IBM **Quantum**

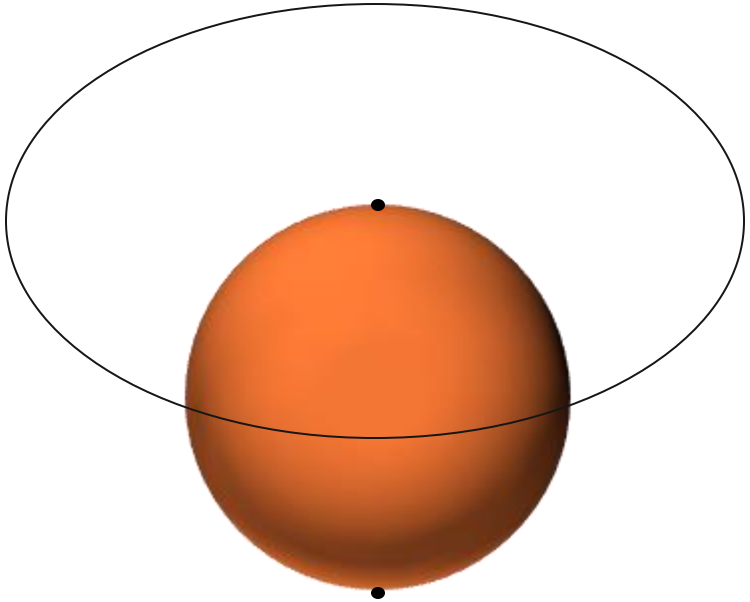
Current State of Quantum Machine Learning (QML)

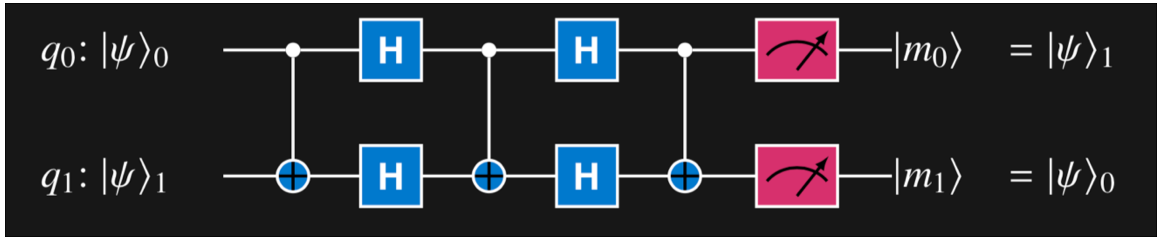
Agenda

- Quantum Computing Overview
- Quantum Machine Learning
- Data Encoding and Mapping
- Near-term methods: QSVM, QNN, QGAN, PQK

Quantum Computing Overview

Quantum bits (qubits) and quantum circuits

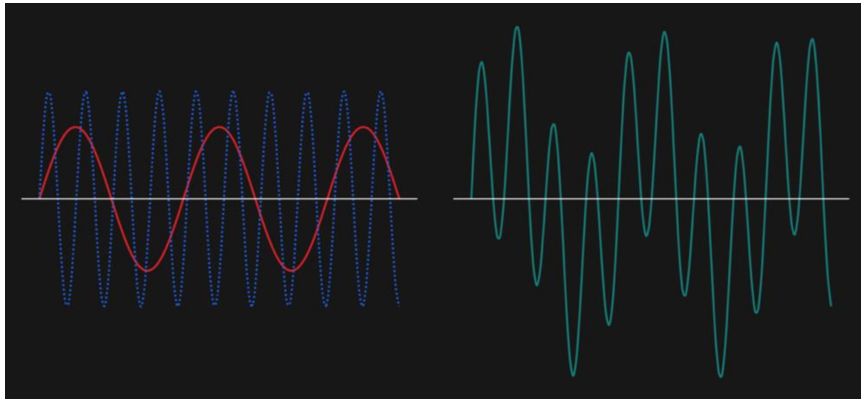




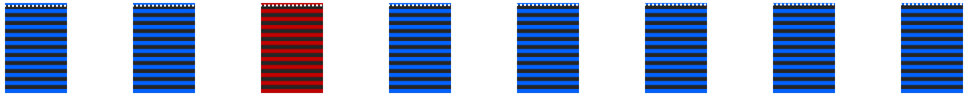
quantum bit qubit

quantum circuit

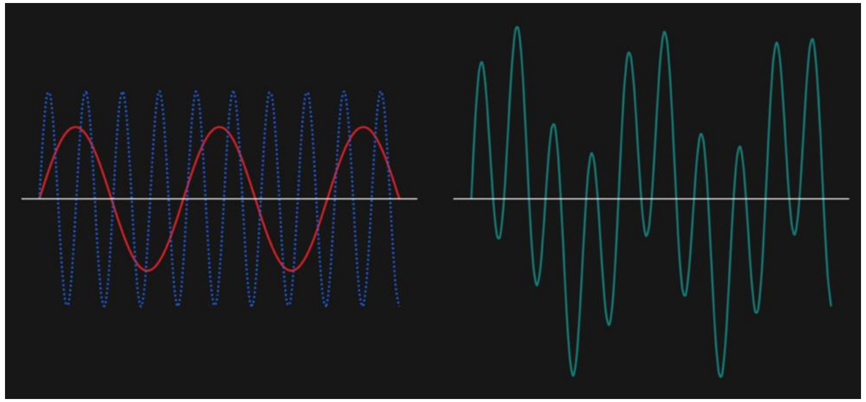
Quantum computing uses essential ideas from quantum mechanics



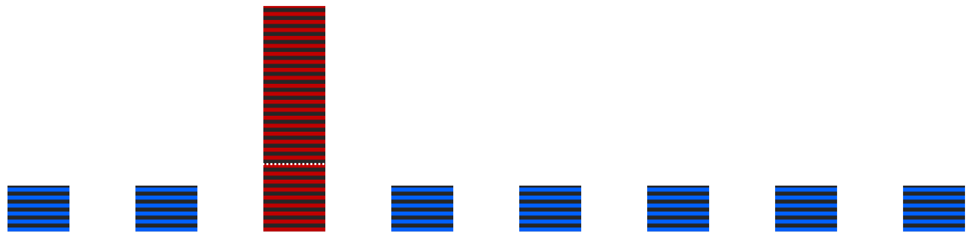
- Interference allows us to increase the probability of getting the right answer and decrease the chance of getting the wrong one.



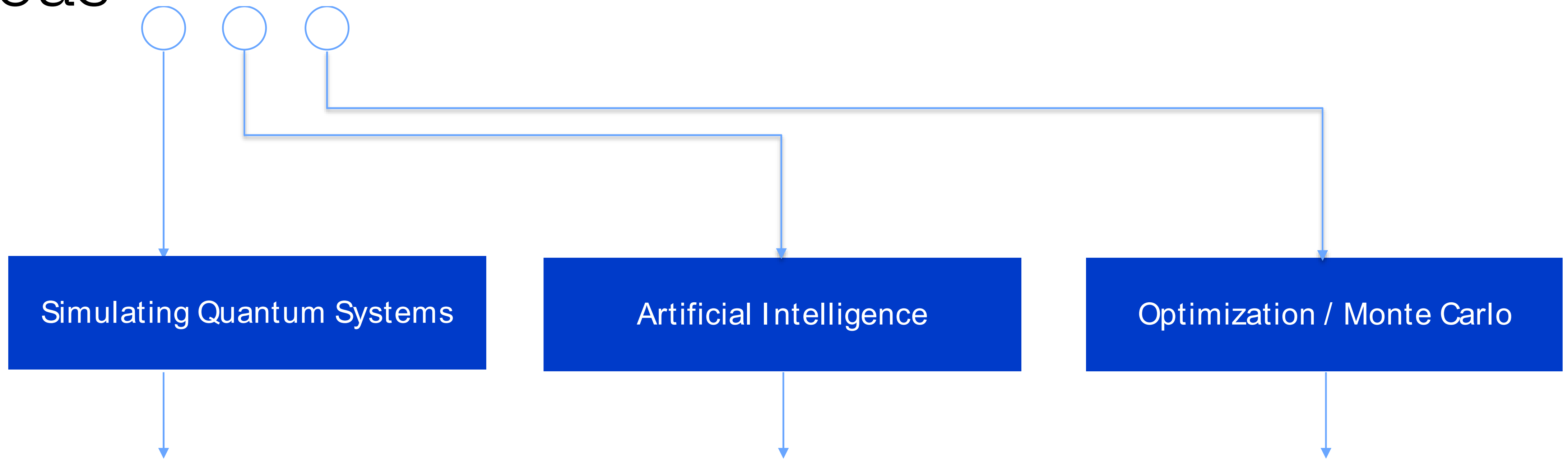
Quantum computing uses essential ideas from quantum mechanics



- Interference allows us to increase the probability of getting the right answer and decrease the chance of getting the wrong one.



Quantum applications span three general areas



Quantum Machine Learning

Machine Learning and Quantum Machine Learning

Type of Algorithm

Processing Device

Type of Data

Data Generating System

	Classical	Quantum
Classical	CC	CQ
Quantum	QC	QQ

Maria Schuld, Francesco Petrruccione
"Supervised Learning with Quantum Computers", Page 6

Considerations beyond "the straightforward"

- *CC: in this context: Machine Learning based on methods borrowed from Quantum Information research, or "Quantum-Inspired"*
- *QC: how can Machine Learning help with Quantum Computing?*
- *CQ: synonym for QML. Data come from classical systems like text, images, time series, macro-economic variables*
- *Requires Quantum-Classical interface*
- *QQ: closely related to CQ. Data can be measured from quantum system or dataset can be Quantum States*

Quantum Machine Learning in a Nutshell

Classical

- Dataset D
New input $\bar{x}$
- Machine Learning Algorithm
- Prediction $\bar{y}$

Quantum

- Dataset D
New input $\bar{x}$
- Encoding
- Quantum Machine Learning Algorithm
- Readout
- Prediction $\bar{y}$

M. Schuld, F. Petruccione: Supervised Learning with Quantum Computers
M. Schuld, N. Killoran: arxiv 1803.07128

Data Encoding

One of the most important parts of QML Algorithms

Frameworks, software and hardware that address the ***interface*** *between classical memory and Quantum Device are key for runtime evaluations*

Encoding strategies

- *Qubit-Efficient* State Preparation
- Encode data in *Superposition*
- *Amplitude-Efficient* State Preparation

Example encoding methods

- Basis encoding
- Amplitude/angle encoding
- Encode dataset via *Hamiltonian*
- Data Encoding as a *Feature Map*

Near-Term and Fault-Tolerant Methods

Emerging Approach - Third Wave

- Based on deeper understanding of ML Potential of Quantum phenomena
- Combine the CML knowledge with Quantum Information Theory
- Train Quantum Models we cannot simulate any more …
- QUANTUM INSPIRED METHODS

> 2021

Near Term Approach - Second Wave

- “What type of ML Model fits the physical characteristics of a small-scale Quantum Device?”
- New Models and Algorithms derived
- “Empirical” and “Heuristic” mindset
- ISSUE: IS A CLASSICAL ML MINDSET SUFFICIENT… ?

> 2017

Long Term Benefit – First Wave

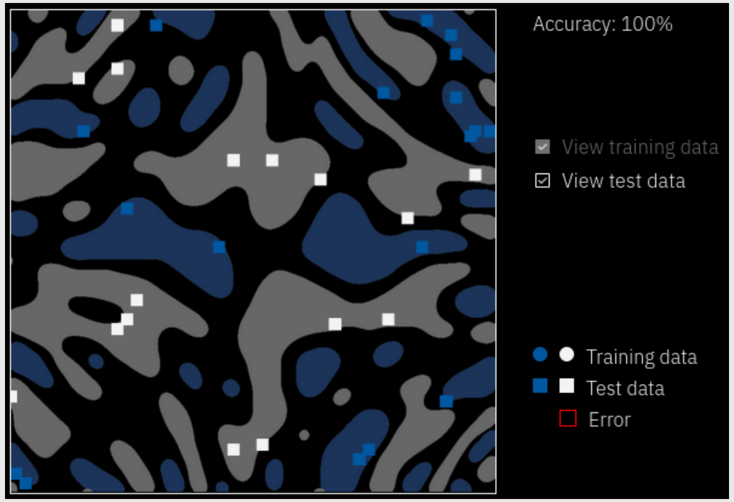
- Assumes Fault-Tolerant Quantum Computers
- Rather “Academic” and “Mathematical” mindset
- ISSUE: WE ONLY HAVE NISQ TODAY …

> 2013

Near-term QML algorithms

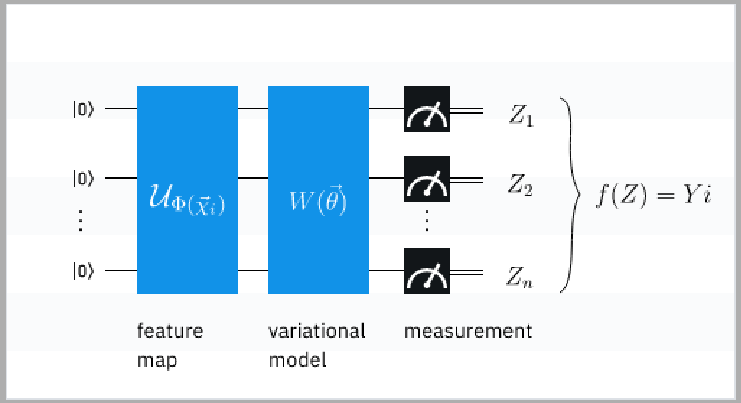
- Quantum SVM
- Classification/regression tasks
- Quantum kernel estimation method
- Benefit: quantum feature space

- Key paper: Havlíček et al, *Nature* 567, pp 209–212 (2019)



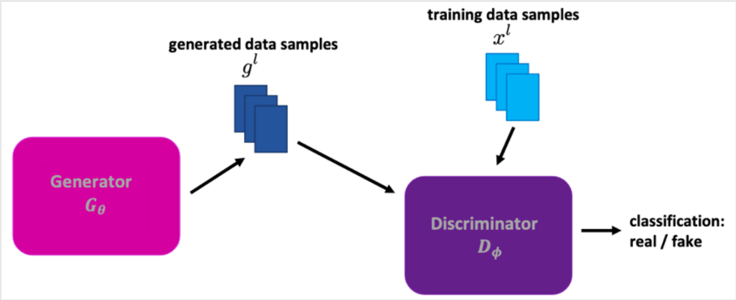
- Quantum Neural Networks
- Classification/regression tasks
- Variational quantum circuits
- Benefits: model expressibility, resilience to barren plateaus

- Key paper: Abbas et al, *Nature Comp. Sci.* 1, pp 403–409 (2021)

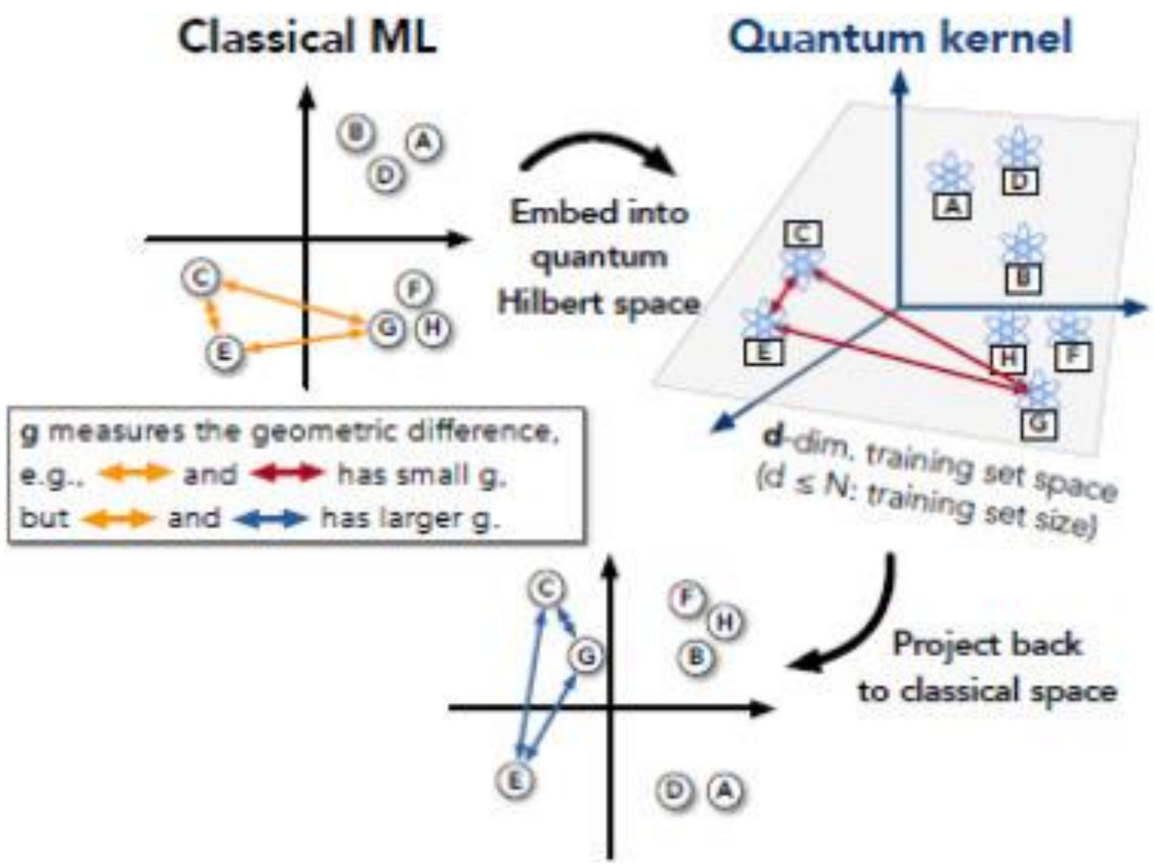


- Quantum GANs
- Data generation tasks
- Quantum/classical neural networks
- Benefits: efficient data sampling

- Key paper: Zoufal et al, *npj Quant. Info.* 5, no: 103 (2019)



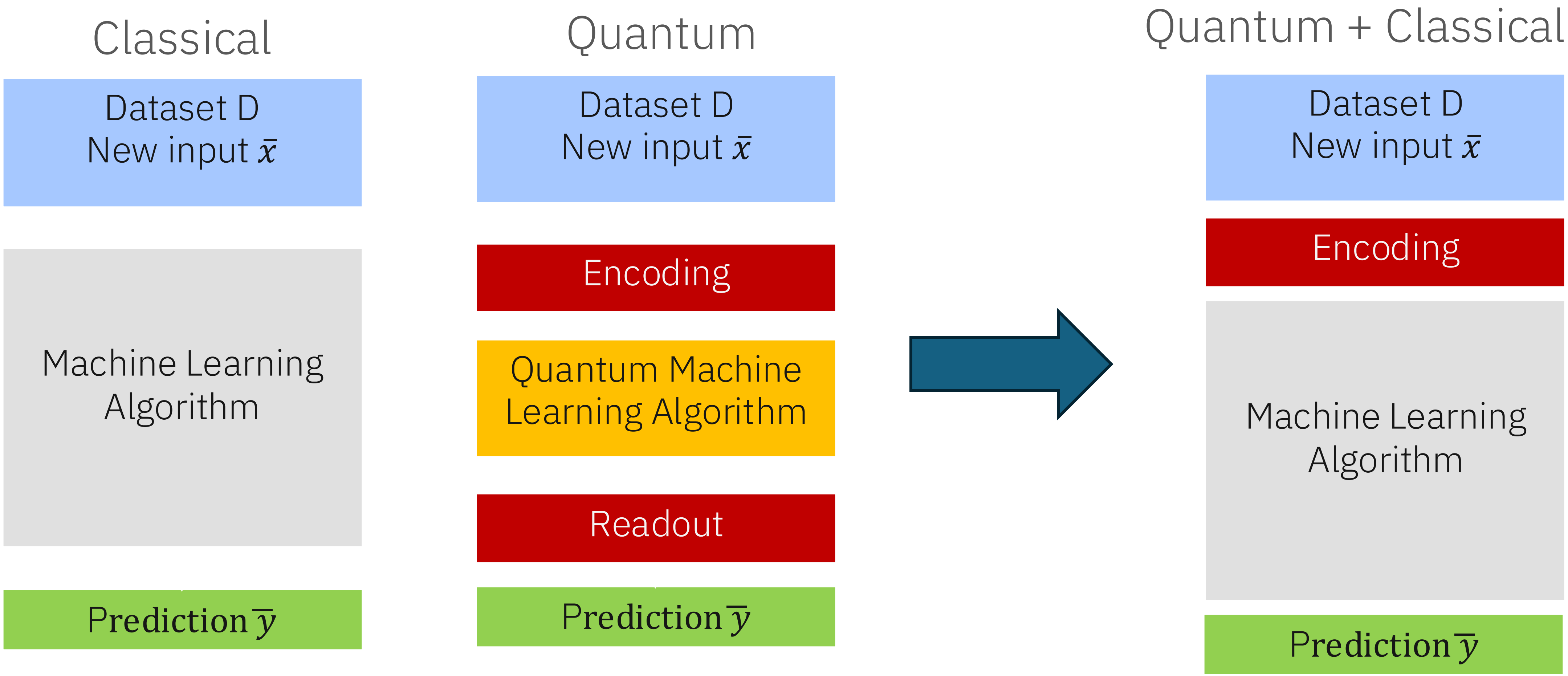
Projected Quantum Kernel



Power of data in quantum machine learning

Hsin-Yuan Huang, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven & Jarrod R. McClean

Nature Communications **12**, Article number: 2631 (2021) | Cite this article


Classical
Dataset D
New input $\bar{x}$
Machine Learning Algorithm
Prediction $\overline{y}$
Quantum
Dataset D
New input $\bar{x}$
Encoding
Quantum Machine Learning Algorithm
Readout
Prediction $\overline{y}$
Quantum + Classical
Dataset D
New input $\bar{x}$
Encoding
Machine Learning Algorithm
Prediction $\overline{y}$

Fault-Tolerant QML

2013

*Are **Quantum Computers** "better" at Machine Learning than Classical Computers?*

- "Traditional Approach" to QML, assuming abundance of perfect Qubits ☺
- NOT novel methods, but novel implementations of the SAME METHODS
- Quality of Algorithm judged thru asymptotic computational complexity

arXiv:1307.0411 (quant-ph)

[Submitted on 1 Jul 2013 (v1), last revised 4 Nov 2013 (this version, v2)]

Quantum algorithms for supervised and unsupervised machine learning

Seth Lloyd, Masoud Mohseni, Patrick Rebentrost

Quantum 'BLAS'

- Amplitude Encoding
- Runtime: $O(log(N.M))$ with $N = 2^n$

- HHL: Invert Data Matrix:
- $$w = (X^T X)^{-1} X^T y$$
- *Linear Regression*
- *[Quantum Data Fitting - N. Wiebe, D. Braun, S. Lloyd - 2012]*

- SVMs: Invert Kernel Matrix
- *Classification*
- *[Quantum SVM for Big Data Classification - P. Rebentrost, M Mohnseni, S. Lloyd - 2013]*

- Hopfield NN: Invert Adjacency Matrix
- *Associative Memory Recall*
- *[Quantum Hopfield NN - P. Rebentrost, T Bromley, C. Weedbrook, S. Lloyd - 2014]*

Based on Grover's Search

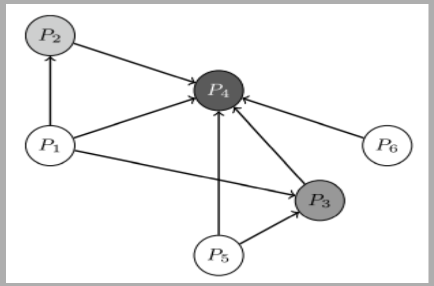
- Amplitude Amplification
- Speedup: $\boldsymbol{O(n^2)}$

- Accelerate Amplitude Amplification
- [Quantum Associative Memory – Ventura, Martinez - 1998]

- Minimize $\mathbf{C}(\boldsymbol{x})$ using Grover
- [A Quantum Algorithm for Finding the Minimum - C. Dürr, P. Hoyer - 1999]

- Perceptron + Amplitude Amplification
- [Quantum Perceptron Models – N. Wiebe, A Kapor, K. Svore - 2016]

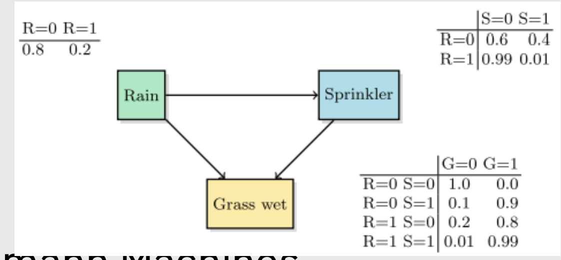
- Quantum Walks (*E.g.: PageRank*)
- [Quantum speed-up of Markov chain- based algorithms]



Probabilistic Methods

- Measurements sampled from classical probability distributions

- Bayesian Net
- Elegant State Preparation
- Runtime is $O(n^2)$better than classical rejection sampling
- [Quantum inference on Bayesian Networks, G H Low, T J Yoder, I L Chuang - 2014]



- Boltzmann Machines
- Qubit Efficient State Preparation
- Mean-field approximation
- [Quantum Deep Learning- N. Wiebe, A. Kapoor, K.Svore- 2015]

Data Encoding and Mapping

Encoding Data

- Basis Encoding
  Encode each n-bit feature into n qubits
  $$x = (b_{n-1}, \ldots, b_1, b_0) \quad \rightarrow \quad |x\rangle = |b_{n-1}, \ldots, b, b_0\rangle$$
  Combine M data points in superposition

- Amplitude Encoding
  Encode into quantum state amplitudes
  $$x = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad \rightarrow \quad |\psi_x\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$$
  Combine M data points in superposition

- Angle Encoding
  Encode values into qubit rotation angles
  $$|x\rangle = \bigotimes^{N} \cos(x_i)\,|0\rangle + \sin(x_i)\,|1\rangle$$

- Arbitrary Encoding (Feature Map)
  Encode N features on N rotation gates in constant-depth circuit with n qubits
  $$x = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad \rightarrow \quad |\psi_x\rangle = \mathcal{U}_{\Phi(x)}|0\rangle$$

Encoding	# Qubits	State prep runtime
Basis	nN	$O(N)$
Amplitude	$\log(N)$	$\frac{O(N)}{O(\log(N))}$
Angle	N	$O(N)$
Arbitrary	n	$O(N)$

N features each

Quantum feature map - Simplistic

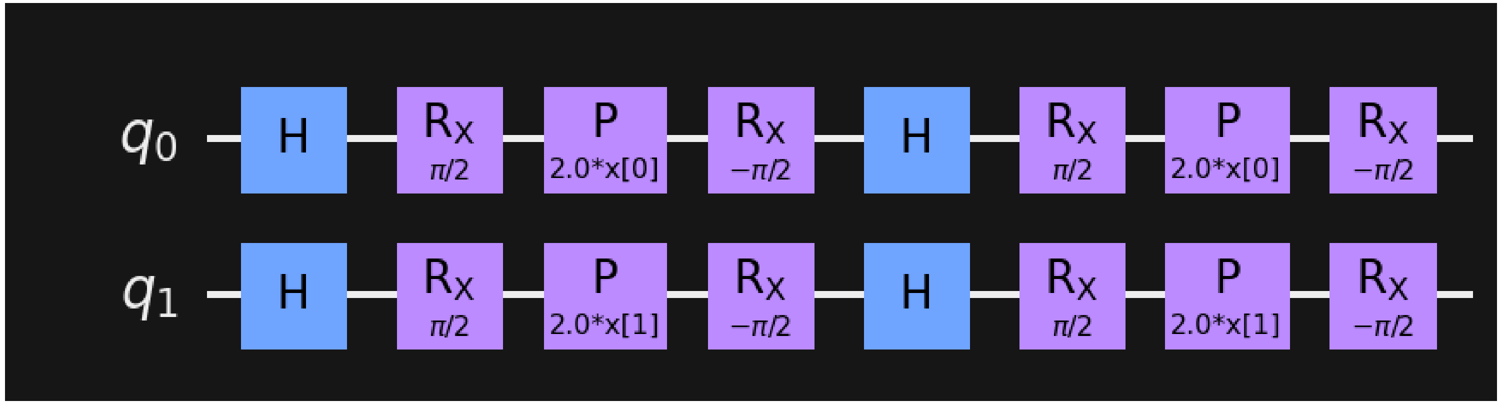
- **Process**
- Apply H gates on all qubits
- Apply parameterized Pauli rotations gates for each feature x_i
- Repeat k times

Encoded state: $|\Phi(x)\rangle = \mathcal{U}_{\Phi(x)}|0\rangle$

$$\mathcal{U}_{\Phi(x)} = \left(U_{\Phi(x)} H^{\otimes n}\right)^d$$

$$U_{\Phi(x)} = \exp\left(i \sum_i \underbrace{\phi_i(x) P_i}_{\text{Pauli rotation operator}} + \cdots\right)$$

```
feature_map = PauliFeatureMap(feature_dimension=2, reps=2,
                              entanglement='linear', alpha=2.0,
                              paulis=['Y'], data_map_func=None)
```



- **Pro:** Simple to implement
- **Con:** Does not exploit high dimensionality

Quantum feature map - Complex

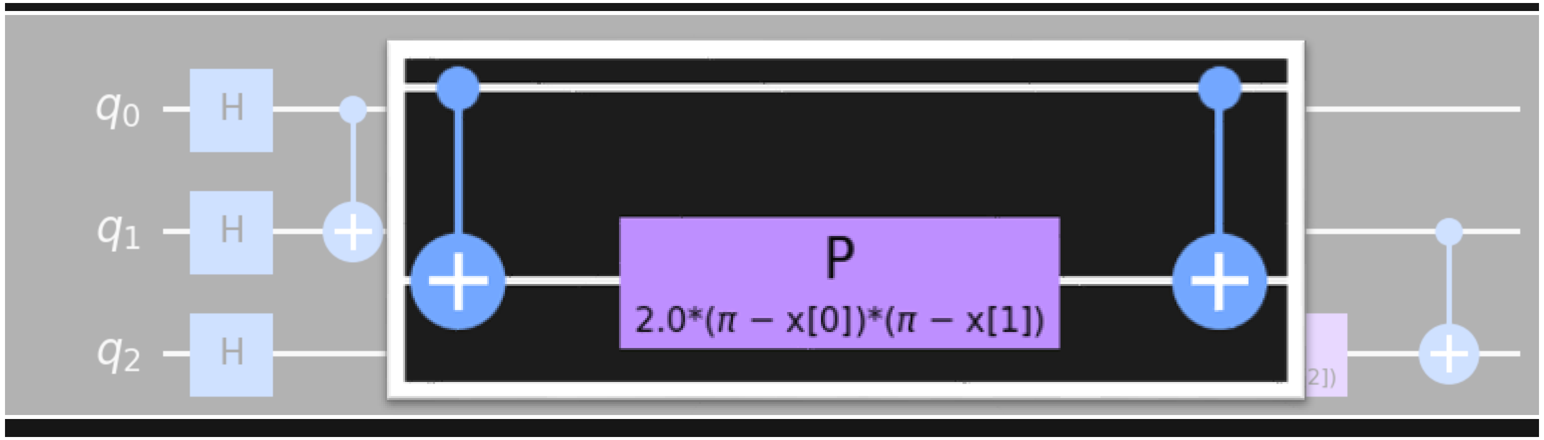
- **Process**
- Apply H gates on all qubits
- Apply entangling gates and Pauli rotations gates for each feature x_i
- Repeat k times

$$|\Phi(x)\rangle = \mathcal{U}_{\Phi(x)}|0\rangle$$

$$\mathcal{U}_{\Phi(x)} = (U_{\Phi(x)}H^{\otimes n})^d$$

$$U_{\Phi(x)} = \exp\left(i\sum_i \phi_i(x)P_i + i\sum_{i,j} \underbrace{\phi_{i,j}(x)P_{ij}}_{\text{2-qubit Pauli rotation operator}} + \cdots\right)$$

```
feature_map = PauliFeatureMap(feature_dimension=3, reps=1,
                              entanglement='linear', alpha=2.0,
                              paulis=['ZZ'])
```



- **Pro:** Exploits entanglement
- **Con:** Adds more gates

Quantum feature map – Complex (2)

- **Process**
- Apply H gates on all qubits
- Apply entangling gates and Pauli rotations gates for each feature x_i
- Repeat k times

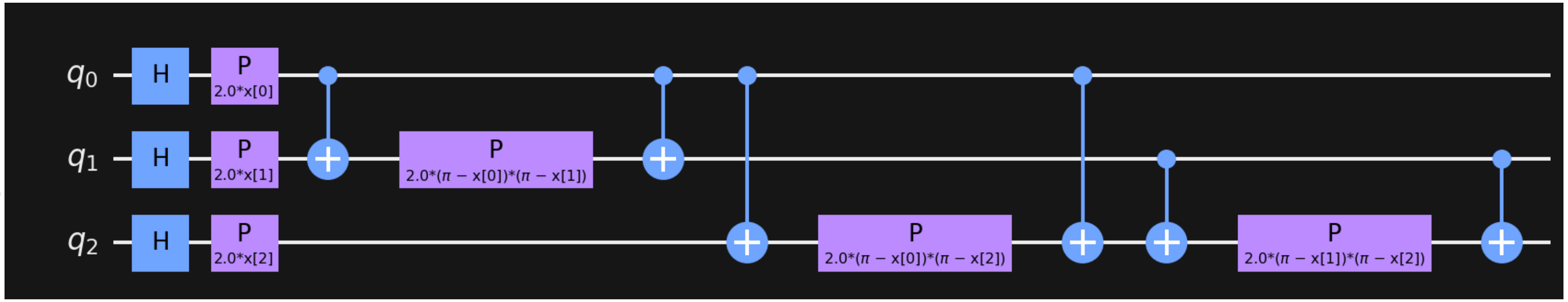
$$|\Phi(x)\rangle = \mathcal{U}_{\Phi(x)}|0\rangle$$

$$\mathcal{U}_{\Phi(x)} = (U_{\Phi(x)}H^{\otimes n})^d$$

$$U_{\Phi(x)} = \exp\left(i\sum_i \phi_i(x)P_i + i\sum_{i,j} \phi_{i,j}(x)P_{ij} + \cdots\right)$$

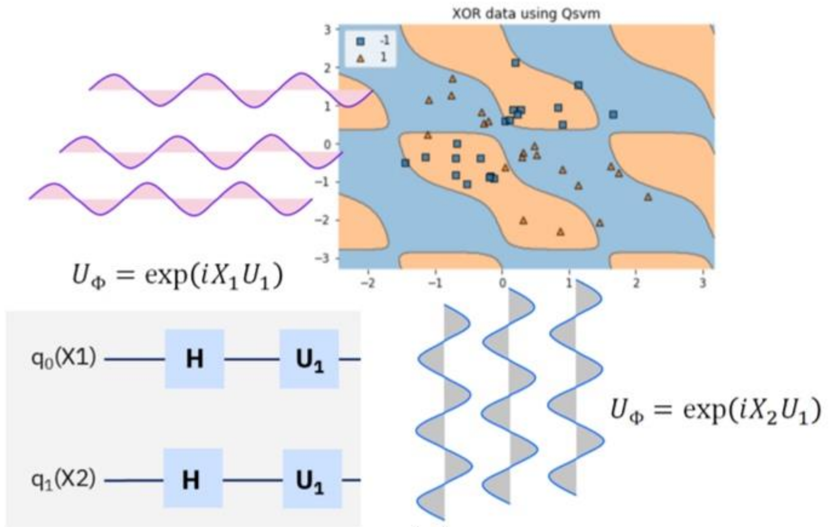
```
feature_map = PauliFeatureMap(feature_dimension=3, reps=1,
                              entanglement='full', alpha=2.0,
                              paulis=['Z','ZZ'])
```

- **Pro:** Exploits entanglement
- **Con:** Adds more gate



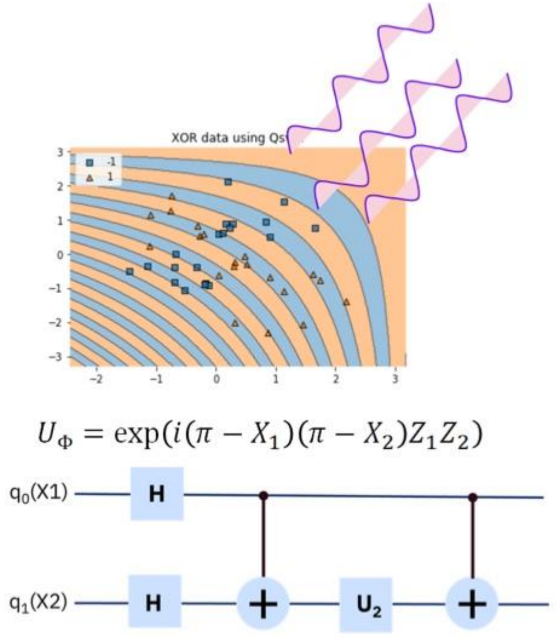
Quantum feature maps

Simple rotations



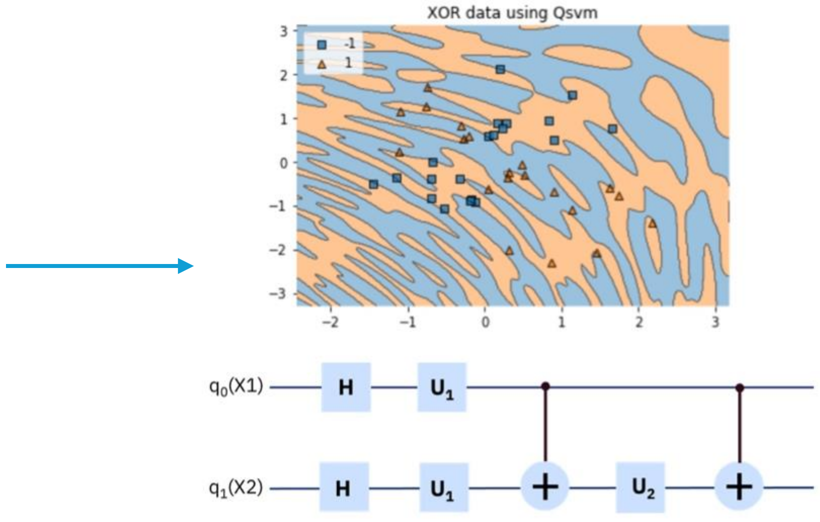
(a)

Entangled unitary rotations



(b)

Simple and Entangled unitary rotations



Park et al, arXiv: 2012.07725v1

Thank you