Quantum Machine Learning and Optimization

Carla Rieger (CERN, TU Munich)

CERN Openlab Summer Student Lectures, 31.7.2023

Recap: the key features of Quantum Computing

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

Quantum Superposition State

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Quantum Entanglement
(here: Bell state)

Recap: the key features of Quantum Computing

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

Quantum Superposition State

Can enable speed-up though **highly parallel** computations

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Quantum Entanglement
(here: Bell state)

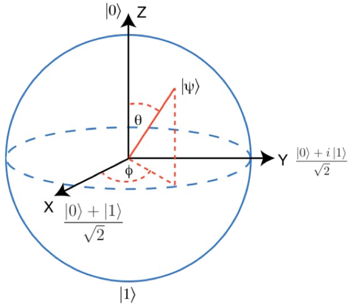
Also, **non-classical correlations** may speed-up computations

Interlude: Bell at CERN

John Stewart Bell commenting on the famous Bell's inequalities at CERN in 1982.

Source: https://physicsworld.com/a/saved-by-bell/

Recap: Basic one qubit gates



Quantum Theory is unitary — gates are represented by unitary matrices U → $U^\dagger U = \mathbb{1}$

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Bit flip

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Phase flip

$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

Bit + phase flip

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Hadamard gate

$$H|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle), \ x \in \{0, 1\}$$

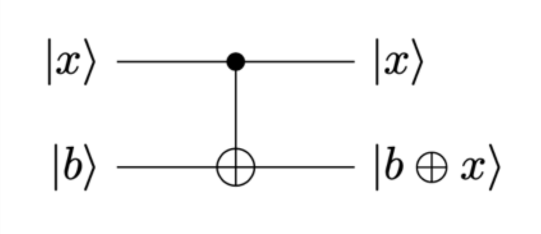
Apply H on computational basis state

- *Pauli matrices (together with identity matrix) form basis of 2x2 matrices*
- *any 1-qubit rotation can be written as a linear combination of Pauli gates*

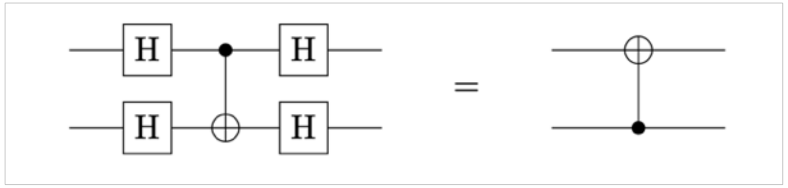
Nielsen, Michael A., and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

Recap: Basic two qubit gate

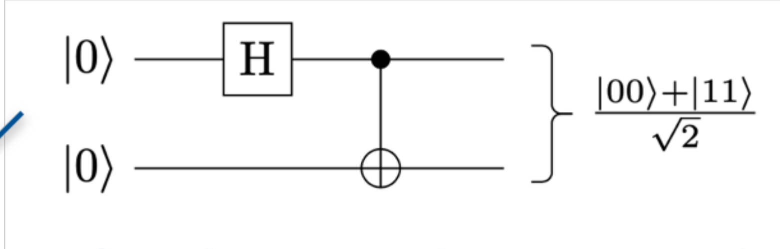
- since Quantum Theory is *unitary*, gates must be *reversible*
- *CNOT gate:* reversible XOR gate



CNOT gate

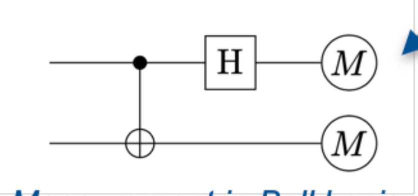


Switched CNOT gate



***Creating entanglement** —> switch computational to Bell basis*

$$
\begin{cases}
|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \\
|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \\
|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \\
|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)
\end{cases}
$$



Measurement in Bell basis

Nielsen, Michael A., and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

Aim of Quantum Computing

→ Do **classically intractable** computations **efficiently** on a Quantum Computer leveraging Quantum Effects

Applications of Quantum Computing

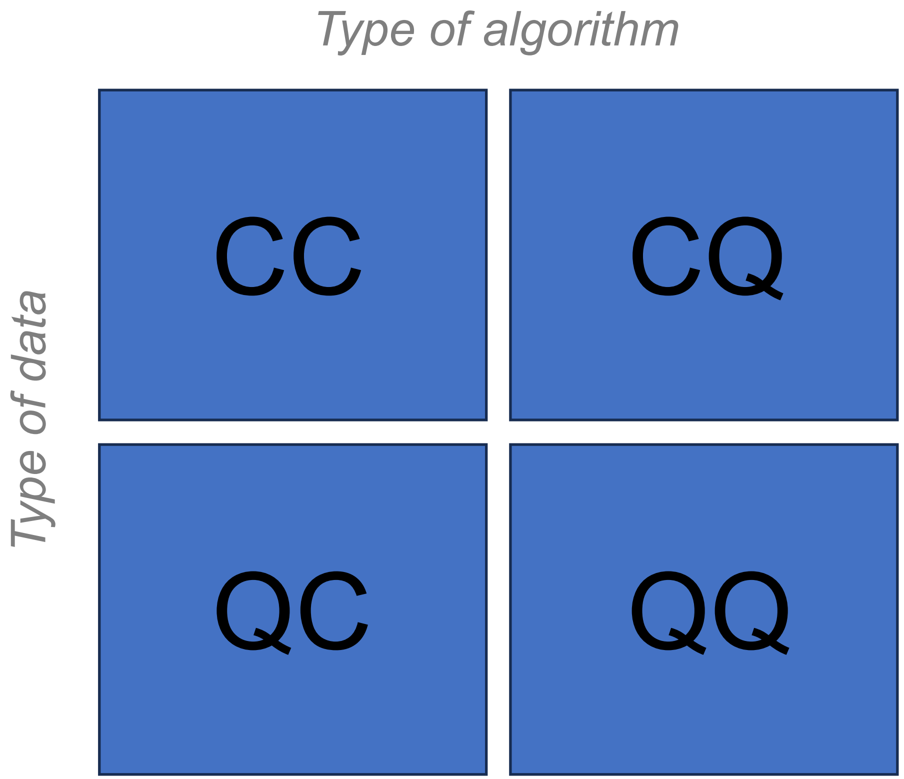
One may successfully leverage quantum effects for:

- Efficient sampling, search and optimization (e.g., Grover's search algorithm)
- Linear algebra, matrix computations and machine learning (e.g., HHL-algorithm)
- Algorithms and protocols for Cryptography and Communication (e.g., Shor's algorithm, Quantum Key distribution)

Based on previous year's talk

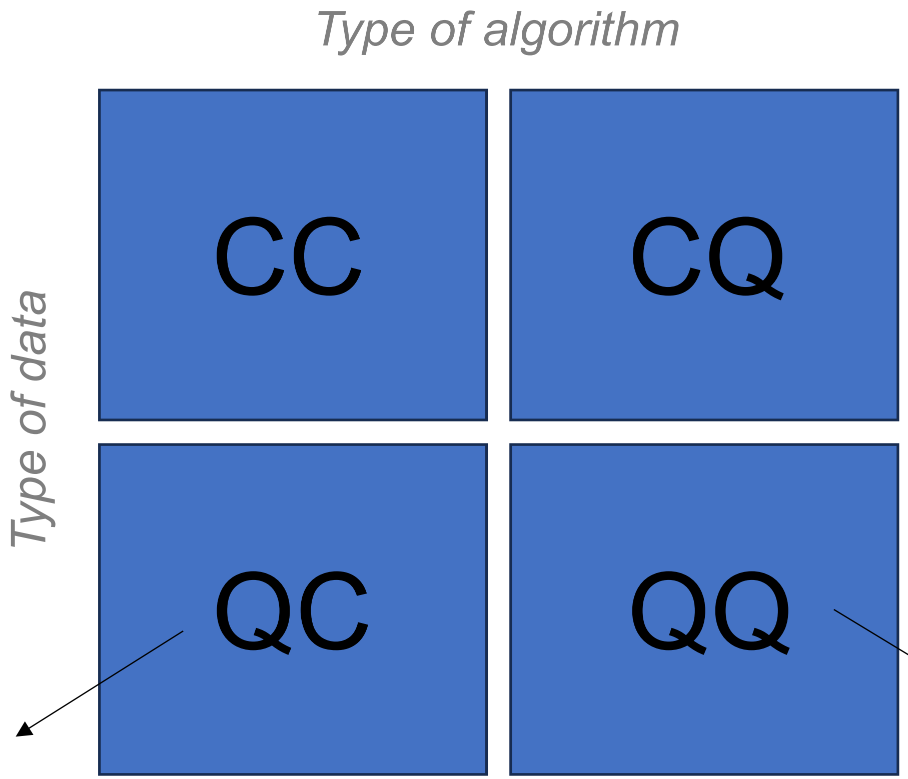
What is Quantum Machine Learning?

Fields in Quantum Computing



Source: Qiskit Textbook

Fields in Quantum Computing

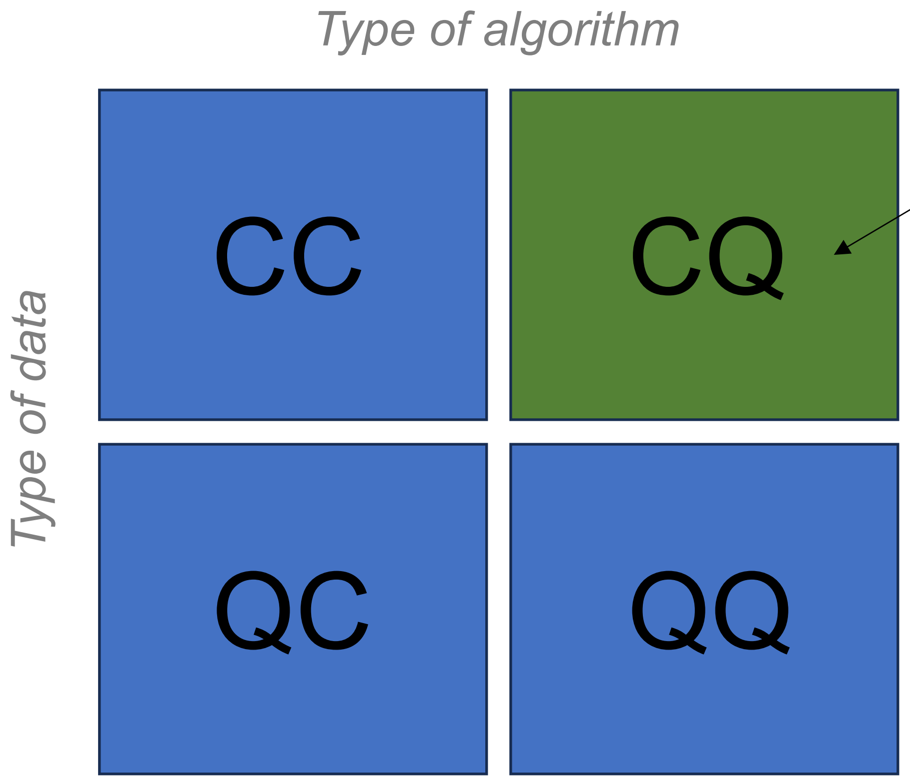


Simulation of Quantum Systems using classical ML

Simulation of Quantum Systems using a Quantum Computer

Source: Qiskit Textbook

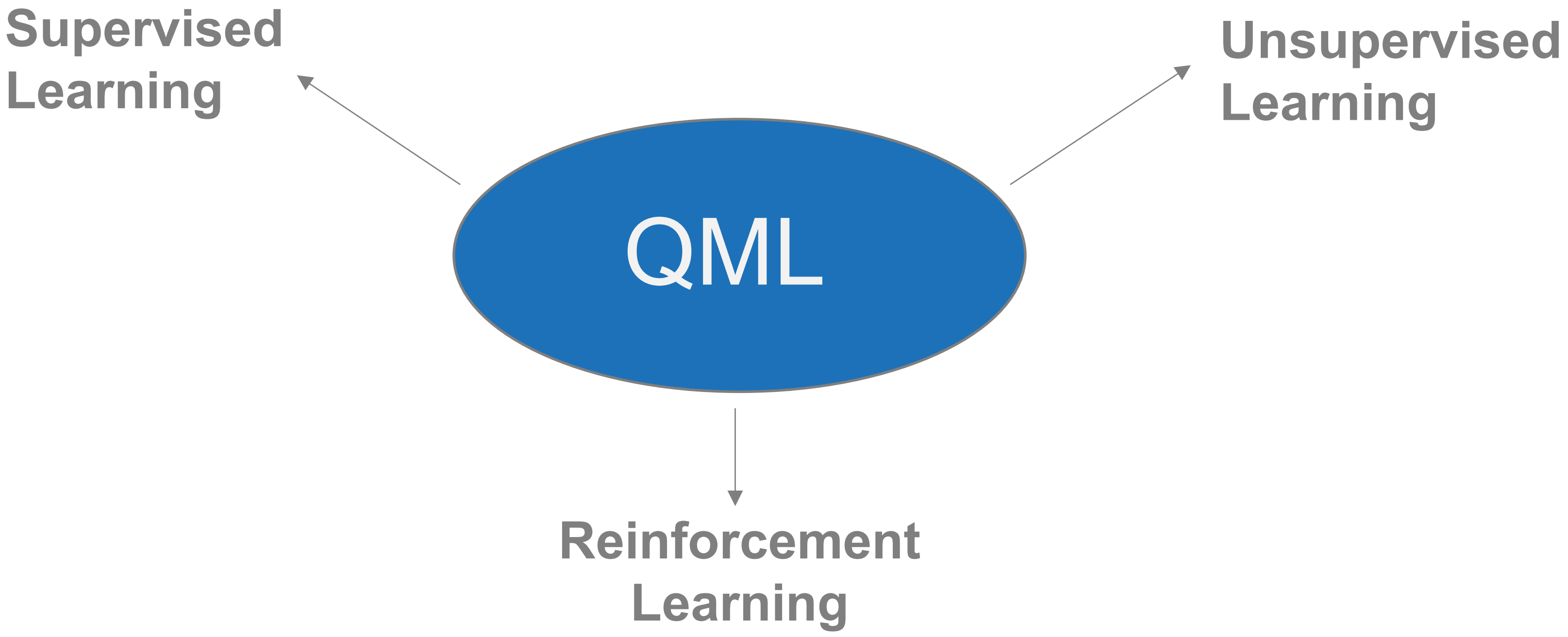
Fields in Quantum Computing



Here: **focus on quantum algorithm with classical input data**

Source: Qiskit Textbook

Fields in Quantum Machine Learning (QML)



Source: Qiskit Textbook

Fields in Quantum Machine Learning (QML)

QML

Supervised Learning

e.g., **Quantum Classifier**: aims to learn *input-output relation* of labeled dataset $f: x_{in} \mapsto x_{out}$ by optimizing quantum network

Unsupervised Learning

e.g., **Quantum GAN**: aims to *learn the underlying probability distribution* $\pi(y)$ of a given data set and *generates samples* from it using quantum network

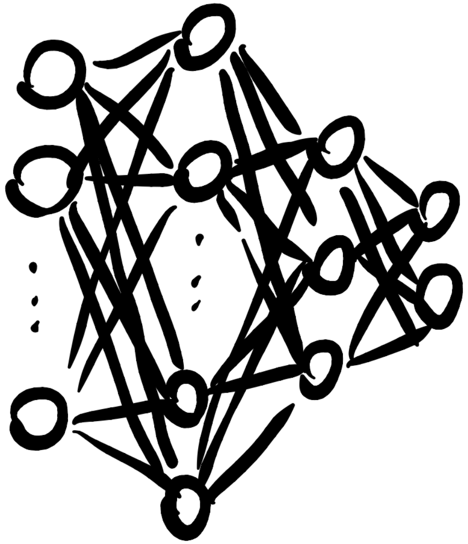
Reinforcement Learning

e.g., **Quantum Reinforcement Learning**: find *policy* for agent that *maximizes reward* (expected reward computed using QC)

Source: Qiskit Textbook

Supervised Learning in Quantum Computing: Quantum Classifiers

→ Goal: learn input-output relation of labeled data



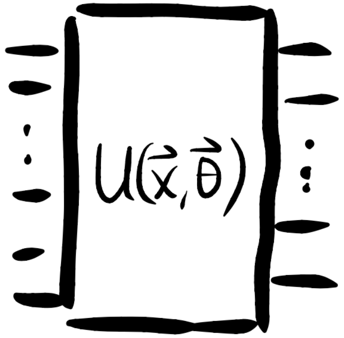
$\psi(\vec{x},\vec{\theta}) : \mathbb{R}^{n_{in}} \to \mathbb{R}^{n_{out}}$

input trainable weights

Classical Neural Network

↔

$|0\rangle^{\otimes n_{in}}$



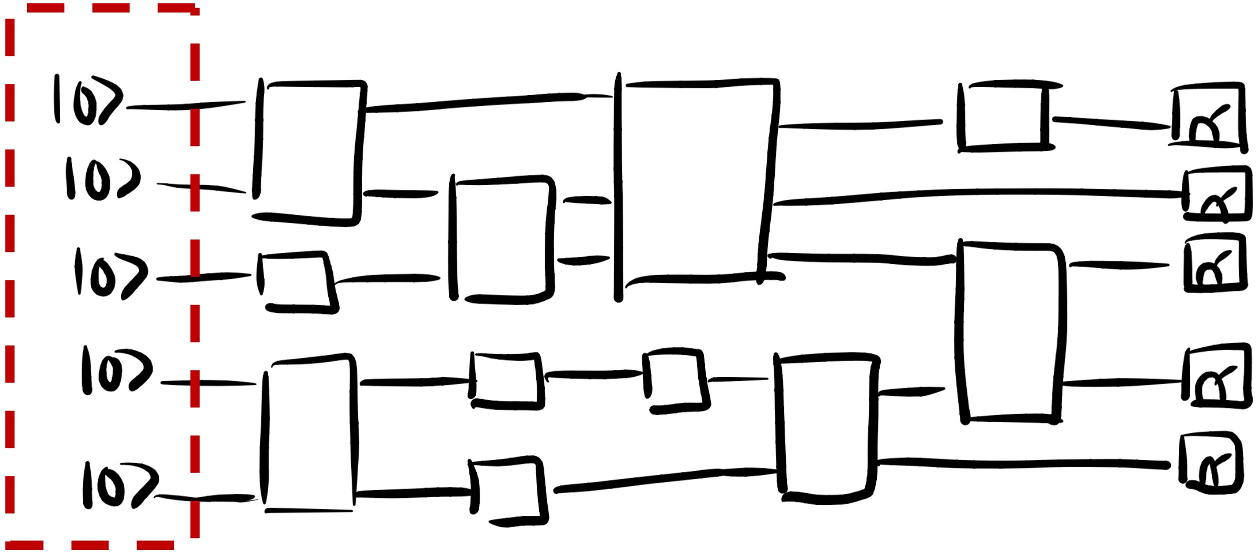
$|\psi_{out}(\vec{x},\vec{\theta})\rangle = U(\vec{x},\vec{\theta})|0\rangle^{\otimes n_{in}}$

input trainable weights

$y(\vec{x},\vec{\theta}) = \langle \psi_{out}(\vec{x},\vec{\theta})|\hat{O}|\psi_{out}(\vec{x},\vec{\theta})\rangle$

Parametrized Quantum Circuit

Quantum Circuits and *the Born rule*

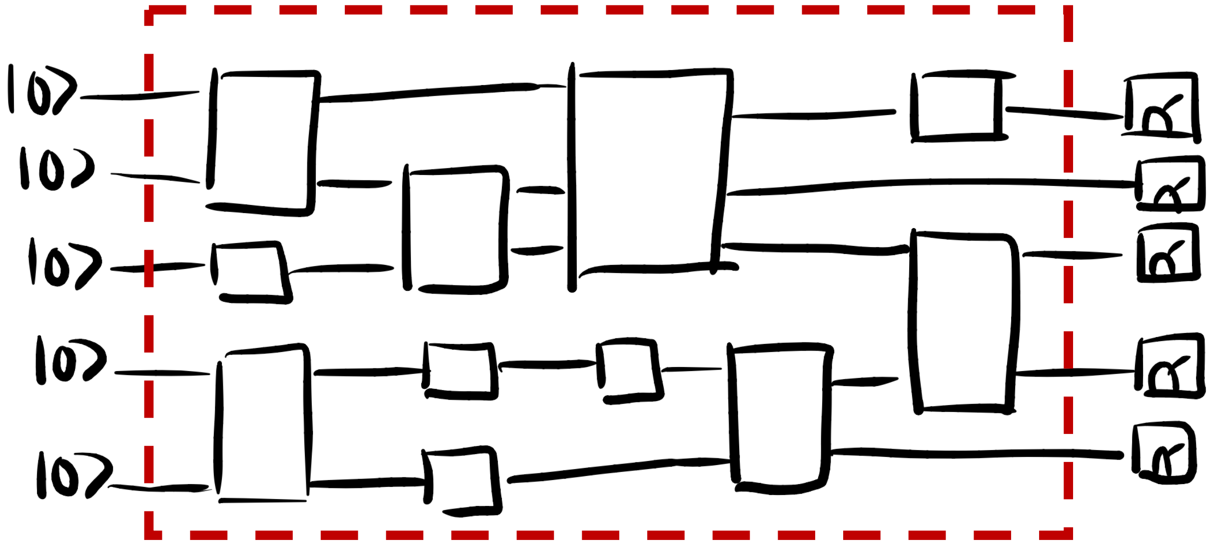


An arbitrary quantum circuit generating the state $|\Psi\rangle$

Initialization:

→ initialize qubits in computational basis state

Quantum Circuits and *the Born rule*

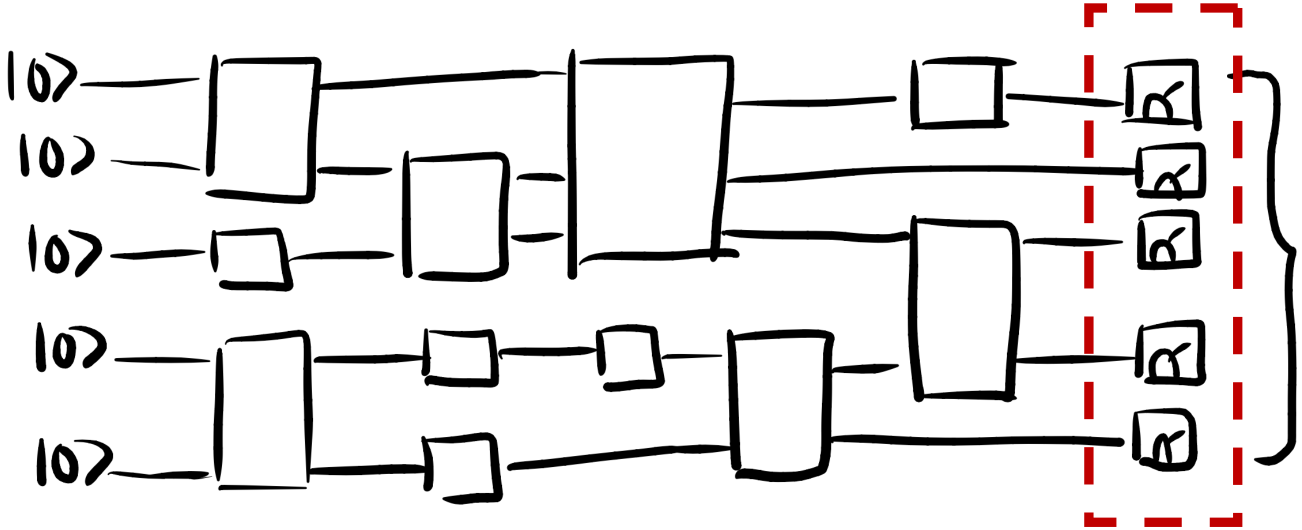


An arbitrary quantum circuit generating the state $|\Psi\rangle$

Evolve initial state:

→ Apply set of **unitary** gates that may **encode classical input data x** and include **parametrized gates**

Quantum Circuits and *the Born rule*



An arbitrary quantum circuit generating the state $|\Psi\rangle$

Quantum Measurement

→ retrieve a classical output distribution $|\langle x|\Psi\rangle|^2$ of classical output states

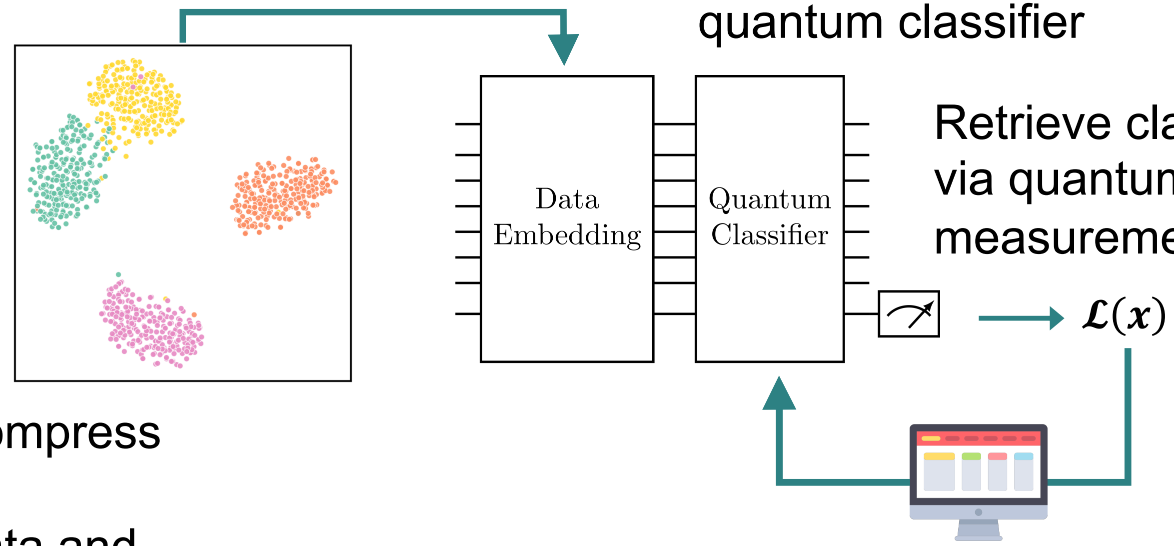
(with $x \in \{0,1\}^n$) according to Born rule

Parametrized Quantum Circuits – *the data processing pipeline*

Encode (classical) data in Quantum state

Apply a variational quantum circuit as e.g., quantum classifier

Retrieve classical output via quantum measurement



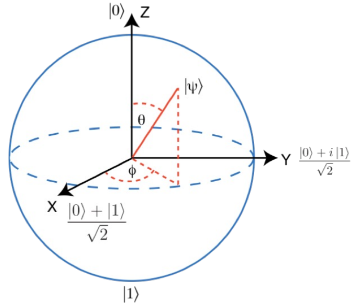
Compute Loss function

(Classically) compress possible high-dimensional data and extract features

Classically optimize variational circuit parameters

$$\theta_{t+1} \longleftarrow \theta_t - \eta \nabla_x \mathcal{L}(x)$$

Embedding classical information in a Quantum Circuit



→ Tradeoff between depth of input encoding quantum circuit and exponential compression of classical input data

Angle encoding:

- Classical input encoded using rotational gates (e.g., $R_x(\theta)$)
- Constant depth wrt. to encoded features
- Number of qubits scales linearly in number of features

$$|\phi(x)\rangle = \bigotimes_{i=1}^{n} R_x(x_i)|0^n\rangle = \bigotimes_{i=1}^{n} (\cos(x_i/2)|0\rangle - i\sin(x_i/2))$$

$n \propto \mathcal{O}(N)$, $n_{gates} \propto n$

Amplitude encoding:

- Classical input encoded as amplitudes of the quantum state
- N-dimensional data point x is encoded by a n-qubit quantum state with $N = 2^n$
- Much deeper circuit depth for encoding, see scaling:

$$|\phi(x)\rangle = \frac{1}{||x||}\sum_{i=1}^{N} x_i|i\rangle$$

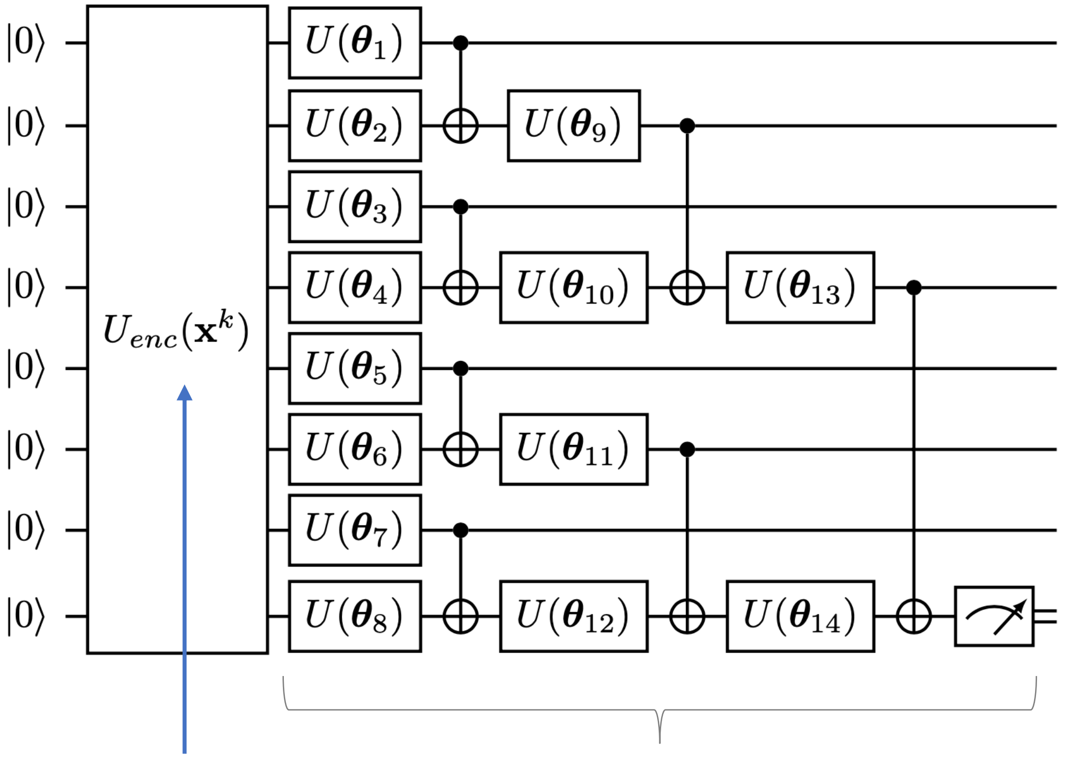
$n \propto \mathcal{O}(\log(N))$, $n_{gates} \propto \mathcal{O}(poly(N)) = \mathcal{O}(poly(2^n))$

Schuld, Maria, and Francesco Petruccione. *Supervised learning with quantum computers*. Vol. 17. Berlin: Springer, 2018.
Image source: Nielsen, Michael A., and Isaac Chuang. "Quantum computation and quantum information." (2002).

Quantum Classifier example: Quantum Tree Tensor Network

Quantum Tree Tensor Network with generic single-qubit unitary gates $U(\theta, \phi, \lambda)$

$$U(\theta,\phi,\lambda) = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i(\phi+\lambda)}\cos(\theta/2) \end{pmatrix}$$



Apply **QTTN as binary classifier**: measure one qubit

We encode our classical input features here

Variational part

See: Grant, Edward, et al. "Hierarchical quantum classifiers." *npj Quantum Information* 4.1 (2018): 65.

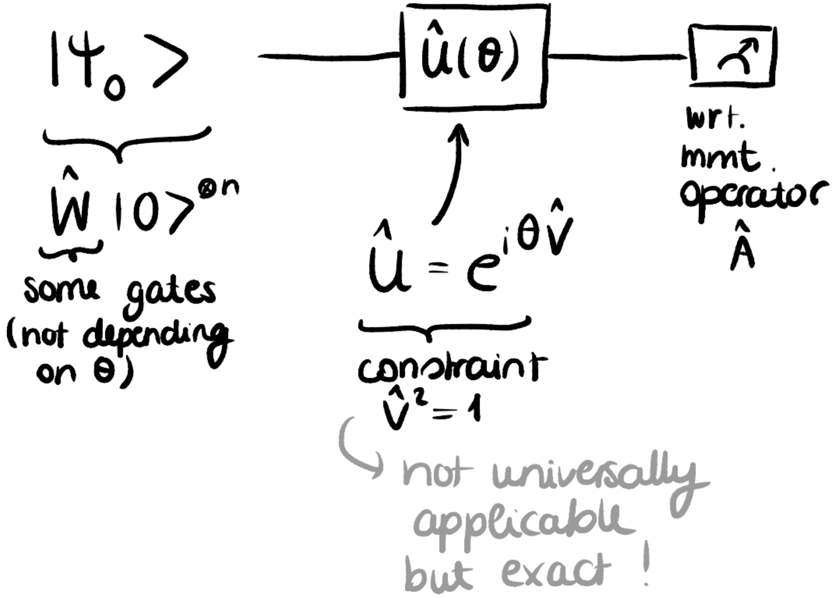
Parameter optimization

$$\theta \to \theta - \eta \nabla_\theta f$$

$$\uparrow \langle \hat{A}(\theta) \rangle$$

The parameter-shift rule (gradient-based)

→ Compute **partial derivative** of variational circuit parameter θ, alternative to analytical gradient computation and classical finite difference rule (numerical errors and resource cost considerations)



$$\Rightarrow \nabla_\theta \langle \hat{A} \rangle = u\left[\langle \hat{A}(\theta + \tfrac{\pi}{4u}) \rangle - \langle \hat{A}(\theta - \tfrac{\pi}{4u}) \rangle\right]$$

→ Evaluate Quantum Circuit twice at shifted parameters to compute gradient

Source: https://pennylane.ai/qml/demos/tutorial_stochastic_parameter_shift
https://pennylane.ai/qml/demos/tutorial_spsa

Parameter optimization

$$\theta_{k+1} \leftarrow \theta_k - a_k \underbrace{\hat{g}_k(\hat{\theta}_k)}_{\text{stochastic estimate of } \nabla_\theta f}$$

Simultaneous perturbation stochastic approximation (SPSA) (gradient-free)

⟶ If gradient computation not possible, too resource-intensive, or noise-robustness required (slower convergence but fewer function evaluations)

⟶ Gradient is approximated by two sampling steps and parameters are perturbed in all directions simultaneously

"$y(\theta) = f(\theta) + \varepsilon$" random output perturbation

$$\hat{g}_{ki}(\hat{\theta}_k) = \frac{y(\hat{\theta}_k + c_k\Delta_k) - y(\hat{\theta}_k - c_k\Delta_k)}{2c_k\Delta_{ki}}$$

Iterative update rule comparable to classical stochastic gradient descent

$c_k \geq 0$, $\Delta_k = (\Delta_{k_1}, \Delta_{k_2}, \dots, \Delta_{k_p})^T$ perturbation vector (~ randomly sampled from zero-mean distr.)

- Kiss O., Grossi M. et all., Conditional Born machine for Monte Carlo events generation, *Phys. Rev. A 106, 022612 (2022)*
- https://pennylane.ai/qml/demos/tutorial_stochastic_parameter_shift
- https://pennylane.ai/qml/demos/tutorial_spsa

Challenges when using Parametrized Quantum Circuits

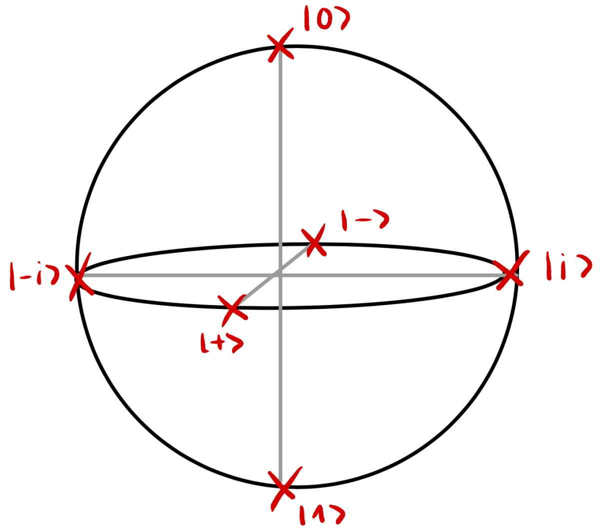
- Efficient **data handling** and data **embedding**
- Find balance: **Generalization** and **representational power** vs. **Convergence**
  - Problem of barren plateaus and vanishing gradients in optimization landscape
  - How well can we survey the Hilbert space (expressibility)?

- Current hardware limitations
  - Limited number of qubits and connectivity
  - **Quantum Noise Effects** (decoherence, measurement errors or gate-level errors)
  - Efficient interplay between classical and quantum computer
- ….

What is Quantum Advantage in QML?

Multiple considerations:

1. Runtime speed-up
2. Sample complexity
3. Representational power



Bloch sphere: only the marked points are produced by the Clifford operators acting on a computational basis state

This includes considerations regarding **classical intractability**:

Focus on Quantum Circuits that are **not efficiently simulable classically**

Nielsen, Michael A., and Isaac Chuang. "Quantum computation and quantum information." (2002).
Gottesman, Daniel. "The Heisenberg representation of quantum computers." *arXiv preprint quant-ph/9807006* (1998).
See also: - Kübler, Jonas, Simon Buchholz, and Bernhard Schölkopf. "The inductive bias of quantum kernels." *Advances in Neural Information Processing Systems* 34 (2021): 12661-12673.
- Huang, HY., Broughton, M., Mohseni, M. *et al.* Power of data in quantum machine learning. *Nat Commun* **12**, 2631 (2021). https://doi.org/10.1038/s41467-021-22539-9

Interlude: Efficient classical simulation of Clifford circuits

The Gottesman-Knill theorem

A quantum circuit build up of Clifford gates can be ***efficiently simulated*** *on a* ***classical computer****.*
(Qubit preparation and measurement in computational basis.)

There are more detailed considerations of cases with different computational complexities.

→ Even **highly entangled states** can be simulated efficiently classically.

Generating set of the Clifford group: $\langle \boldsymbol{H}, \boldsymbol{S}, \boldsymbol{CNOT} \rangle$

Nielsen, Michael A., and Isaac Chuang. "Quantum computation and quantum information." (2002).
Gottesman, Daniel. "The Heisenberg representation of quantum computers." *arXiv preprint quant-ph/9807006* (1998).

What is Quantum Advantage in QML?

Multiple considerations:

1. Runtime speed-up
2. Sample complexity
3. Representational power

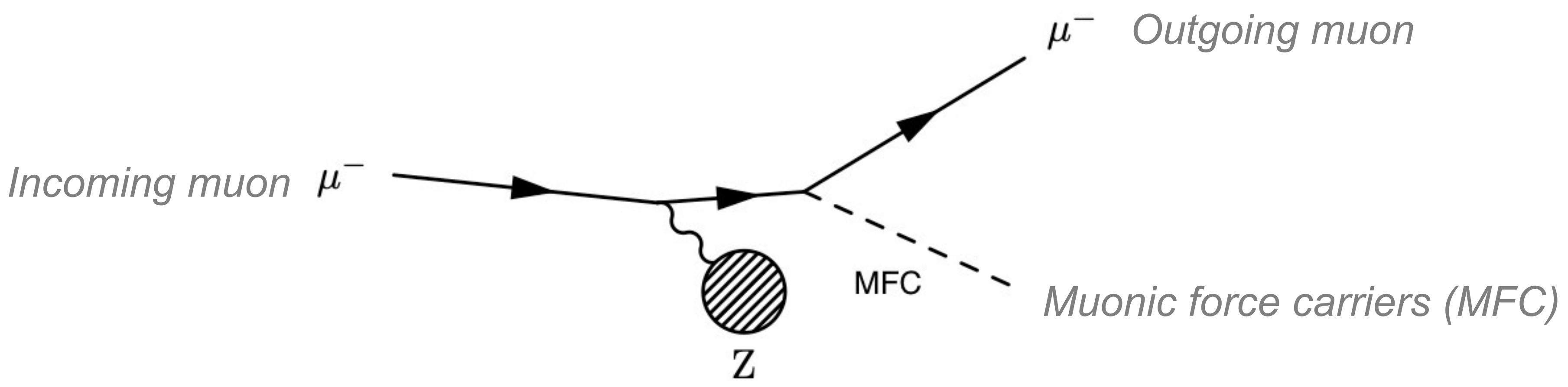
Practical advantage → Practical implementations on NISQ devices

→ *Need for performance metrics and fair comparisons to classical models*

Nielsen, Michael A., and Isaac Chuang. "Quantum computation and quantum information." (2002).
Gottesman, Daniel. "The Heisenberg representation of quantum computers." *arXiv preprint quant-ph/9807006* (1998).
See also: - Kübler, Jonas, Simon Buchholz, and Bernhard Schölkopf. "The inductive bias of quantum kernels." *Advances in Neural Information Processing Systems* 34 (2021): 12661-12673.
- Huang, HY., Broughton, M., Mohseni, M. *et al.* Power of data in quantum machine learning. *Nat Commun* **12**, 2631 (2021). https://doi.org/10.1038/s41467-021-22539-9

CERN use-cases

Quantum Circuit Born Machine for Event Generation



Muon fixed target scattering experiment

→ MFCs are **bosons** which appear in beyond-the-standard-model theoretical frameworks and are **candidates for dark matter**

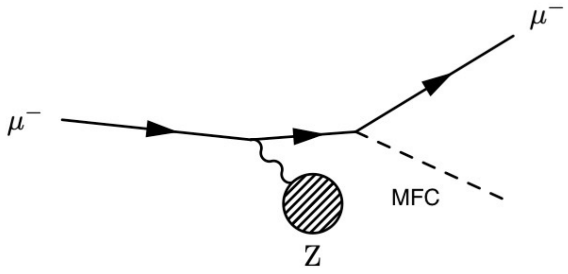
→ Monte Carlo calculations are **expensive** in time and CPU consumption

Kiss O., Grossi M. et all., **Conditional Born machine for Monte Carlo events generation**, *Phys. Rev. A* ***106****, 022612 (2022)*

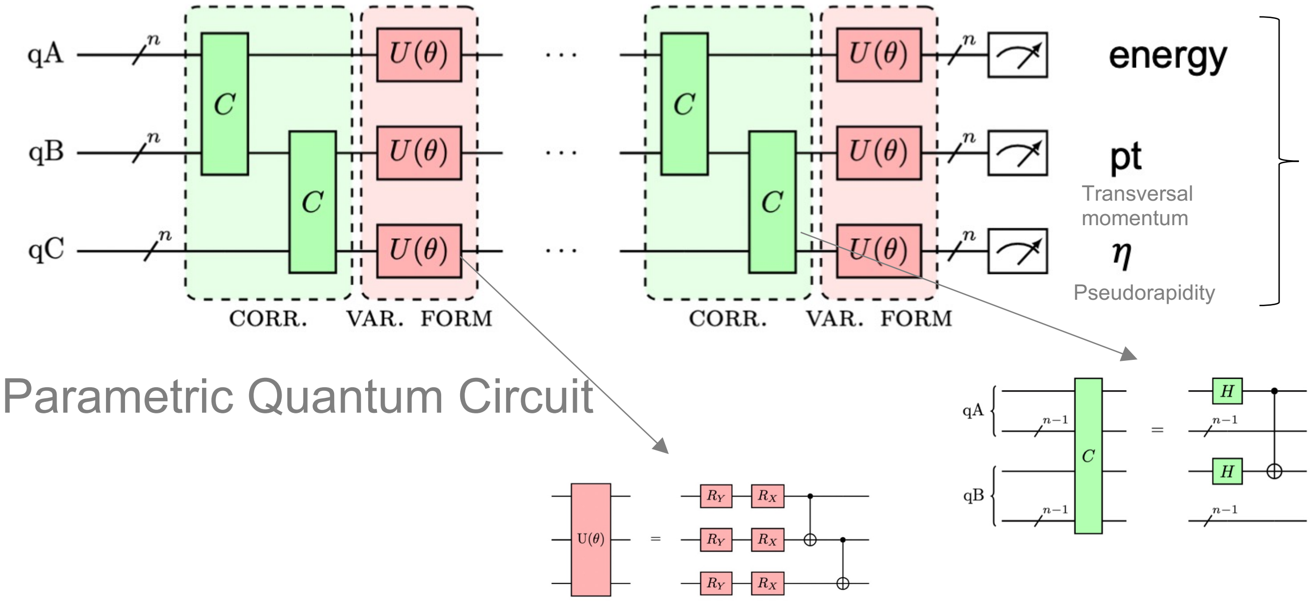
Quantum Circuit Born Machine for Event Generation

Born machine:
Produces statistics according to Born's measurement rule using parametrized quantum circuit $|\psi(\theta)\rangle$

$$p_\theta(x) = |\langle x|\psi(\theta)\rangle|^2,\ x \in \{0,1\}^{3n}$$



Muon fixed target scattering experiment



Generate discrete PDFs
(continuous in the limit increasing no. of qubits)

Parametric Quantum Circuit

Kiss O., Grossi M. et all.,
Conditional Born machine for Monte Carlo events generation,
Phys. Rev. A ***106****, 022612 (2022)*

Coyle, B., Mills, D. et al, **The Born supremacy.** In: *npj Quantum Inf 6, 60* (2020)

Quantum Circuit Born Machine for Event Generation

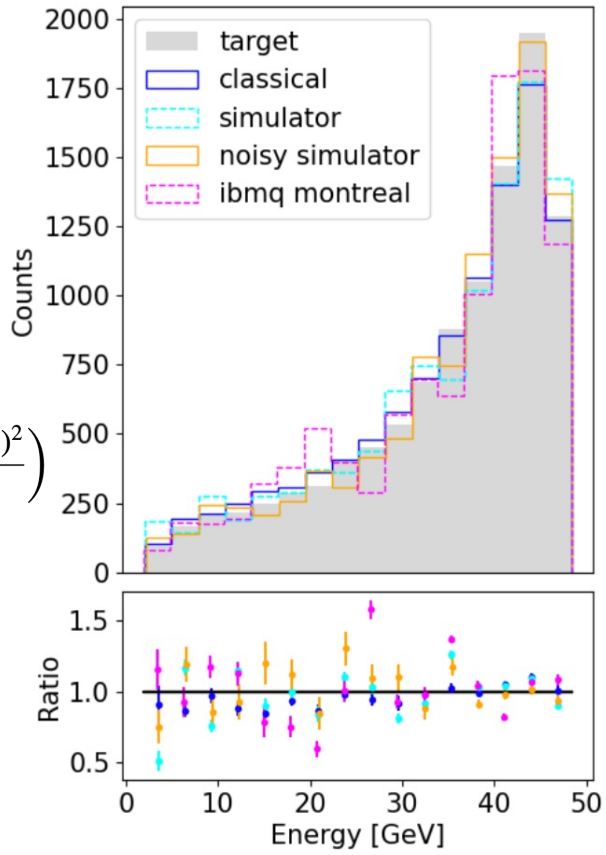
- Generate samples of discrete PDFs with Born machine
- Train using Maximum Mean Discrepancy loss function:

$$\mathrm{MMD(P,Q)} = \mathbb{E}_{\substack{X\sim P\\ Y\sim P}}[K(X,Y)] + \mathbb{E}_{\substack{X\sim Q\\ Y\sim Q}}[K(X,Y)] - 2\mathbb{E}_{\substack{X\sim P\\ Y\sim Q}}[K(X,Y)]$$

Gaussian kernel

$$K(x,y) = \exp\left(-\frac{(x-y)^2}{2\sigma}\right)$$

→ **efficient way to generate multivariate (and conditional) distributions** with only linear connectivity, suitable for NISQ devices (suggested by numerical evidence)

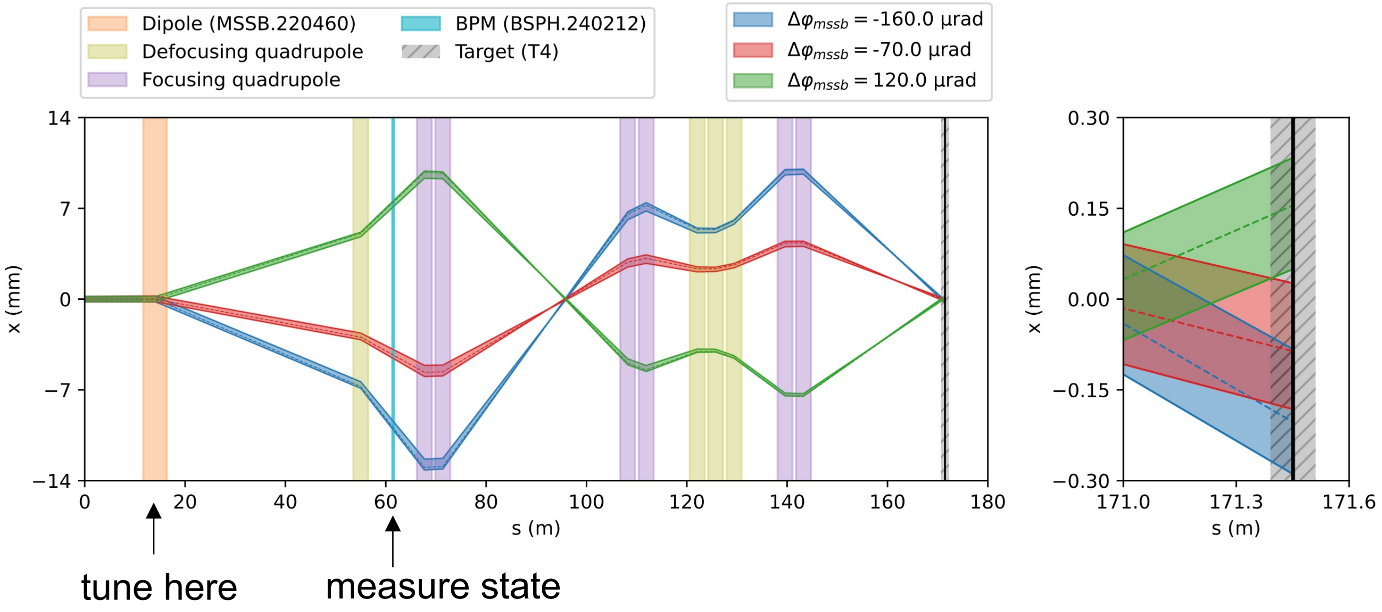


Kiss O., Grossi M. et all., **Conditional Born machine for Monte Carlo events generation**, *Phys. Rev. A* ***106****, 022612 (2022)*

Coyle, B., Mills, D. et al, **The Born supremacy.** In: *npj Quantum Inf 6, 60* (2020)

Quantum Reinforcement Learning (RL)

Michael Schenk et al., **Hybrid actor-critic algorithm for quantum reinforcement learning at CERN beam lines.** arXiv:2209.11044



Beam Target Steering Task

Formulate as RL problem:

- Action: (discrete) deflection angle
- State: (continuous) BPM position
- Reward: integrated beam intensity on target
- Optimality: fraction of states for which the agent takes the right decision

Quantum Reinforcement Learning (RL)

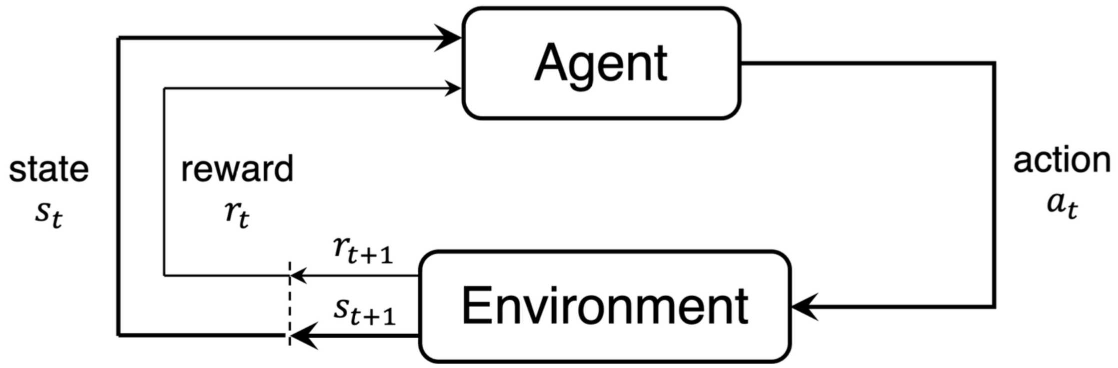
Michael Schenk et al., **Hybrid actor-critic algorithm for quantum reinforcement learning at CERN beam lines.** arXiv:2209.11044

Task: Beam optimization in linear accelerators

→ Use Reinforcement Learning (sample efficient)

Agent interacts with environment

- Follow **policy** $\boldsymbol{\pi(a_t|s_t)}$
- Goal: Find policy that **maximizes reward**



Schema of iterative Feedback-loop in RL

Expected reward is estimated by ***value function*** $\boldsymbol{Q(s,a)}$

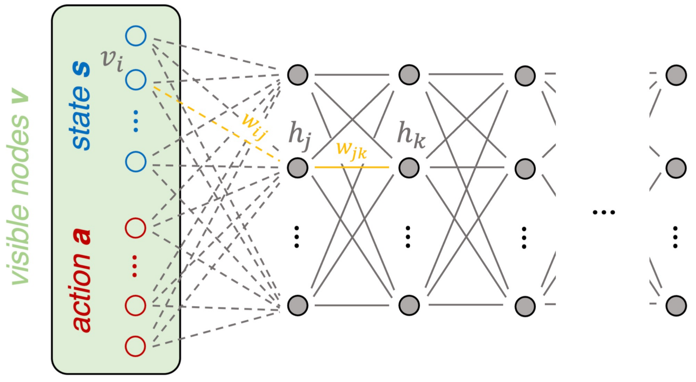
- **DQN:** Deep Q-learning *(NN-based)*
- **FERL:** Free energy-based RL *(clamped Quantum Boltzmann Machine)*

Structure of the **Quantum RL scheme**:

- Agent is **classical**
- Q-function is computed as the **energy of a qubit system**

Quantum Reinforcement Learning (RL)

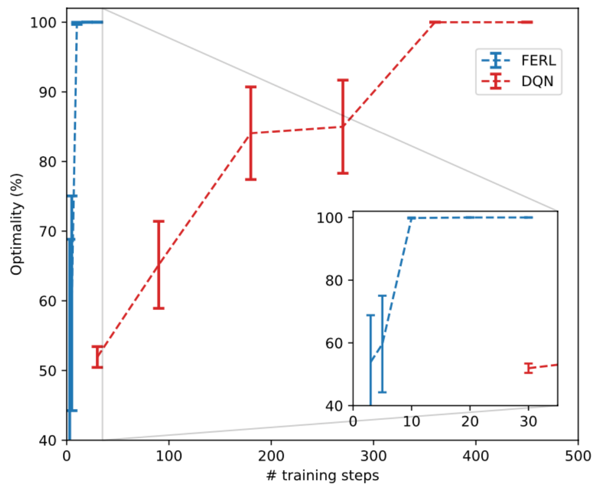
Michael Schenk et al., **Hybrid actor-critic algorithm for quantum reinforcement learning at CERN beam lines.** arXiv:2209.11044



Structure of clamped Quantum Boltzmann Machine (QBM)

→ Weights of QBM can be learned iteratively (analogous to classical Q-learning)

Quantum Annealing →



Convergence Study for one-dim. beam target steering task

→ **Quantum RL converges much faster than** classical Q-learning *(8±2 vs. 320±40 steps with e. r.)*

Transverse Field Ising model

$$\mathcal{H}(\mathbf{v}) = -\sum_{\substack{i\in V,\\ j\in H}} w_{ij} v_i \sigma^z_{h_j} - \sum_{j,k\in H} w_{jk}\sigma^z_{h_j}\sigma^z_{h_k} - \Gamma\sum_{j\in H}\sigma^x_{h_j}$$

$$\hat{Q}(s,a) \approx -F(\boldsymbol{v}) = -\langle H^{\mathrm{eff}}_{\boldsymbol{v}}\rangle - \frac{1}{\beta}\sum_{c}\mathbb{P}(c|\boldsymbol{v})\log\mathbb{P}(c|\boldsymbol{v})$$

Outlook on QML and summary

Research on QML applications in High Energy Physics is producing a **large number of prototypical algorithms for potential future use-cases**

- Current focus on *algorithms for data processing* in a *controlled* environment for current hardware
- Preliminary hints for advantage in terms of *representational power of quantum states*
- Mostly, algorithm performance is *as good as* the classical counterpart
- Need *more robust studies* to relate architecture of quantum computational model and its performance to data sets
- *Identify use-cases* where quantum approach is provably *more efficient* than classical model
- Studying QML algorithms today *links Quantum computing and Learning Theory* and draw separation between classical and quantum learner

Based on previous year's talk

Thank you,

are there any questions?

carla.sophie.rieger@cern.ch