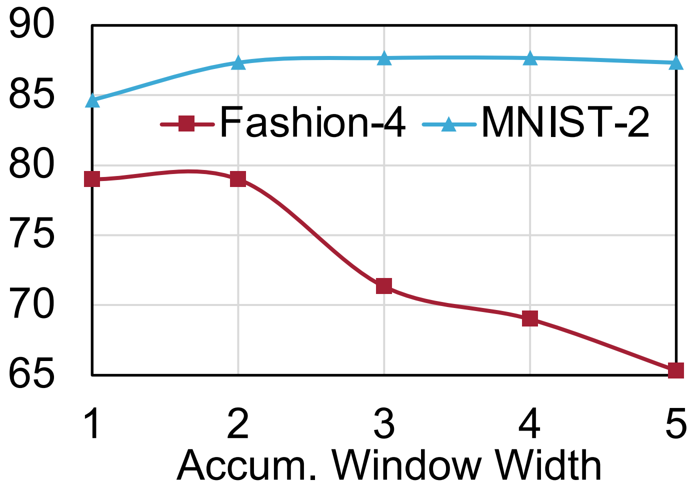
- Hyperparameter setting:
  - Pruning ratio 0.7



Hyperparameters

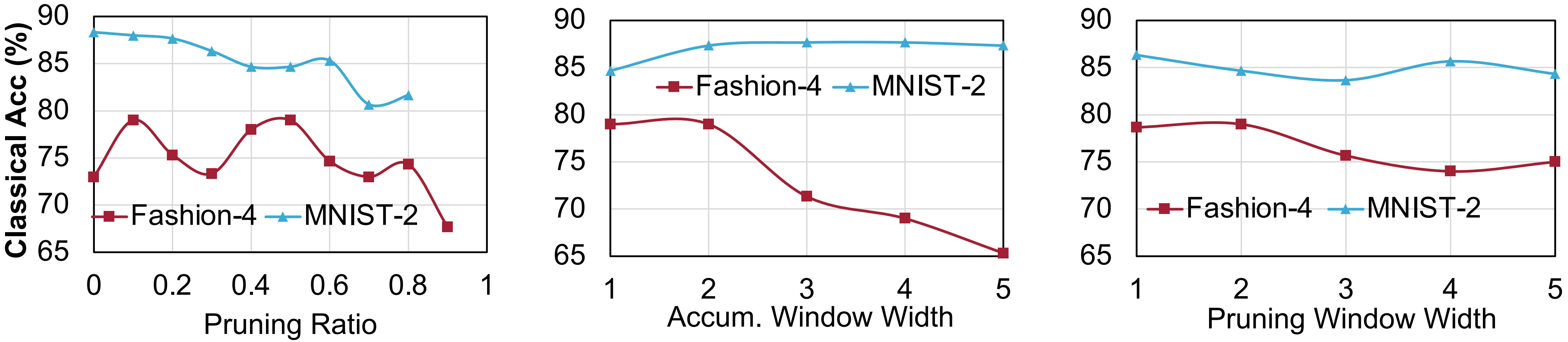
- Hyperparameter setting:
  - Pruning ratio 0.7
  - Accumulation window width: 1





Hyperparameters

- Hyperparameter setting:
  - Pruning ratio 0.7
  - Accumulation window width: 1
  - Pruning window width: 2

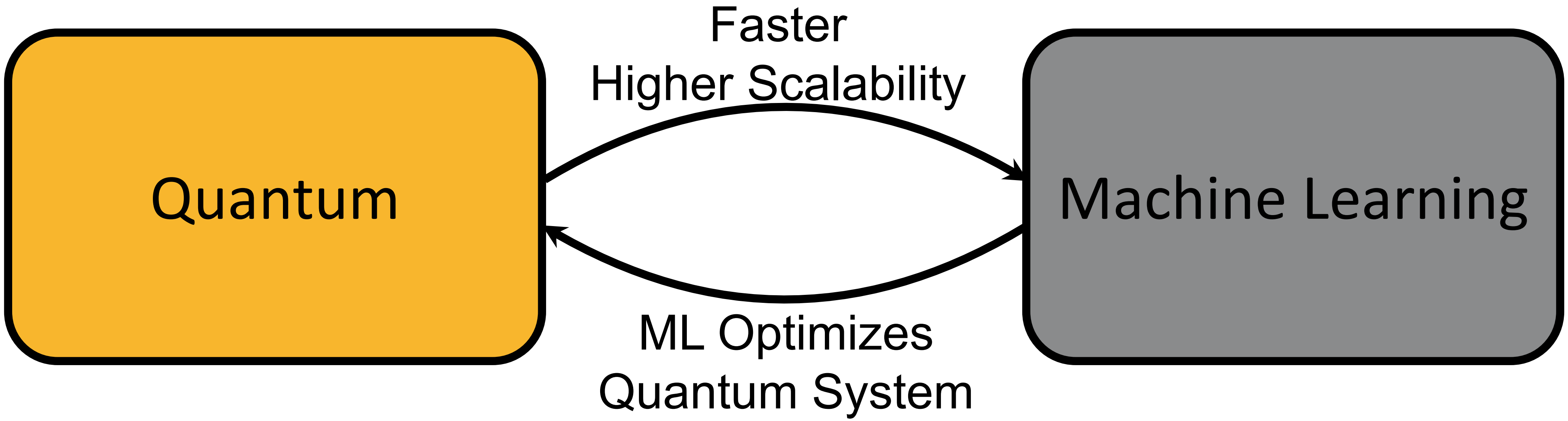


Outline

- Overview
- Background
- QOC
- Evaluation
- TorchQuantum Library
- Conclusion

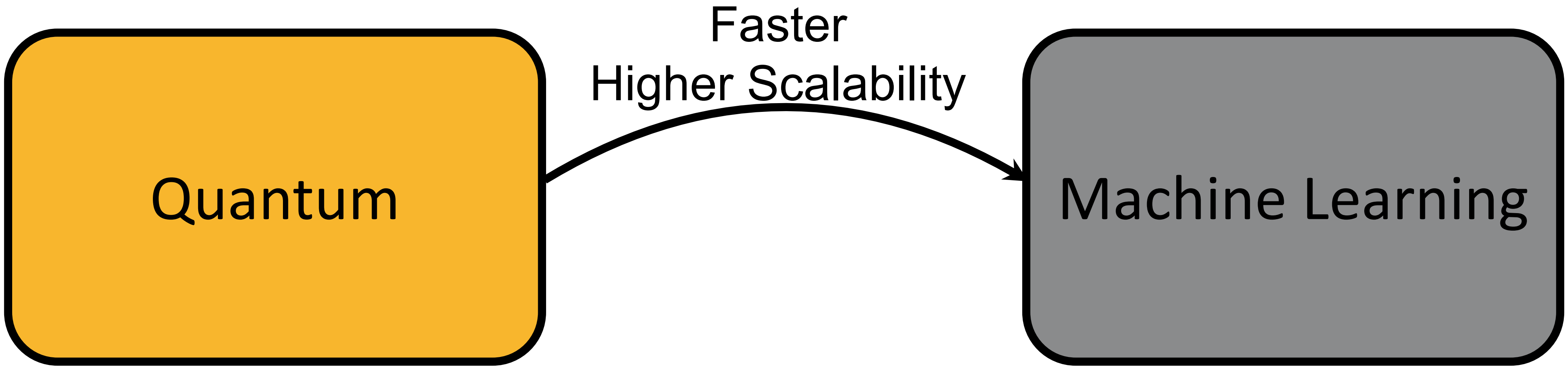
Open-source: TorchQuantum

- TorchQuantum — An open-source library for interdisciplinary research of quantum computing and machine learning
- https://github.com/mit-han-lab/torchquantum



Open-source: TorchQuantum

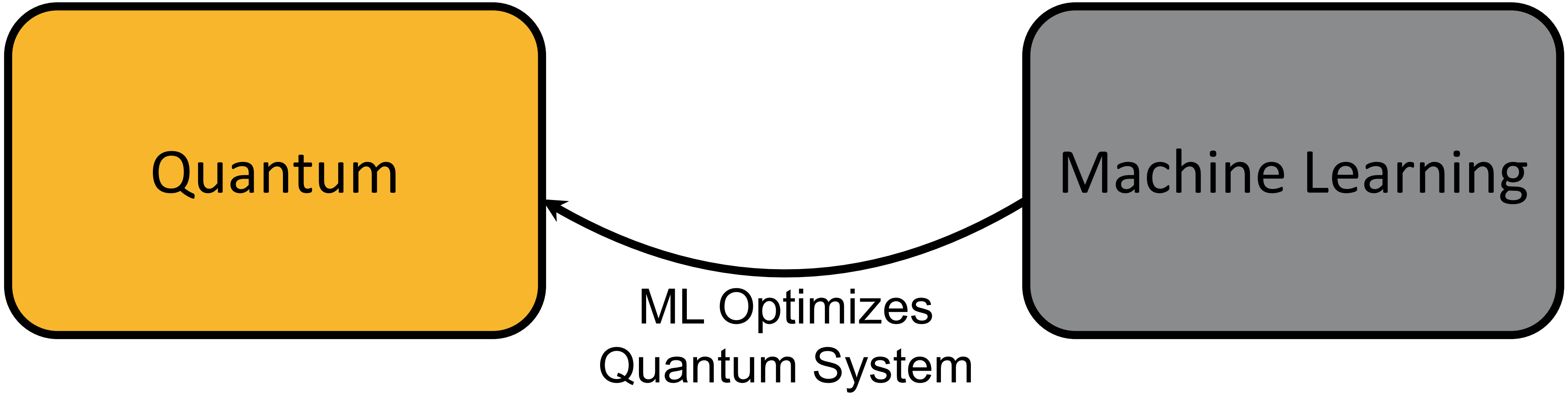
- TorchQuantum — An open-source library for interdisciplinary research of quantum computing and machine learning
- https://github.com/mit-han-lab/torchquantum



- Quantum ML
  - Quantum neural networks
  - Quantum kernel methods

Open-source: TorchQuantum

- TorchQuantum — An open-source library for interdisciplinary research of quantum computing and machine learning
- https://github.com/mit-han-lab/torchquantum



- ML for Quantum
  - ML optimizes quantum compilation

TorchQuantum Features

- Features
  - Easy construction of **parameterized quantum circuits** such as Quantum Neural Networks in PyTorch
  - Support **batch mode inference and training** on GPU/CPU, supports highly-parallelized training
  - Support **easy deployment** on real quantum devices such as IBMQ
  - Provide tutorials, videos and example projects of QML and using ML to optimize quantum computer system problems

TorchQuantum Examples & Tutorials

Torch
Quantum

TorchQuantum Tutorials
Opening

Hanrui Wang
MIT HAN Lab

MIT HAN LAB

Torch
Quantum

TorchQuantum Tutorials
Quanvolutional Neural Network

Zirui Li, Hanrui Wang
MIT HAN Lab

MIT HAN LAB

DESIGN AUTOMATION CONFERENCE 59

Outline

- Overview
- Background
- QOC
- TorchQuantum
- Evaluation
- Conclusion

Conclusion

- QOC: **first experimental demonstration** of quantum on-chip training
  - Higher **scalability**
  - Over 90% and 60% accuracy for 2-class and 4-class classification tasks
- Gradient pruning reduces QC running time by **2x**
- Open-sourced **TorchQuantum** library for Quantum + ML research

Torch Quantum

qmlsys.mit.edu

https://github.com/mit-han-lab/torchquantum

59 DESIGN AUTOMATION CONFERENCE