Quantum Artificial Intelligence

Menica Dibenedetto

10 April 2025

Summer School 2025 - STIAS, Stellenbosch, 7-14 April 2025

Administrative info

- Instructor: Menica Dibenedetto (Assistant Professor, Maastricht University, NL)
- Communication: domenica.dibenedetto@maastrichtuniversity.nl

Administrative info

- 2 appointments (lecture/practical)
- Group projects/paper discussion

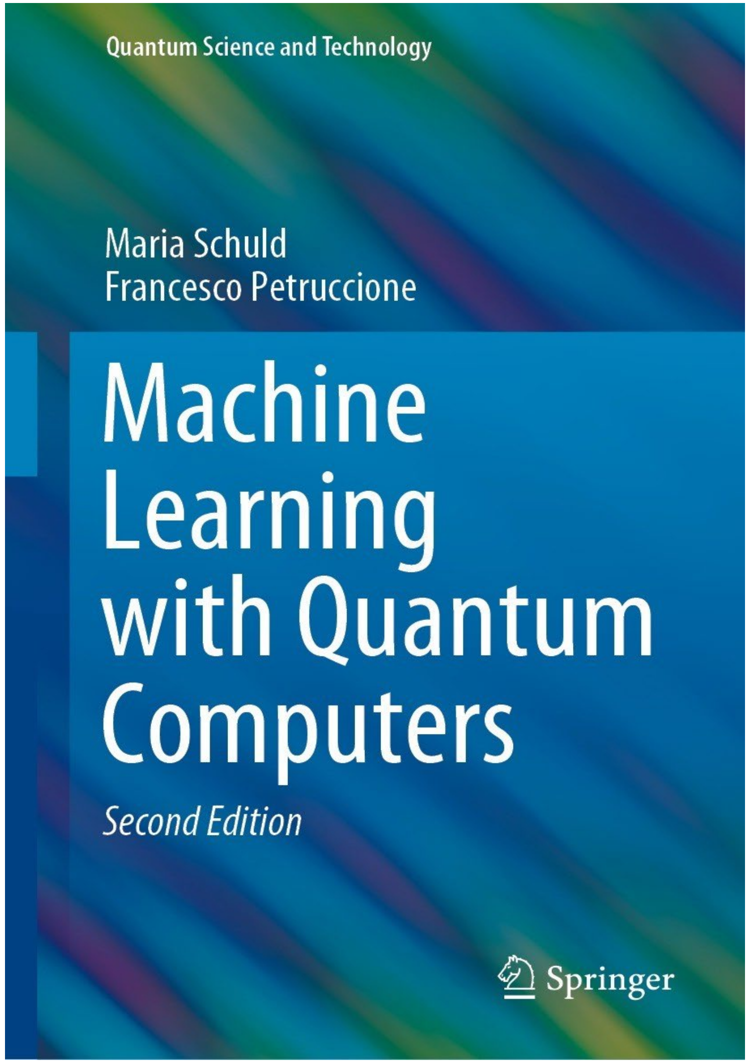
Who are you?

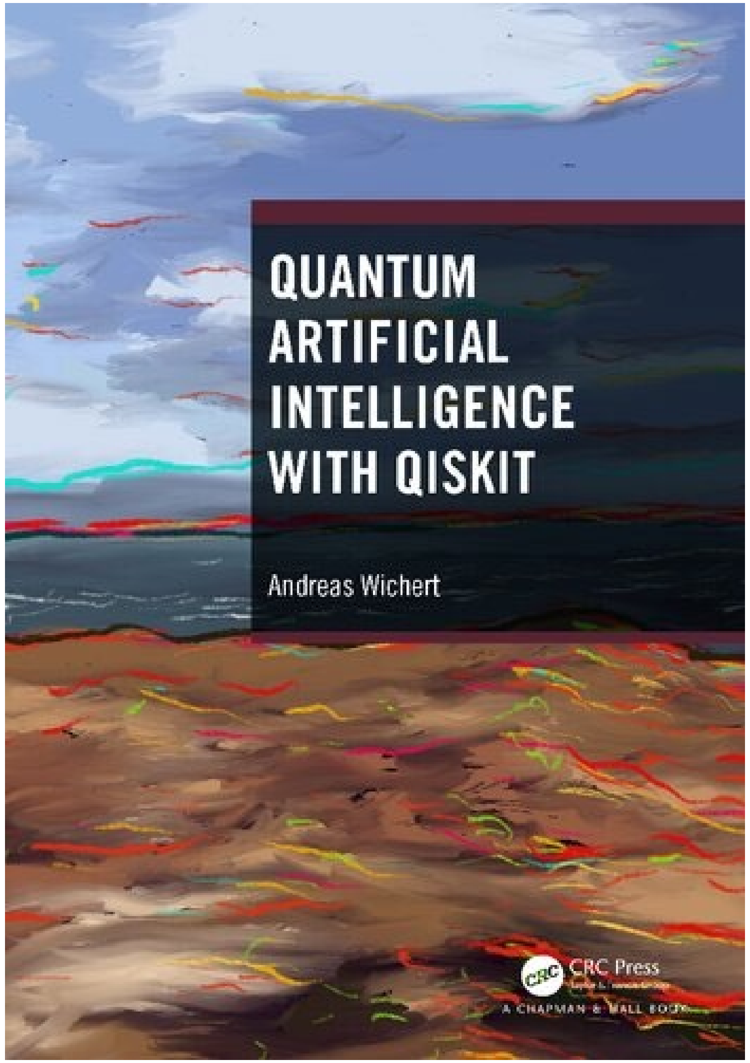
Go on

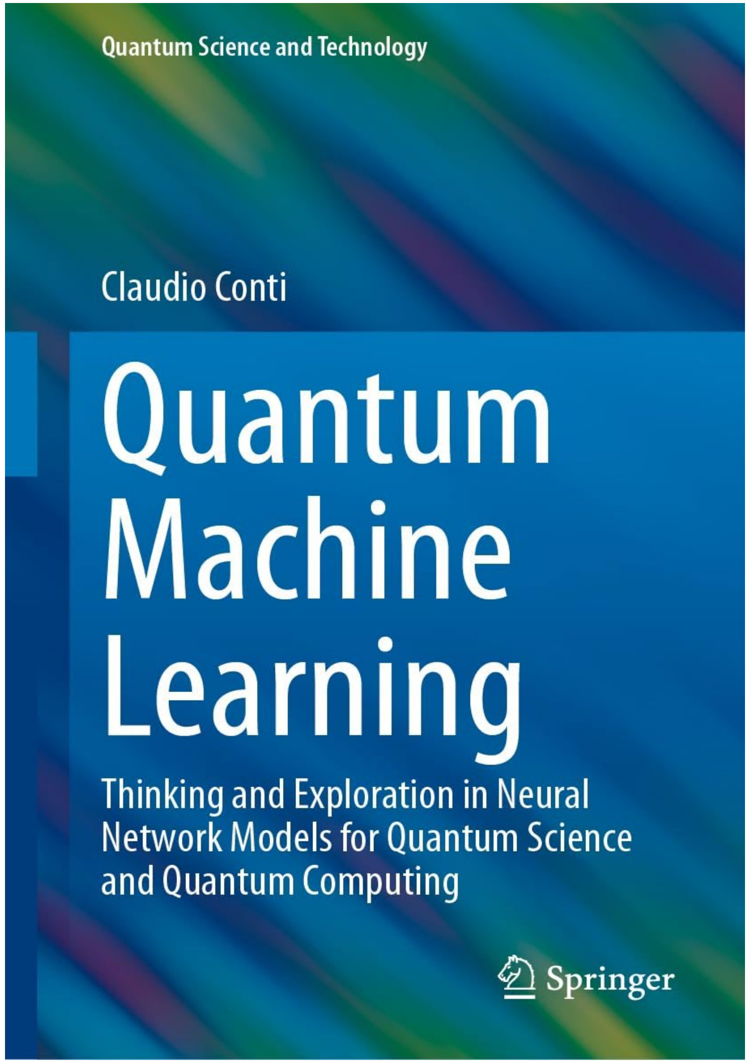
www.wooclap.com

Code: **UHHJST**

Sources


Quantum Science and Technology
Maria Schuld
Francesco Petruccione
Machine Learning with Quantum Computers
Second Edition
Springer


QUANTUM ARTIFICIAL INTELLIGENCE WITH QISKIT
Andreas Wichert
CRC Press
A CHAPMAN & HALL BOOK


Quantum Science and Technology
Claudio Conti
Quantum Machine Learning
Thinking and Exploration in Neural Network Models for Quantum Science and Quantum Computing
Springer

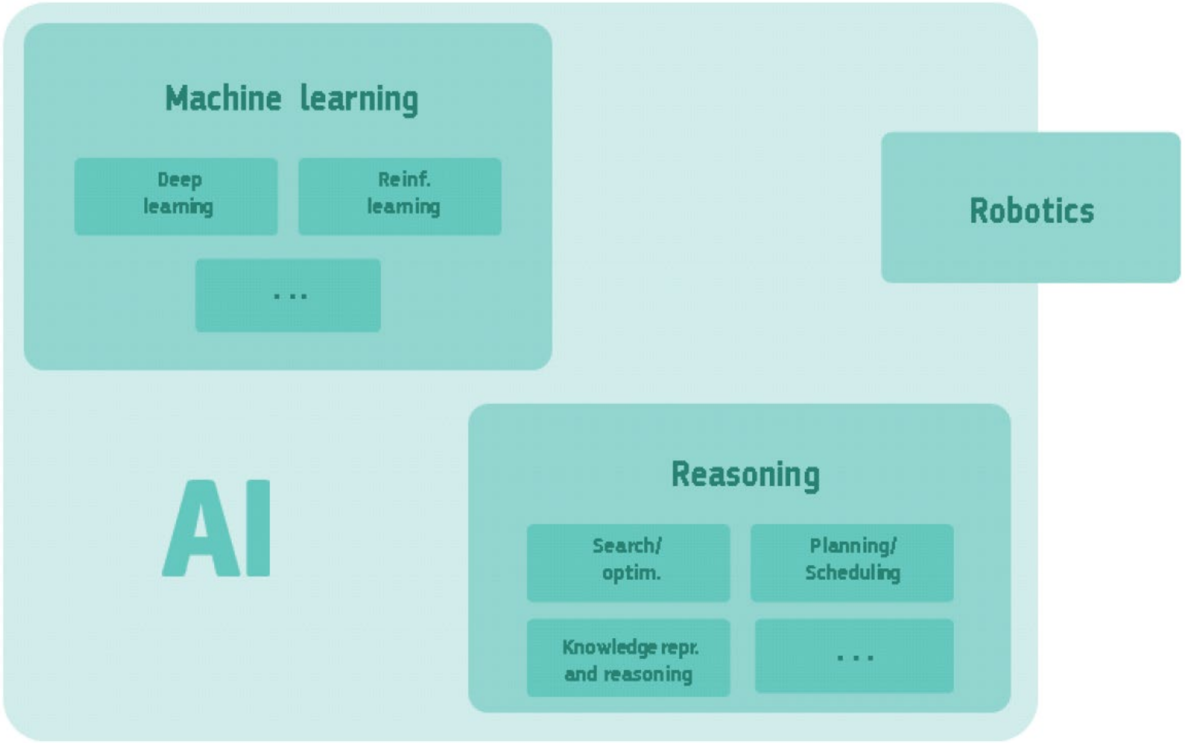
Artificial Intelligence (AI)

- AI was founded as a distinct discipline at the Dartmouth workshop in 1956.
- The term itself was invented by the American computer scientist John McCarthy and used in the title of the conference.
- AI is a subfield of computer science that models the mechanisms of intelligent human behavior.

Definition of Artificial Intelligence (AI)

“Artificial intelligence (AI) refers to systems that display intelligent behaviour by analysing their environment and taking actions – with some degree of autonomy – to achieve specific goals.

AI-based systems can be purely software-based, acting in the virtual world (e.g. voice assistants, image analysis software, search engines, speech and face recognition systems) or AI can be embedded in hardware devices (e.g. advanced robots, autonomous cars, drones or Internet of Things applications).”



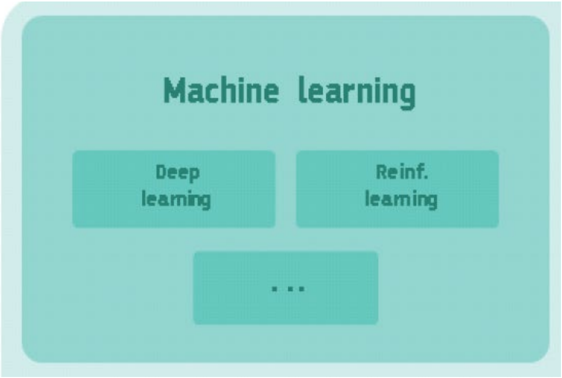
European Commission's Communication on AI

AI algorithms

Computationalism

- Symbolic AI
  - Symbolic representation of the domain in which the problems are solved.

Connectionism



- Statistical Machine Learning
  - Distributed representations

Information is physical

- Embodied intelligences

The Emergence of Quantum Artificial Intelligence

Early Foundations (1990s-2000s)

- 1996: Grover's Search Algorithm
- 2000s: Theoretical Expansion
- Notable Quantum hardware limitations

Rise of Quantum Machine Learning (2010s)

- 2011: D-Wave Systems introduces quantum annealers
- 2016: IBM releases its first cloud-accessible quantum processors
- 2017 onward: QML algorithms developed (Quantum-enhanced support vector machines and clustering)

Quantum AI Growth and Industry Support (2019-Present)

- 2019: Google's quantum supremacy milestone sparked industry interest Industry leaders (IBM, Google, Rigetti) released quantum ML libraries
- Key focus areas today: Quantum optimization, classification, generative models, new learning paradigms....

Questions in search of an answer

- Could the physical nature, as described by quantum physics, also lead to algorithms that imitate human behavior?
- What are the possibilities for the realization of artificial intelligence by means of quantum computation?
- We can add more….

Quantum Computing for AI

AI for Quantum Computing

*Quantum Machine Learning

Quantum Machine Learning

Motivations:

- Quantum Computing can perform linear algebra exponentially faster then classical computers
- Quantum system can generate patterns in data that classical system can't
- QML may be able to identify and classify patterns that are classically intractable

(Blu sky initiative, Michigan Engineering)

Machine Learning

- Supervised Learning P(Y|X)
  - Discriminative models, Classification, Regression,..
  - SVM, NN ...
- Unsupervised Learning P(X = x),
  - Discriminative and Generative Models, Clustering, Feutures extraction, Dimensionality Reduction,..
  - Boltzmann Machines, ...
- Reinforcement Learning (Interaction)
  - Agent–Environment paradigm

Quantum Machine Learning

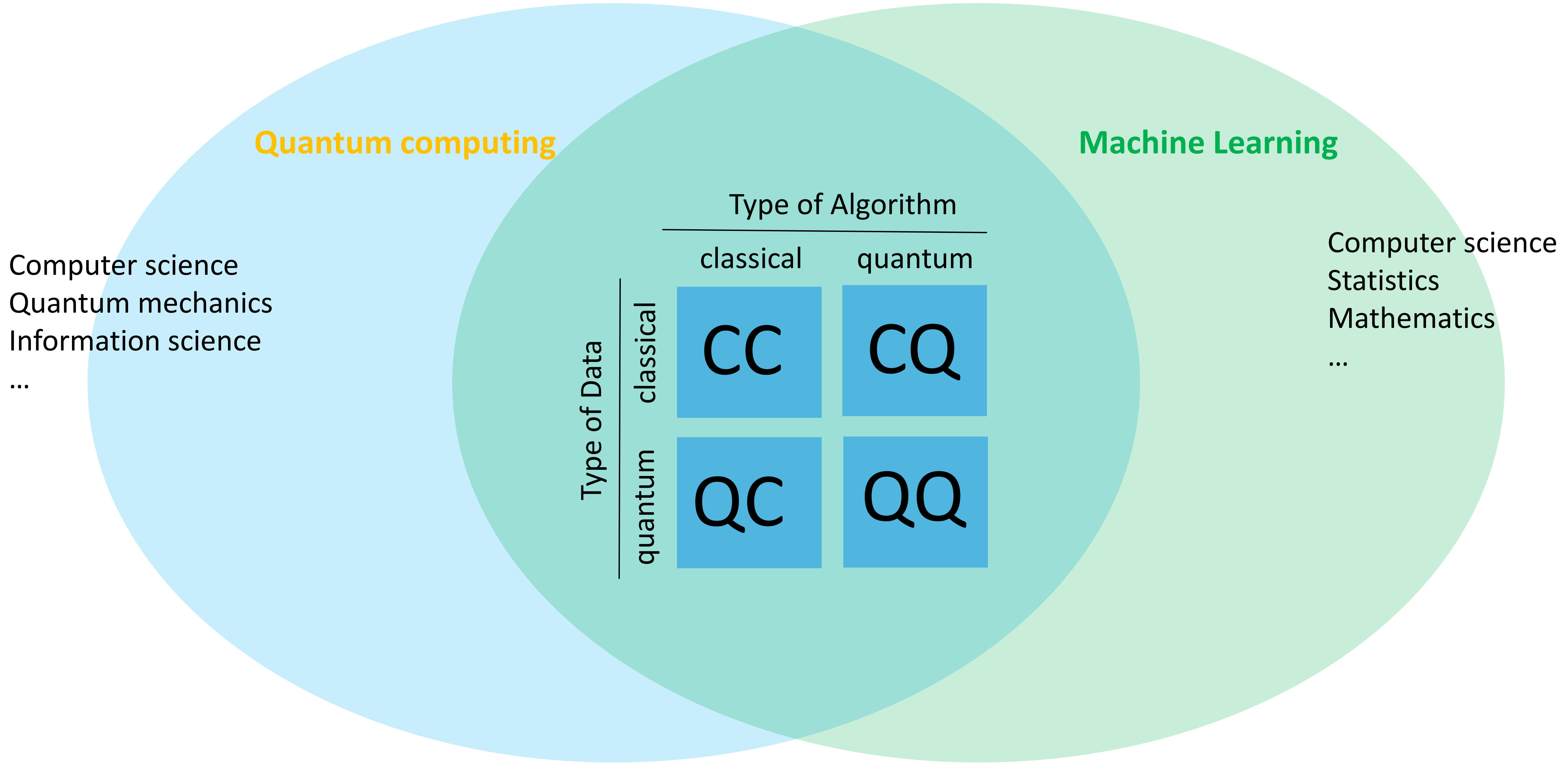


Image credit: Maria Schuld and Francesco Petruccione. Supervised learning with quantum computers. Vol. 17. Springer, 2018.

Quantum Machine Learning

Applications of ML in quantum physics

(1) Estimation and metrology
(2) Quantum control and gate design
(3) Controlling quantum experiments, and machine-assisted research
(4) Condensed matter and many body physics

Quantum enhancements for ML

(1) Quantum perceptrons and neural networks
(2) Quantum computational learning theory
(3) Quantum enhancement of learning capacity
(4) Quantum computational algorithmic speed-ups for learning

Quantum generalizations of ML-type tasks

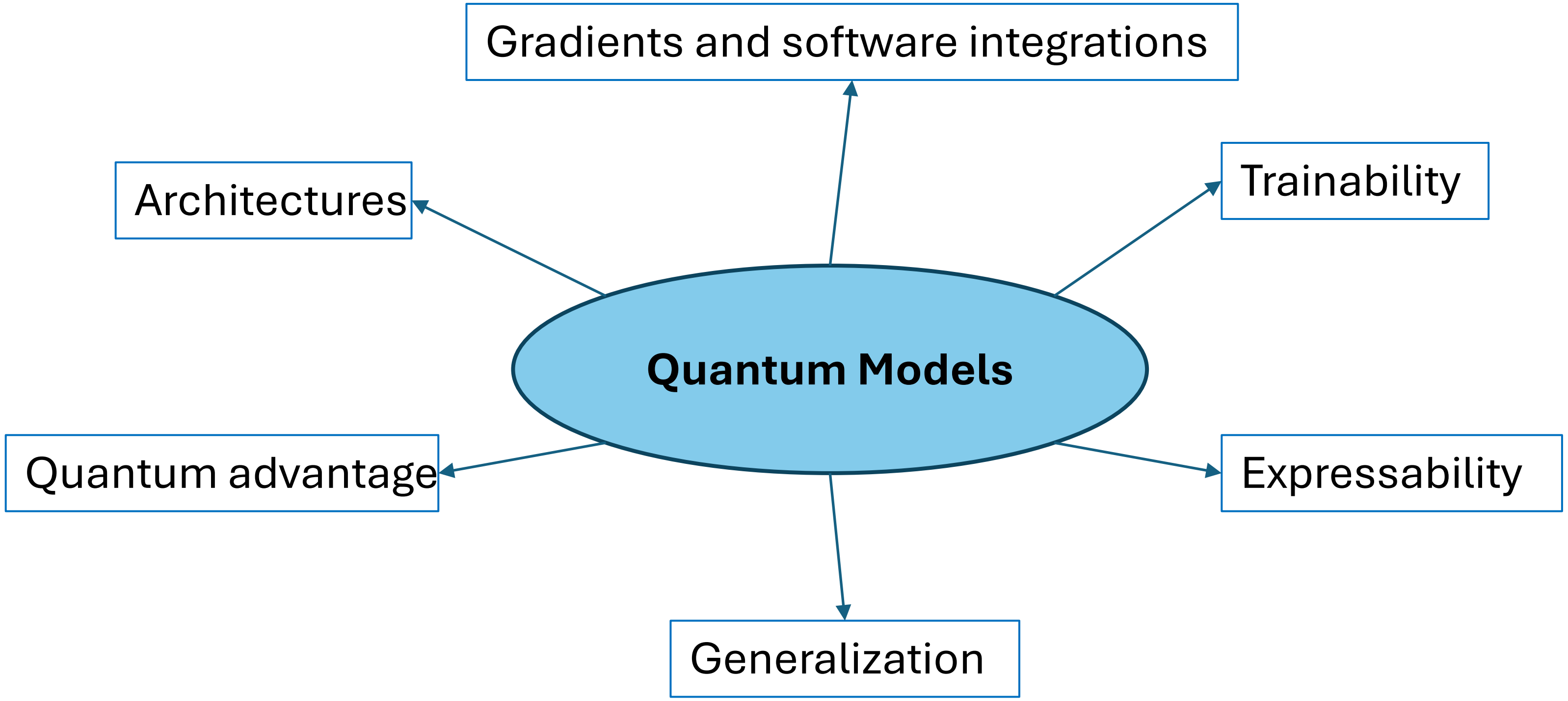
(1) Quantum generalizations: machine learning of quantum data
(2) (Quantum) learning of quantum processes

Quantum learning agents and elements of quantum AI

(1) Quantum-enhanced learning through interaction
(2) Quantum agent-environment paradigm
(3) Towards quantum AI

Machine learning & artificial intelligence in the quantum domain: a review of recent progress To cite this article: Vedran Dunjko and Hans J Briegel 2018 Rep. Prog. Phys. 81 074001

What to address?



Lectures Overview

- A bit of definitions
- Basic structure of a QML model
- Data Encoding
- Kernel-based methods and beyond
- Training and Expressability
- Open Quantum System for ML *Practical*

A bit of formalism...
States and Observables

- Quantum state $|\psi>\in\mathcal{H}$
- Observable O Hermitian in $\mathcal{H}$
- Norm, inner product $\langle\psi|\psi\rangle$

Computational basis

State |0> $=\begin{bmatrix}1\\0\end{bmatrix}$

State |1> $=\begin{bmatrix}0\\1\end{bmatrix}$

$$|\psi>= \alpha_1|0> +\alpha_2|1>$$

$$|\alpha_1|^2+|\alpha_2|^2=1 \qquad \alpha_i\epsilon\mathbb{C}$$

Classical stochastic -> Quantum

- $S = \{s_1,\dots s_N\}$ N events
- $M = \{m_1,\dots m_N\}$ N Random Variables

Matrix →
$$M = \begin{pmatrix} m_1 & & \\ & \ddots & \\ & & m_N \end{pmatrix}$$
Quantum →
Hermitian Operator
$M \in \mathbb{C}^{N\times N}$
$Eig(M) = \{m_1,\dots m_N\}$

- $P = \{p_1,\dots p_N\}$ Probability $\epsilon\ \mathbb{R}$, $p_i \geq 0, \sum p_i = 1$

Vector ↓

$\mathbb{R}^N$
$$\bar{p} = \begin{pmatrix} \sqrt{p_1} \\ \vdots \\ \sqrt{p_N} \end{pmatrix} = \sqrt{p_1}\begin{pmatrix} 1 \\ \vdots \\ 0 \end{pmatrix} + \cdots \sqrt{p_N}\begin{pmatrix} 0 \\ \vdots \\ 1 \end{pmatrix}$$

Quantum ↓

$\mathbb{C}^N$
$$\psi = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} \qquad |\alpha_i|^2 = p_i$$

Expected value $\langle M \rangle = \sum_i p_i m_i = p^T M p$ → $\langle M \rangle = \psi^T M \psi = \langle \psi | M | \psi \rangle$

Unitary evolutions

$$\begin{pmatrix} s_{11} & \cdots & s_{1K} \\ \vdots & \ddots & \vdots \\ s_{K1} & \cdots & s_{KK} \end{pmatrix} \begin{pmatrix} p_1 \\ \vdots \\ p_K \end{pmatrix} = \begin{pmatrix} p'_1 \\ \vdots \\ p'_K \end{pmatrix}, \qquad \sum_{k=1}^{K} p_k = \sum_{k=1}^{K} p'_k = 1$$

$$\begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix} \mathbf{p}_{\text{tod}} = \mathbf{p}_{\text{tom}}$$

Today observation

Tomorrow probability

60% stay the same

40% change

1

0

$$\begin{pmatrix} u_{11} & \cdots & u_{1K} \\ \vdots & \ddots & \vdots \\ u_{K1} & \cdots & u_{KK} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_K \end{pmatrix} = \begin{pmatrix} \alpha'_1 \\ \vdots \\ \alpha'_K \end{pmatrix}, \qquad \sum_{k=1}^{K} |\alpha_k|^2 = \sum_{k=1}^{K} |\alpha'_k|^2 = 1$$

Density matrix

State	Pure	Mixed
$\alpha = (\alpha_1, \alpha_2)$ $\beta = (\beta_1, \beta_2)$	$\rho = \alpha\alpha^\dagger$	$\rho = p_1 \alpha\alpha^\dagger + p_2\, \beta\beta^\dagger$

$$\langle M \rangle = \psi^T M \psi = \langle \psi | M | \psi \rangle = \mathrm{tr}\{\rho M\}$$

Measurement

Computational basis measurement

$$|\psi> = \alpha_1|0> + \alpha_2|1>$$

$P_0=|0><0|$

$P_1=|1><1|$

$p(0)=tr\{P_0|\psi><\psi|\}=<\psi|P_0|\psi>=|\alpha_1|^2$

$p(1)=|\alpha_2|^2$

$$\sigma_z = |0><0|+|1><1|$$

Quantum Models for AI

Near-term

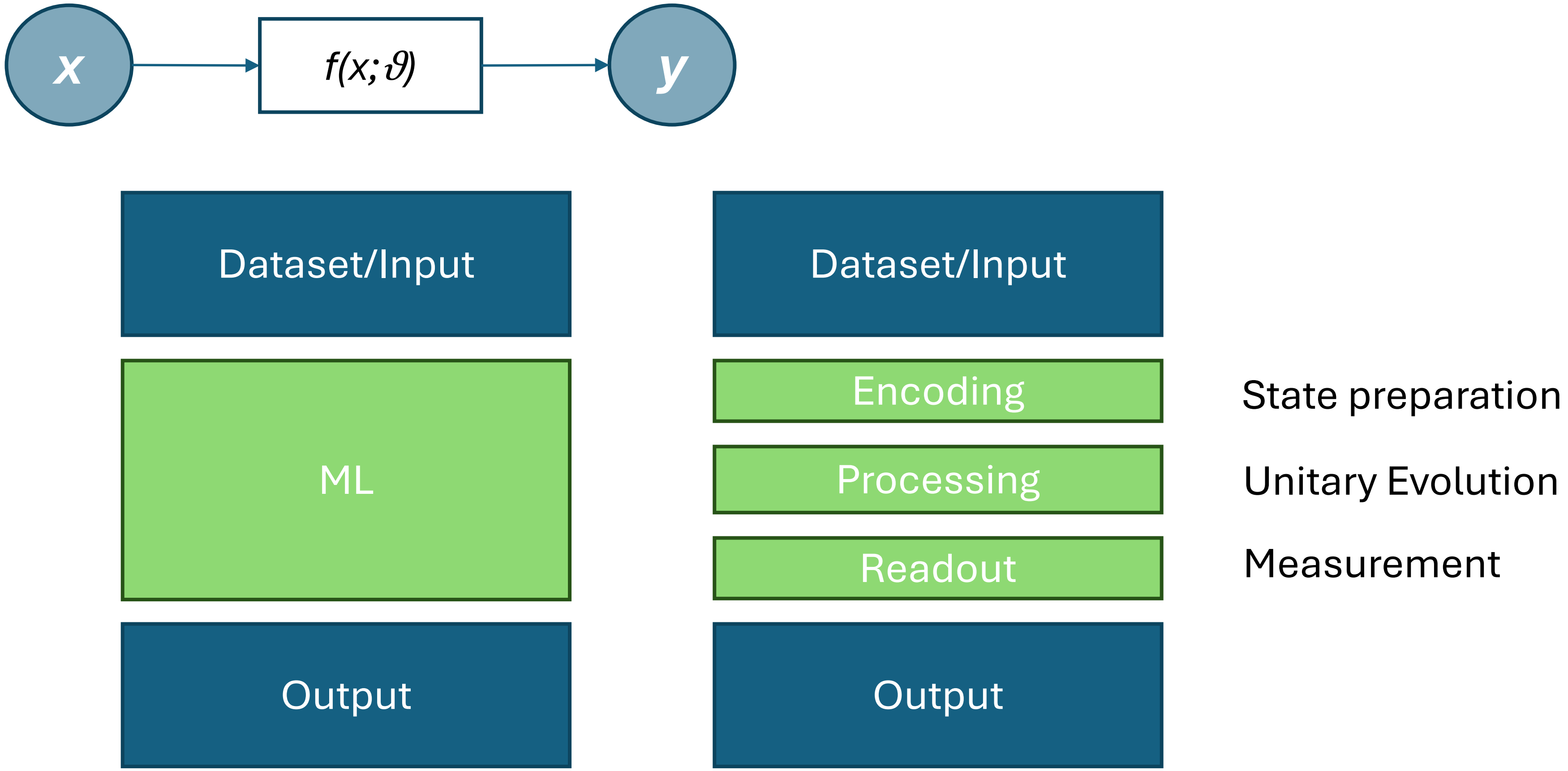
Hybrid

Fault-tolerant

Full quantum

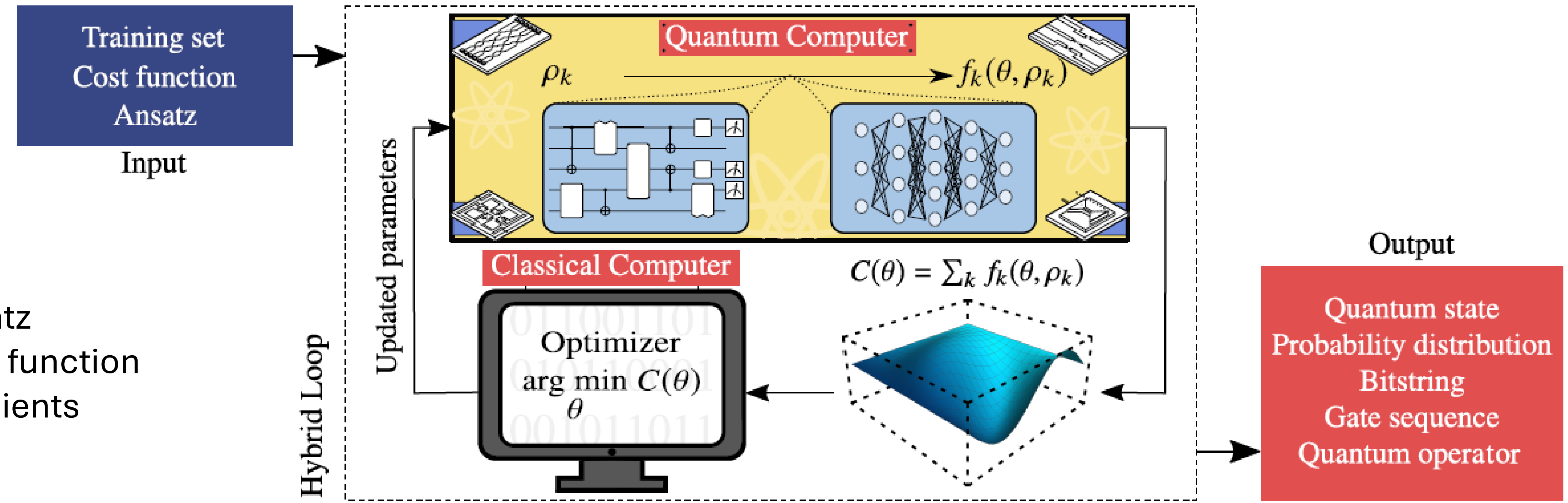
Quantum ML System

With classical data



NISQ Quantum Machine Learning

Variational Quantum Algorithm = Variational Circuits = Parametrized Circuits



- Ansatz
- Cost function
- Gradients

Cerezo, M., Arrasmith, A., Babbush, R. *et al.* Variational quantum algorithms. *Nat Rev Phys* **3**, 625–644 (2021). https://doi.org/10.1038/s42254-021-00348-9

Quantum Machine Learning models

Deterministic

Probabilistic

Implicit

Explicit

Data re-uploading

Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods**. Nat. Comm. (2023)

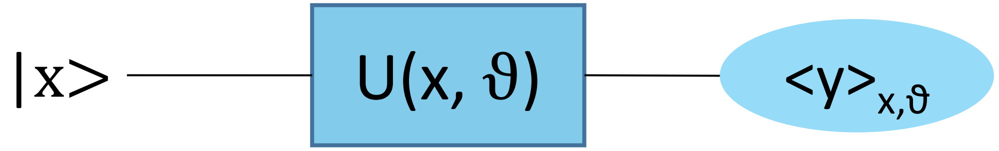
Quantum Machine Learning models

Deterministic

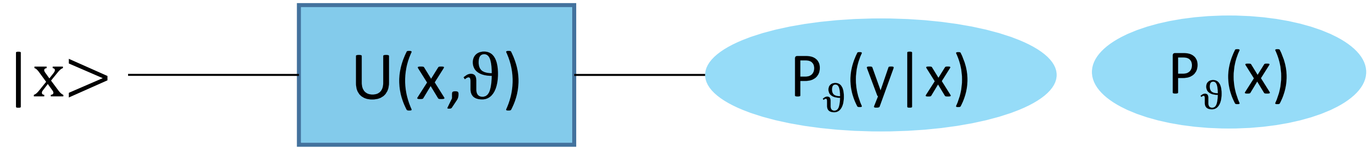
Probabilistic

Variational Quantum Circuits

- Deterministic quantum models
  - Example: Variational Quantum Classifier



- Probabilistic quantum models
  - Example: Variational Generator



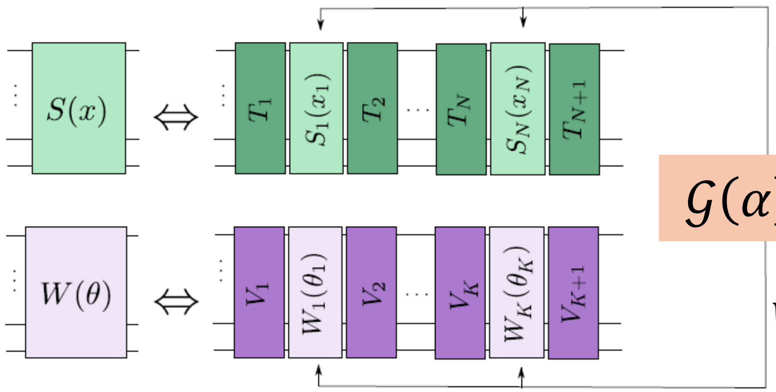
Deterministic quantum models

$\mathcal{X}$ input domain, $x\epsilon\mathcal{X}, \theta\epsilon\mathbb{R}^k$

$U(x,\theta)$ unitary $U(x,\theta) = \mathrm{W}(\theta)\mathrm{S}(\mathrm{x})$

$|\psi(x,\theta)\rangle = U(x,\theta)|0\rangle$

$\mathcal{M}$ Hermittian operator (observable)



$$S(x) = T_{N+1}\prod_{i=1}^{N} S_i(x_i)T_i$$

$$\mathcal{G}(\alpha) = e^{-i\alpha G}$$

$$W(\theta) = V_{K+1}\prod_{k=1}^{K} W_k(\theta)V_k$$

$$f(x)_\theta = \langle\psi(x,\theta)|\mathcal{M}|\psi(x,\theta)\rangle = tr\{\mathcal{M}\rho(x,\theta)\}$$

$$\rho(x,\theta) = U(x,\theta)^\dagger|0\rangle\langle 0|U(x,\theta)$$

Measurement in diagonal basis

$$\mathcal{M} = \sum_i \mu_i|\mu_i\rangle\langle\mu_i|$$

$$f(x)_\theta = \sum_i \mu_i|\langle\mu_i|\psi(x,\theta)\rangle|^2 = \sum_i \mu_i p(\mu_i)$$

$$\widehat{f(x)} = \frac{1}{S}\sum \mu^{(s)}$$

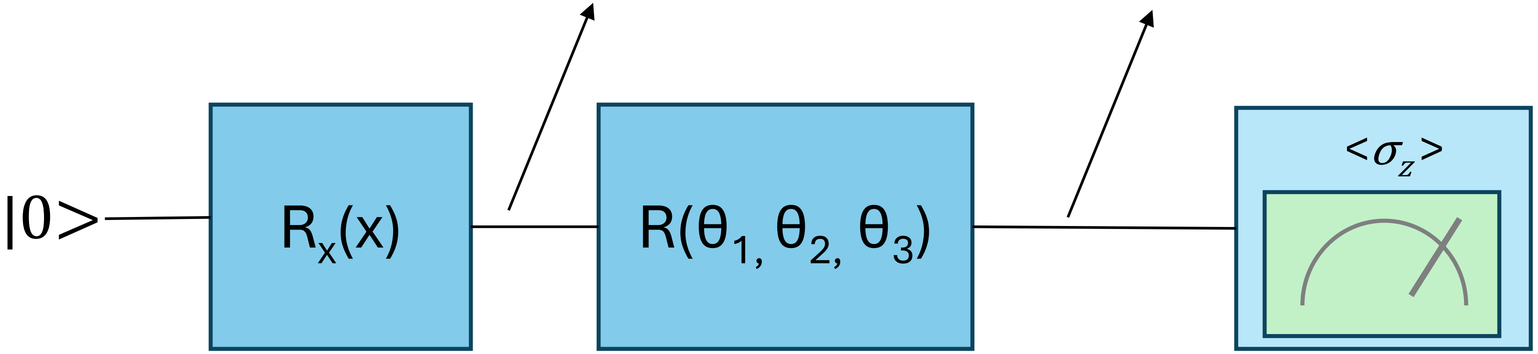
Example: Deterministic quantum models

Variational Quantum Classifier

$|\phi(x)\rangle = R_X(x)|0\rangle$

$|\psi(x, \theta)\rangle = R(\theta_1, \theta_2, \theta_3)\, |\phi(x)\rangle$

$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$



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

$$R_X(x) = \begin{bmatrix} \cos\frac{x}{2} & -i\sin\frac{x}{2} \\ -i\sin\frac{x}{2} & \cos\frac{x}{2} \end{bmatrix}$$

$$f_\theta(x) = \langle \psi(x, \theta) \mid \sigma_z \mid \psi(x, \theta) \rangle = \cos(\theta_2)\cos(x) - \sin(\theta_1)\sin(\theta_2)\sin(x)$$

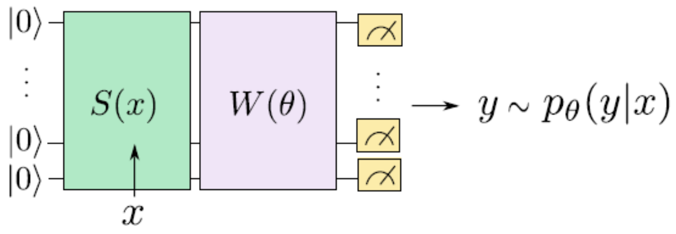
- Binary classifier
- Probabilistic classifier

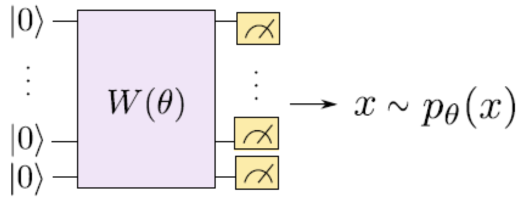
Probabilistic quantum models

Generative models

$\mathcal{X}$ *input domain*, $\mathcal{Y}$ *output domain*

$|\psi(x,\theta)\rangle = U(x,\theta)|0\rangle$





Born Machines

Supervised

Unsupervised

$$M = \sum_{y \epsilon \mathcal{Y}} y|y\rangle\langle y|$$

$$p(y|x) = |\langle y|\psi(x,\theta)\rangle|^2$$

$$M = \sum_{x \epsilon \mathcal{X}} x|x\rangle\langle x|$$

$$p(x) = |\langle x|\psi(\theta)\rangle|^2$$

$$f(x)_\theta = \langle x| \underbrace{|\psi(\theta)\rangle\langle\psi(\theta)|}_{\mathcal{M}} |x\rangle$$

Example: Probabilistic quantum models

Variational Generator

- Inspired by Boltzmann Machines
- Unsupervised

4 qubits: visible layer
3 qubits: hidden (unmeasured) layer

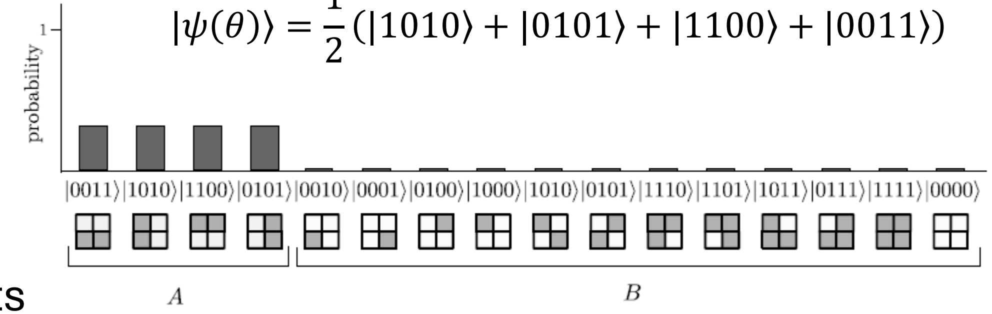
Basis state of 7 qubits → Images of 4 qubits

Injective mapping

$|0011\rangle$

Bar-or-strips

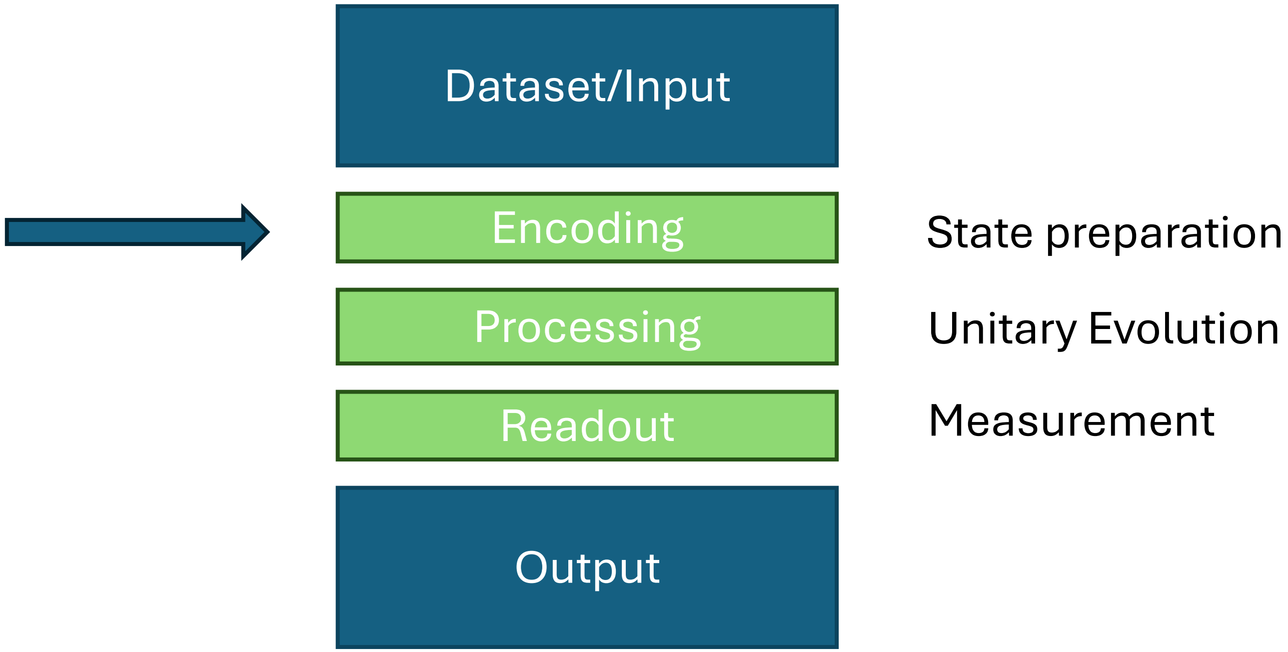
$$|\psi(\theta)\rangle = \frac{1}{2}(|1010\rangle + |0101\rangle + |1100\rangle + |0011\rangle)$$



$$|0000000\rangle \xrightarrow{W(\theta)} |\psi(\theta)\rangle = W(\theta)|0000000\rangle \longrightarrow \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$Model\colon\ p(x) = |\langle x|\psi(\theta)\rangle|^2, x\ \{0,1\}^{\otimes 4}$$

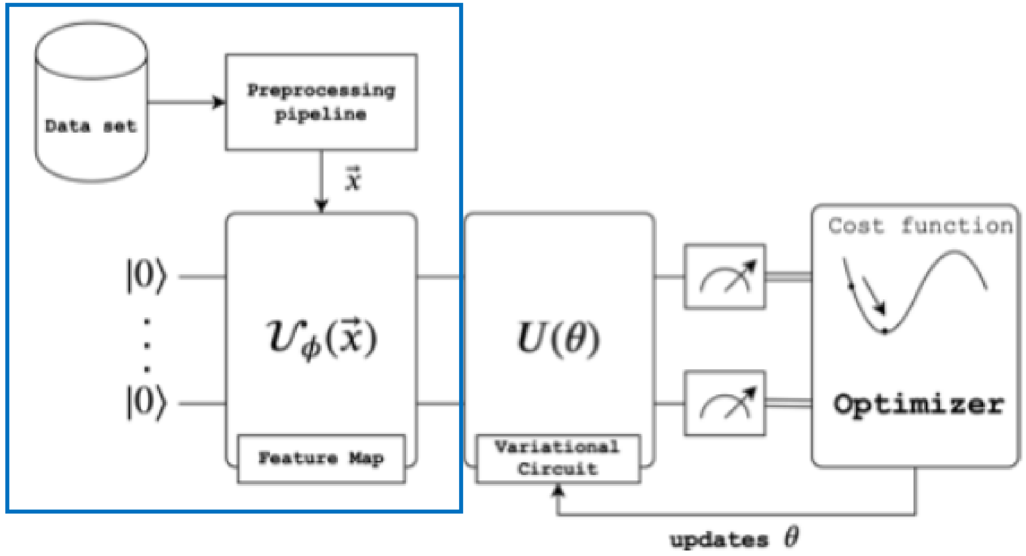
Quantum Machine Learning in NISQ



Data Encoding

- State preparation vs Data encoding
- Bottleneck for the runtime of the algorithm
  - *In QC* an efficient algorithm runs in polynomial time in the number of qubits
  - *In ML* an efficient algorithm runs in polynomial time in the dimension of the data inputs N and the number of data points M.
- Amplitude-efficient/qubits-efficient
- Data encoding -> Feature map -> Kernel methods

Data Encoding



- N qubits system in the ground state
- Data accessible form a classical memory
- Classical pre-processing? Sometimes it is needed
- Dataset of N-dim real-valued feature vectors
- Labels? Encoded in qubits entangled with inputs data

Data Encoding

- Basis Encoding
- Amplitude Encoding
- Angle (or phase or rotation) Encoding
- Hamiltonian Encoding

Encoding	# qubits	Runtime	Input type
Basis	$N\tau$	$\mathcal{O}(N\tau)$	Single input (binary)
Amplitude	$\log N$	$\mathcal{O}(N)$/ $\mathcal{O}(\log(N))^{a}$	Single input
Angle	N	$\mathcal{O}(N)$	Single input
Hamiltonian	$\log N$	$\mathcal{O}(MN)$/ $\mathcal{O}(\log(MN))^{a}$	Entire dataset

Advanced data encoding for Image Representation

- NEQR: Novel Enhanced Quantum Representation
- QPIE: Quantum Probability Image Encoding
- FRQI: Flexible Representation for Quantum Images
- OQIM: Order-Encoded Quantum Image Model

Basis Encoding

- Convert integer to binary representation

$x \to b_s b_{\tau_l - 1} \dots b_1 b_0 \cdot b_{-1} b_{-2} \dots b_{-\tau_r}$, where $\tau = 1 + \tau_l + \tau_r$

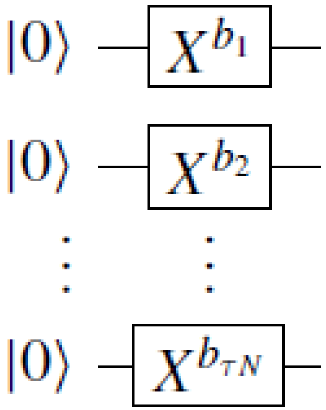
- Convert binary in quantum state
- The amplitude just mark the result of computation
- Qubit-efficient (n gates at most)

Advantages: Ease of preparation

Disadvantages: Qubit count

Basis Encoding

- Simple Algorithm: flip the qubits representing non-zero bits



$$U(b) = \prod_{i=0}^{\tau N} X^{b_i}$$

Basis Encoding

- How to encode a Dataset $\mathcal{D}$?

$$x^m \epsilon\, \mathcal{D} :\; b_m = (b_1^m, \dots, b_n^m),\; b_i^m \epsilon \{0,1\} \text{ for i=1,...n}$$

$$|\mathcal{D}\rangle = \frac{1}{\sqrt{M}} \sum_{m=1}^{M} |x^m\rangle$$

Sparse!

Amplitude encoding

- Use the amplitude of a quantum state to represent classical data
- Step 1: normalize it to unit length
- Step 2: pad it to zeros if required

Advantages: fewer qubits n=logN, n=logNM, N input features, M instances

Disadvantages: preparation, readout

Amplitude encoding

- Vector

$$|\psi_X\rangle = \sum_{i=0}^{N-1} x_i |i\rangle$$

- Dataset

$$|\psi_{\mathcal{D}}\rangle = \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} |\psi_{X^m}\rangle |m\rangle$$

Amplitude vector of dimension NM

$$\alpha = (x_1^1, \dots x_N^1, \dots, x_1^M, \dots, x_N^M)^T$$

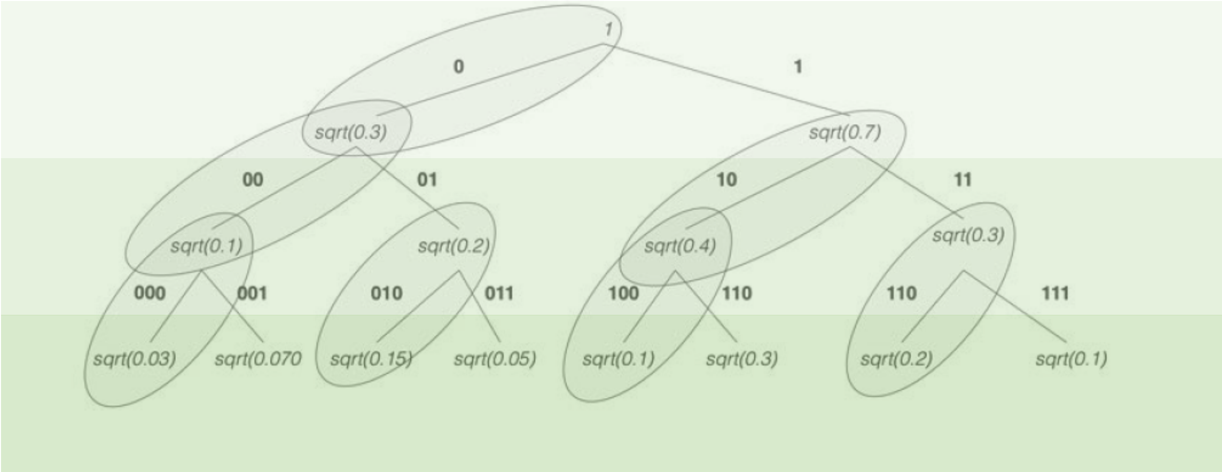
How to prepare this arbitrary state?

$$|\psi\rangle = \sum_i \alpha_i |i\rangle$$

Amplitude-Efficient state preparation

- Top-down Binary tree (Möttönen)

Cascade of multi-controlled Rotations



Level 1 $|q_2\rangle$

Level 2 $|q_2 q_1\rangle$

Level 3 $|q_2 q_1 q_0\rangle$

Requires an exponential number of operations regarding the number of qubits

Angle encoding

- Angle more suitable for representing continues values
- Manipulate the phase relationship between qubits using rotation gates not directly changing their amplitudes

Advantages: linear time in the number of features and qubits

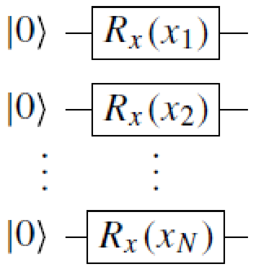
Disadvantages: expensive to encode a dataset at once

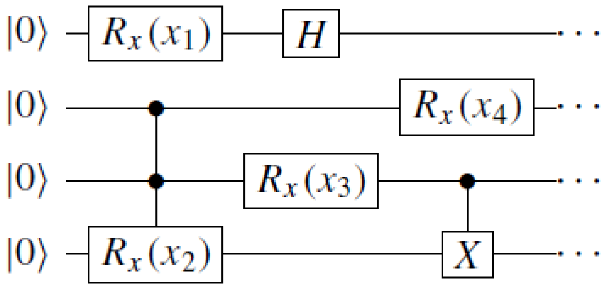
Angle encoding

- Scalars can be associated with the time t when implementing a unitary transformation $U(t) = e^{-itH}$ defined by an Hamiltonian H.
- A subclass of this strategy uses Pauli rotation gates
- Rotation can be Controlled on qubits

$$x = (x_1, x_2, \ldots, x_N)$$

$$U(x) = R_x(x_1) \otimes \ldots \otimes R_x(x_N)$$





Hamiltonian encoding

First approach

Ising based: map the problem to the Ising model

$$H = -\sum_{i,j} \alpha_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i$$

Advantages: easy to implement

Disadvantages: limited scope

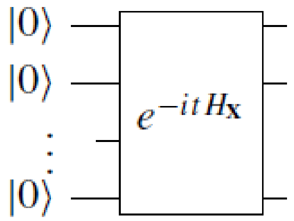
Hamiltonian encoding

Second approach

- Ginen X: Dataset dim MxN, rows: feature vectors
- Associate the Hamiltonian H with a matrix X
- Pre-processing tricks that embed the data matrix into Hermitian matrix

$$H_X = \begin{pmatrix} 0 & X \\ X^\dagger & 0 \end{pmatrix}$$

$$|\psi'\rangle = e^{-iH_x t}|\psi\rangle$$



- Hamiltonian encoding allows us to extract eigenvalues of matrices, or to multiply them to an amplitude vector.

Advantages: natural to encode

Disadvantages: hardware constraints and conditions

Why data encoding is important

- Interpreting data encoding as feature map
- Feature map change the structure of the data in a non-trivial manner
- Except for amplitude encoding, all the strategies perform non-linear operations in the data

Encoding from a conceptual picture

- Map from input space to state space of the quantum system
- If an *inner product* is defined on the state space: *feature map*

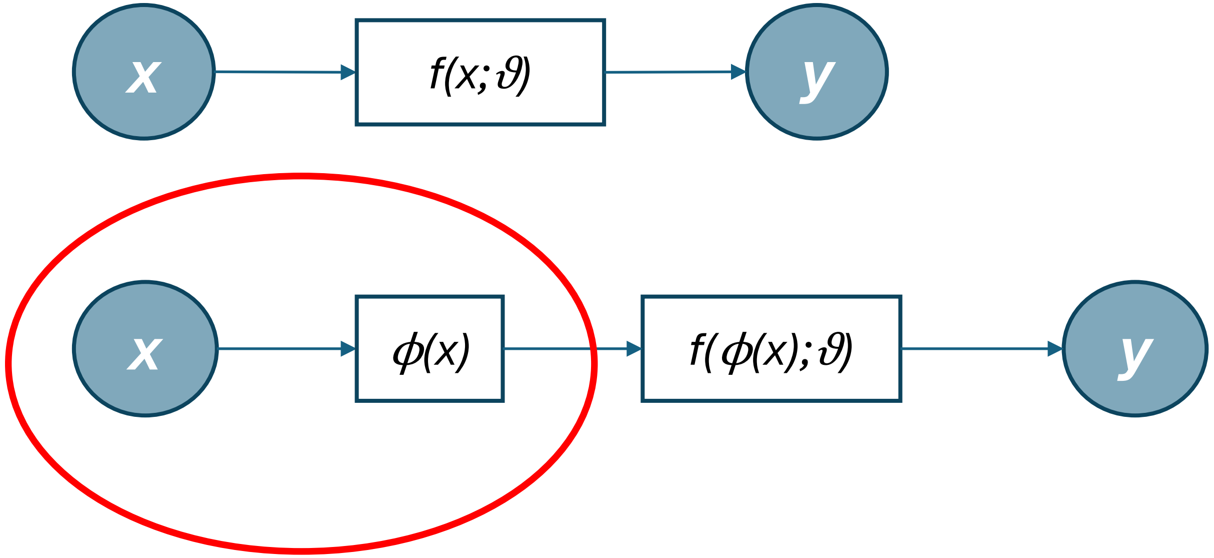
Inner product = distance -> learning

- Feature map play a role in Kernel-based ML models

Kernel-based Quantum Models

Kernel Theory ⟷ Quantum models

Feature maps



Kernel Methods in ML

- Solve ML tasks based on the idea of a Similarity Measure (Kernel)

Kernel Methods in ML

- Solve ML tasks based on the idea of a Similarity Measure (Kernel)
- Similarity measure

Name	Kernel	Hyperparameters
Linear	$\mathbf{x}^T\mathbf{x}'$	–
Polynomial	$(\mathbf{x}^T\mathbf{x}' + c)^p$	$p \in \mathbb{N}, c \in \mathbb{R}$
Gaussian	$e^{-\gamma \|\|\mathbf{x}-\mathbf{x}'\|\|^2}$	$\gamma \in \mathbb{R}^+$
Exponential	$e^{-\gamma \|\|\mathbf{x}-\mathbf{x}'\|\|}$	$\gamma \in \mathbb{R}^+$
Sigmoid	$\tanh(\mathbf{x}^T\mathbf{x}' + c)$	$c \in \mathbb{R}$

Kernel Definition

Let $\mathcal{X}$ be a non-empty set (input domain). A function

$$\kappa : \mathcal{X} x \mathcal{X} \rightarrow \mathbb{R}$$

Is called Kernel if the Gram matrix K with entries

$$K'_{m,m} = \kappa(x^m, x^{m\prime})$$

Is *positive semi-definite*

For any set of inputs $\mathcal{D} = \{x^1, \dots, x^M\} \subseteq \mathcal{X}$ and complex numbers $c_1, \dots, c_M$ we have that:

$$\sum_{m,m'=1}^{M} c_m c^*_{m'} \kappa(x^m, x^{m\prime}) \geq 0$$

Feature map Definition

- Let $\mathcal{X}$ be a non-empty set (input domain)
- $\mathcal{H} \subseteq \mathbb{R}^{\mathrm{n}}$ is the feature space
- Feature map $\phi\colon \mathcal{X} \to \mathcal{H} \subseteq \mathbb{R}^{\mathrm{n}}$
- $x \to \phi(x)$ non-linear: $\phi(\lambda x + \mu y) \neq \lambda\phi(x) + \mu\phi(y)$

Kernel $\leftrightarrow$ Feature map

- Kernel function can always be written as the *inner product* of data mapped in a suitable feature space by a *feature map*

$$k(x, y) \leftrightarrow (\phi, \mathcal{H})$$
$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$$

- Expressing a model in terms of kernel function allows us to use the *kernel trick* to turn one model into another by replacing kernel

Linear model

- Let’s consider a supervised learning tasks
- Given a dataset of labeled sample (labeled by an unknown function)
  - Label discrete: classification
  - Label continuous: regression

Linear Classifiers

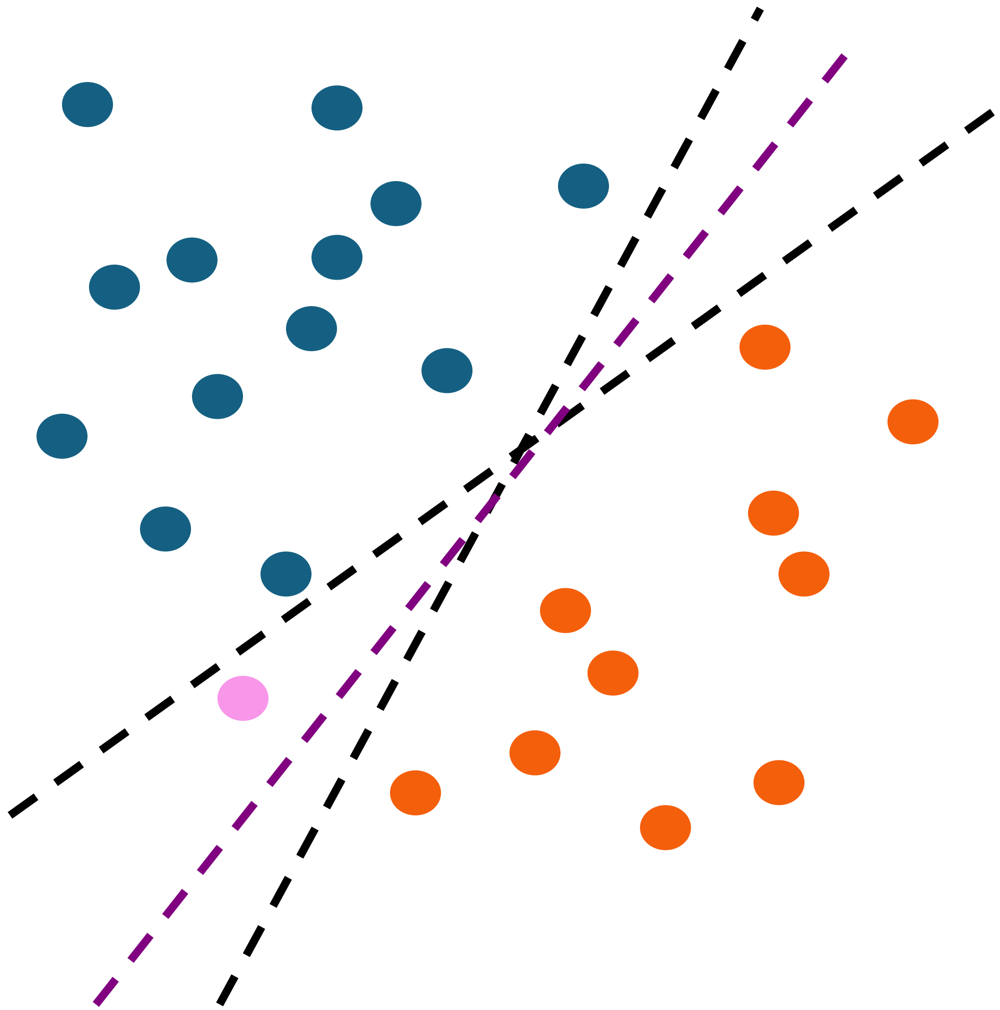
Given a set of labeled points

$$(\vec{x}_i, y_i)_{i=1\ldots N}$$

$\vec{x}_i \in \mathbb{R}^n \qquad y_i \in \{+1, -1\}$

We want to predict the label of an unseen point

Find a hyperplane which separates our labeled data. Use this as our decision rule



Linear Classifiers

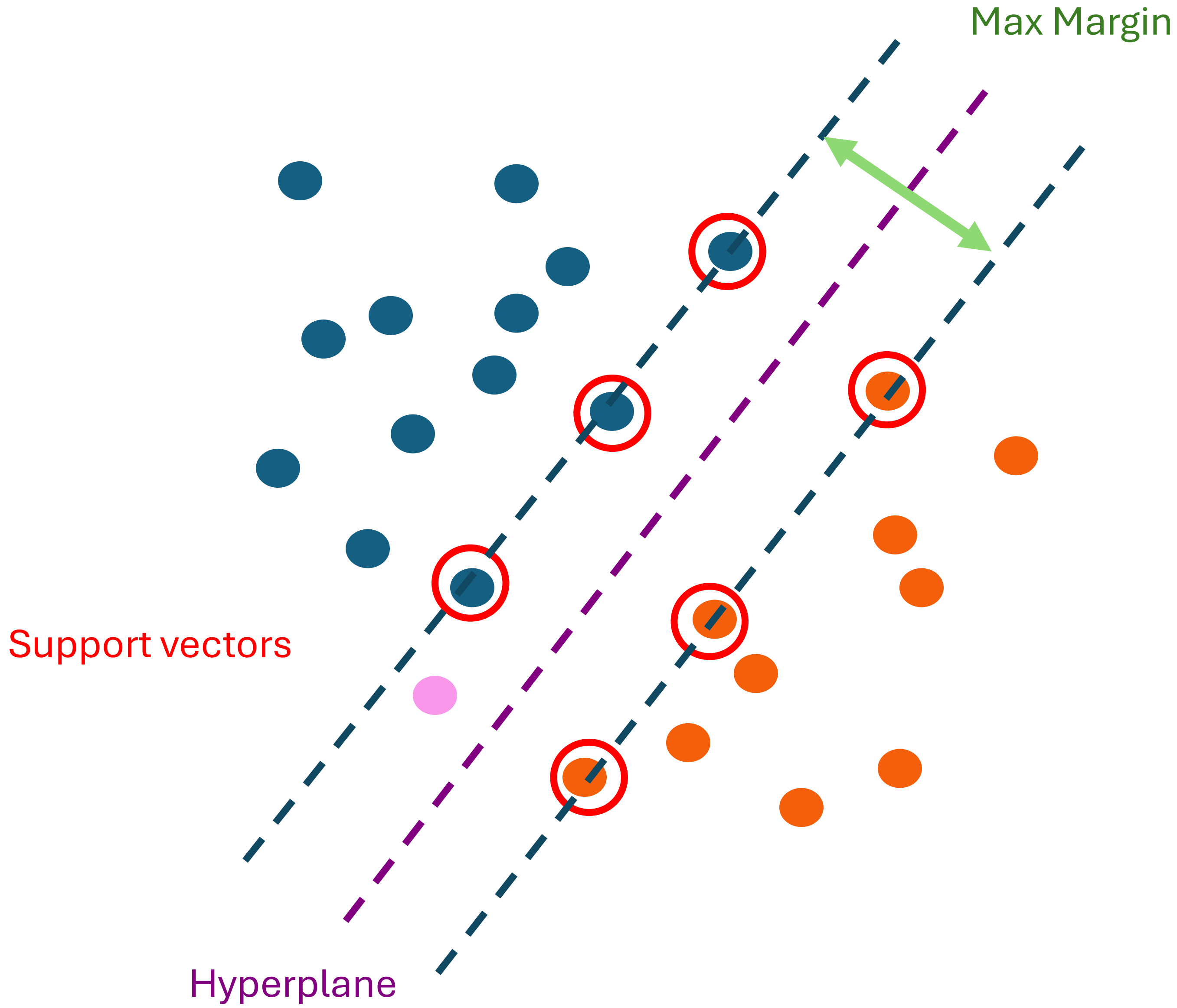
Given a set of labeled points

$$(\vec{x}_i, y_i)_{i=1\ldots N}$$

$$\vec{x}_i \in \mathbb{R}^n \qquad y_i \in \{+1, -1\}$$

We want to predict the label of an unseen point

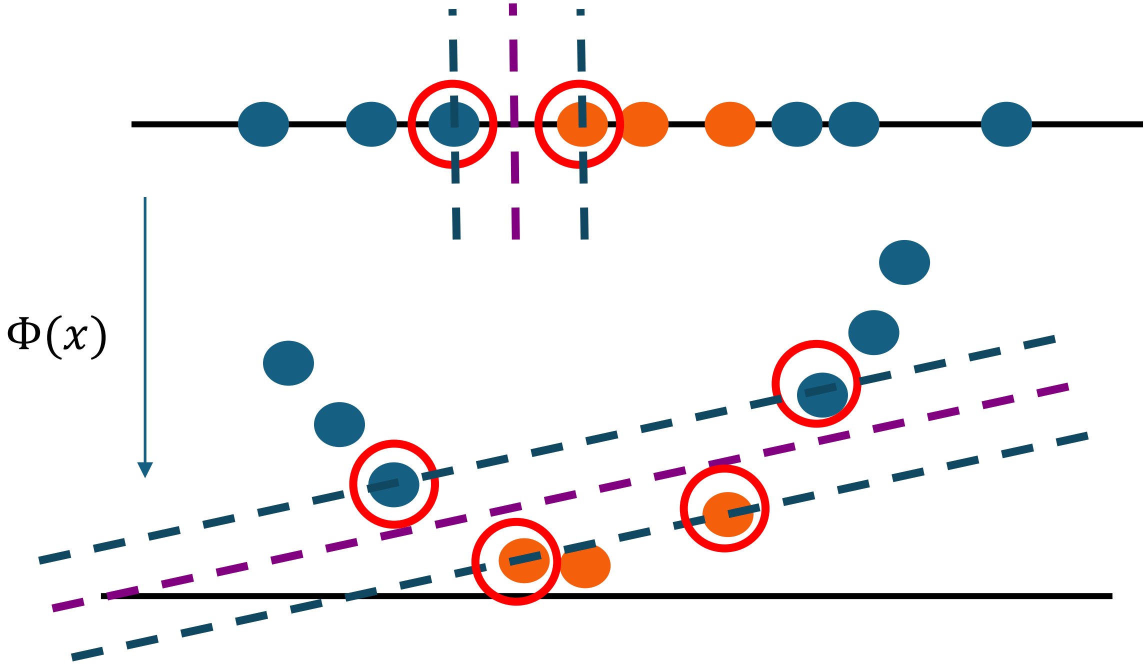
Find a hyperplane which separates our labeled data. Use this as our decision rule



Features map

- Linearly separable datasets the task can be solved in the data domain
- Non-linearly separable datasets may become linearly separable by including new features.

  This transformation is called a feature map Φ



Decision rule:

$w^T x + b = 0$ $\quad$ $label(x) = sign(w^T x + b)$

$w^T \Phi(x) + b = 0$ $\quad$ $label(x) = sign(w^T \Phi(x) + b)$

Solving for the optimal separating hyperplane

- Primal Problem

$$\min_{a,w,b} L_P = \frac{\|w\|^2}{2} - \sum_{i \epsilon T} a_i [y_i (w^T \Phi(x_i) + b) - 1]$$

- Dual form

$$\frac{\partial L_P}{\partial w} = 0 \qquad \frac{\partial L_P}{\partial b} = 0$$

$$\max_{a} L_D(a) = \sum_{i \epsilon T} a_i - \frac{1}{2} \sum_{i,j \epsilon T} a_i a_j y_i y_j \, \Phi(x_i)^T \Phi(x_j)$$

"kernel trick"

$$\Phi(x_i)^T \Phi(x_j) = K_{ij} = K(x_i, x_j)$$

Decision rule: $label(s) = sign(\sum_{i \epsilon N_s} a_i y_i K(x_i, s) + b)$

If K_{ij} is efficiently computable we can efficiently solve the dual form of our problem

Kernel function

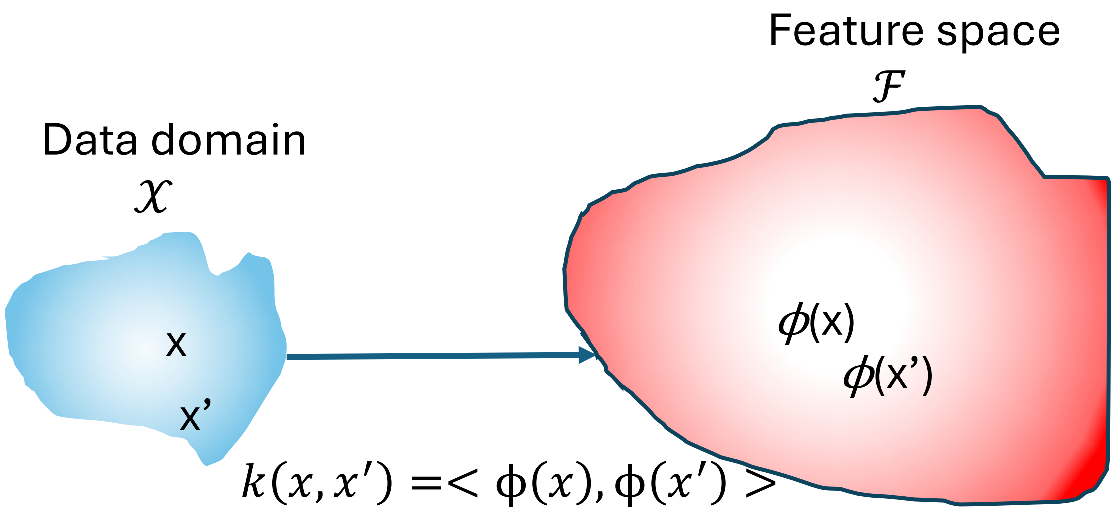
- We do not need to calculate the actual embedding
- Gram matrix (kernel matrix) define distance or similarity in the original space
- After embedding you can calculate the inner product in the embedding space, you just need to calculate the product but do not need to know the vector
- Application: k-means, SVM,…

Linear Model Definition

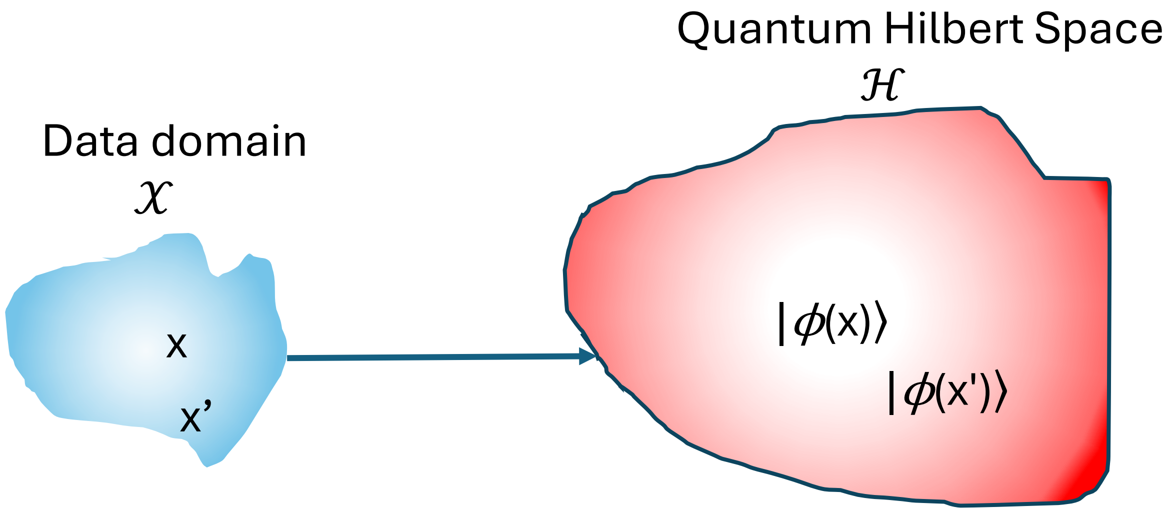
- Let $\mathcal{X}$ be a data domain
- And $\phi: \mathcal{X} \rightarrow \mathcal{F}$ a feature map
- We call a linear model in $\mathcal{F}$ any function $f(x) = \langle \phi(x), w \rangle_{\mathcal{F}}$
- With $w \in \mathcal{F}$

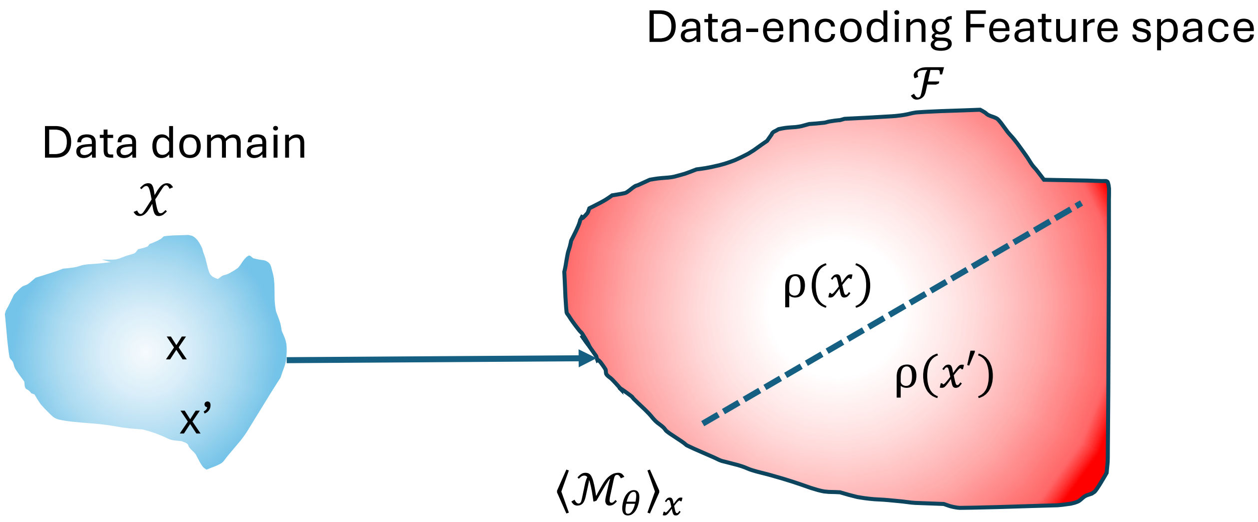
From this definition we immediately see that deterministic quantum models are linear models

Kernel Methods



Quantum models





Data-encoding feature map

1. Dirac vectors as feature vectors

$$\phi: x \rightarrow |\phi(x)\rangle \qquad \langle\phi(x)|\phi'(x)\rangle$$

2. Density matrices to represent feature encoding states

$$\phi: x \rightarrow \rho(x) \qquad tr\{\rho(x), \rho(x)\}$$

$$\rho(x) = |\phi(x)\rangle\langle\phi(x)|$$ If are pure states

Data-encoding feature map definition

- Given a n-qubit quantum system state $|\psi\rangle$, let $\mathcal{F}$ be the space of complex valued 2^n*2^n dimensional matrices equipped with the H-S inner product

$$\langle\rho, \sigma\rangle_{\mathcal{F}} = tr\{\rho^{\dagger}\sigma\}$$

- The data encoding feature map is defined as the transformation

$$\phi: \mathcal{X} \rightarrow \mathcal{F}$$
$$\phi(x) = |\phi(x)\rangle\langle\phi(x)| = \rho(x)$$

- And can be interpreted by a data encoding quantum circuit U(x)

Data-encoding feature map -> Quantum Kernel

- Theorem:

Let $\phi: \mathcal{X} \rightarrow \mathcal{F}$ be a data-encoding feature map over $\mathcal{X}$

The inner product of two feature vectors is a kernel

$$k(x, x') = tr[\rho(x')\rho(x)] = |\langle\phi(x')|\phi(x)\rangle|^2$$

**prove*

Starting point

- A large class of supervised, deterministic quantum models can be formulated as kernel methods
- Quantum Models are *linear models* in the feature space of the data encoding feature map
- (Valid for VQM and FT)
- This allows us to apply the results of kernel methods to QML
- Kernel calculated by a quantum computer -> QKE (Quantum Kernel Estimator)

Quantum Model definition

- Let $\rho(x)$ be a quantum state that encode classical data $x \in \mathcal{X}$
- $\mathcal{M}$ a Hermitian operator representing a quantum measurement
- A quantum model is

$$f(x) = tr\{\rho(x)\mathcal{M}\}$$

- the expectation value of the quantum measurement as a function of the data input
- The space of all quantum models contains functions $f: \mathcal{X} \to \mathbb{R}$
- For pure state embedding with $\rho(x) = |\phi(x)\rangle\langle\phi(x)|$

$$f(x) = \langle\phi(x)|\mathcal{M}|\phi(x)\rangle$$

Deterministic quantum models are linear models in data-encoding feature space

Theorem

- Let $f(x) = tr\{\rho\mathcal{M}\}$ be a deterministic quantum model
- with a feature map $\phi: x \in \mathcal{X} \to \rho(x) \in \mathcal{F}$
- Then f is a linear model in $\mathcal{F}$

- Note: the measurement $\mathcal{M}$ can always be expressed as a linear combination of data encoding states $\mathcal{M} = \sum_k \gamma_k \rho(x^k)$ where $x^k \in \mathcal{X}$

Quantum measurements are linear combinations of data-encoding states

Theorem

- Let $f_{\mathcal{M}}(x) = tr\{\rho\mathcal{M}\}$ be a quantum model
- There exist a measurement $\mathcal{M}_{\text{exp}} \in \mathcal{F}$ of the form

$$\mathcal{M}_{exp} = \sum_k \gamma_k \rho(x^k)\,, x^k \in \mathcal{X}$$

Such that $f_{\mathcal{M}}(x) = f_{\mathcal{M}_{exp}}(x)\ \ \forall x \in \mathcal{X}$

The RKHS of Quantum Kernels

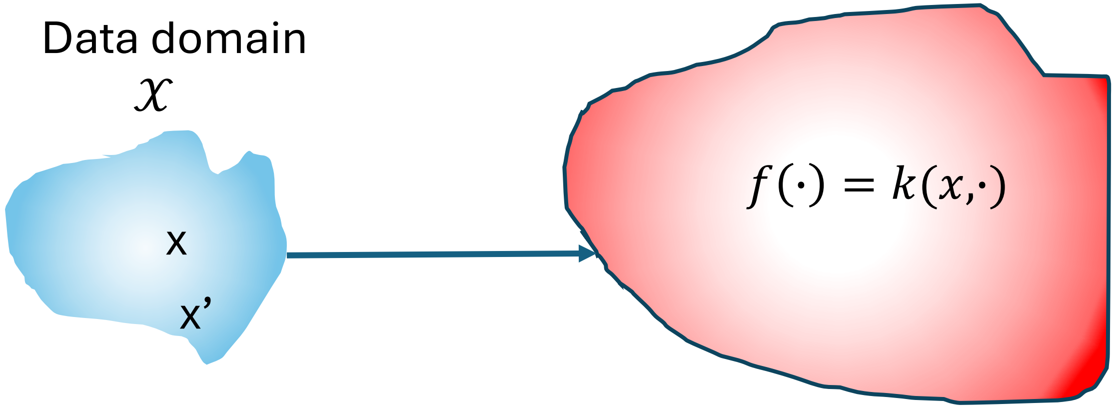
Reproducing Kernel Hilbert Space

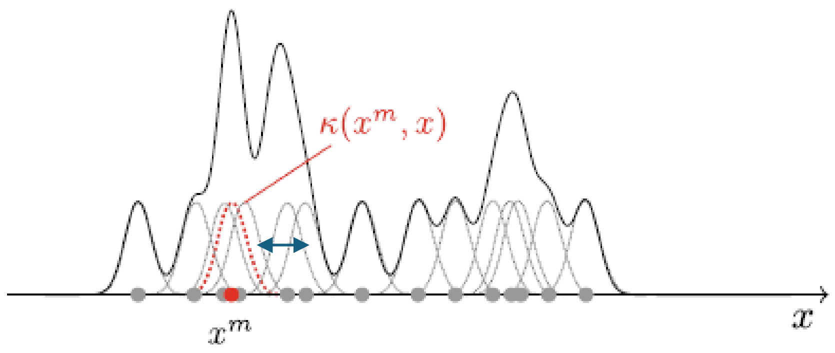
Alternative feature space

Derived directly form the Kernel

RKHS

Hilbert space of functions F





- Universality of QMs as function approximators
- Optimization

RKHS definition

- Let $\mathcal{X} \neq 0$
- The RKHS of k over $\mathcal{X}$ is the Hilbert space F created by completing the span of functions $f: \mathcal{X} \to \mathbb{R}$, $f(\cdot) = k(x,\cdot), x \in \mathcal{X}$
- For two functions

$$f(\cdot) = \sum_i \alpha_i k(x^i,\cdot) , g(\cdot) = \sum_j \beta_j k(x^j,\cdot) \in \mathrm{F}$$

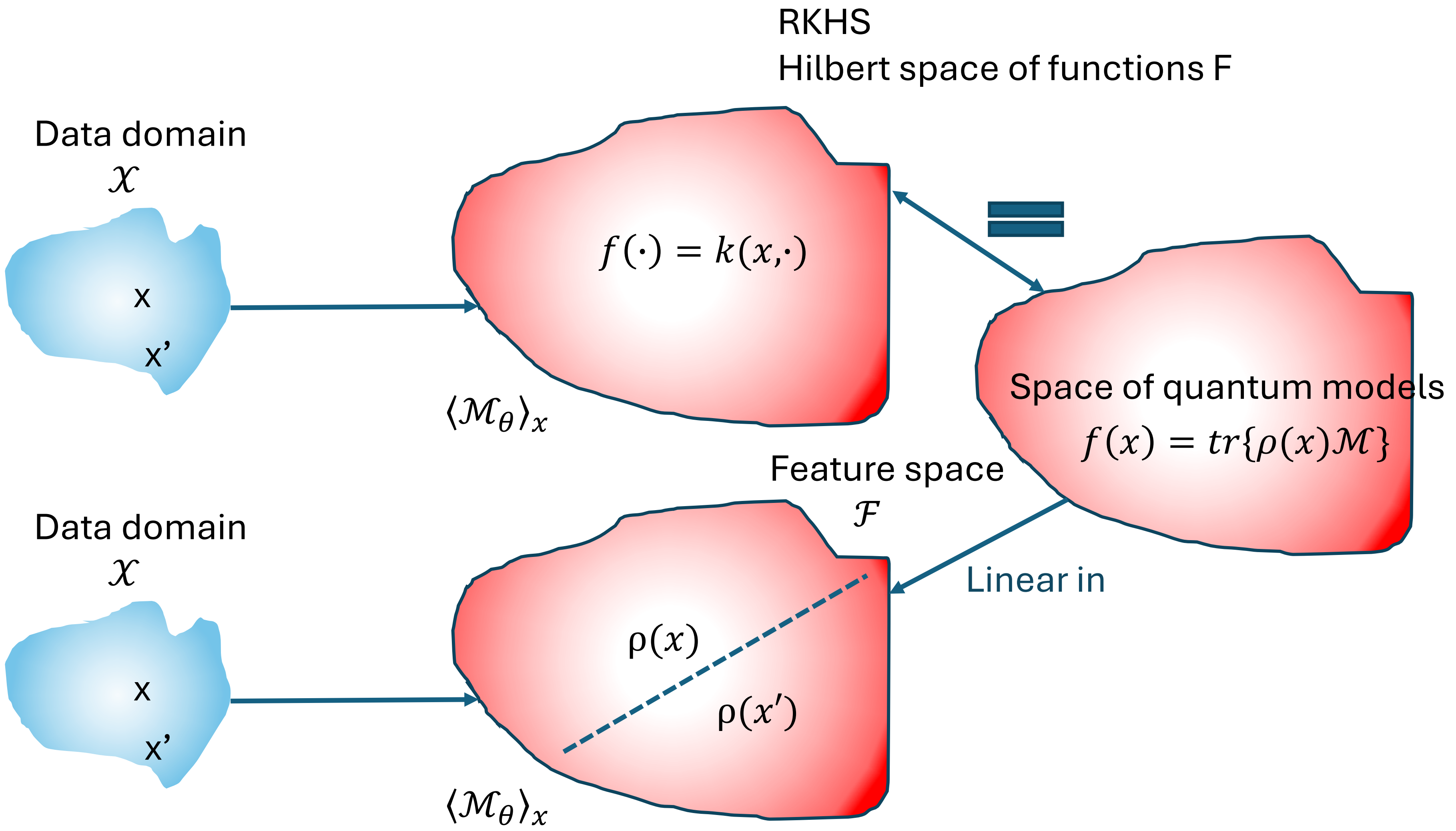
- The inner product is defined as

$$\langle f, g \rangle_F = \sum_{ij} \alpha_i \beta_j k(x^i, x^j)$$

- With $\alpha_i, \beta_j \in \mathbb{R}$

Functions in the RKHS of F of the quantum kernel are Linear models in the data-encoding feature space $\mathcal{F}$ (and vice-versa)

Theorem



Summary

- QMs are *linear models* in the *data-encoded "feature vectors"*
- QMs that minimize typical ML cost functions have measurement that can be written as *Kernel expansions in the data*
- The problem of finding the optimal measurement for typical ML cost functions trained with M data samples can be formulated as an M-dim optimization problem

Expressivity

Variational quantum circuits

- Which functions can these models learn for a given ansatz?

Ansatz

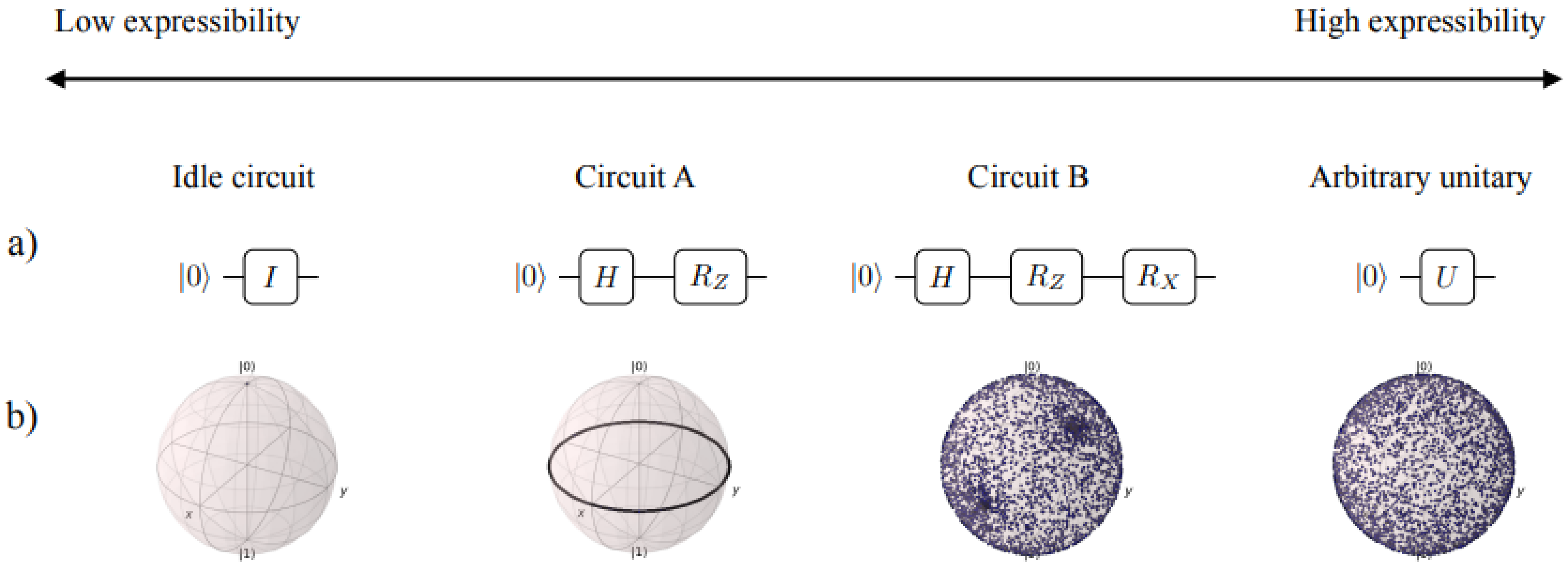
- Problem inspired ansatz
- Generic, problem agnostic
- Hardware efficient (reducing circuits depth)
- Variational Hamiltonian ansatz

Ansatz quality

Given the wide range of ansatz a question is weather a given architecture can prepare a target state by optimizing its parameters

- Expressivity
  - Expressive circuit can be used to uniformly explore the entire space of quantum states
  - How? Compare the distribution of states obtained from the circuit
- Entangling capacity
  - Average entanglement of states produced from randomly sampling the circuit parameters

Expressivity - Generalization



Sim et al. on [Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms]
Havlíček et al. on [Supervised learning with quantum-enhanced feature spaces]

Variational quantum circuits

- Which functions can these models learn for a given ansatz?

  Variational Quantum Classifier

  Quantum models can be expressed as a sum of trigonometric functions

$$f_{\theta}(\mathrm{x}) = \langle \psi(x, \theta) \mid \sigma_z \mid \psi(x, \theta) \rangle = \cos(\theta_2)\cos(x) - \sin(\theta_1)\sin(\theta_2)\sin(x)$$

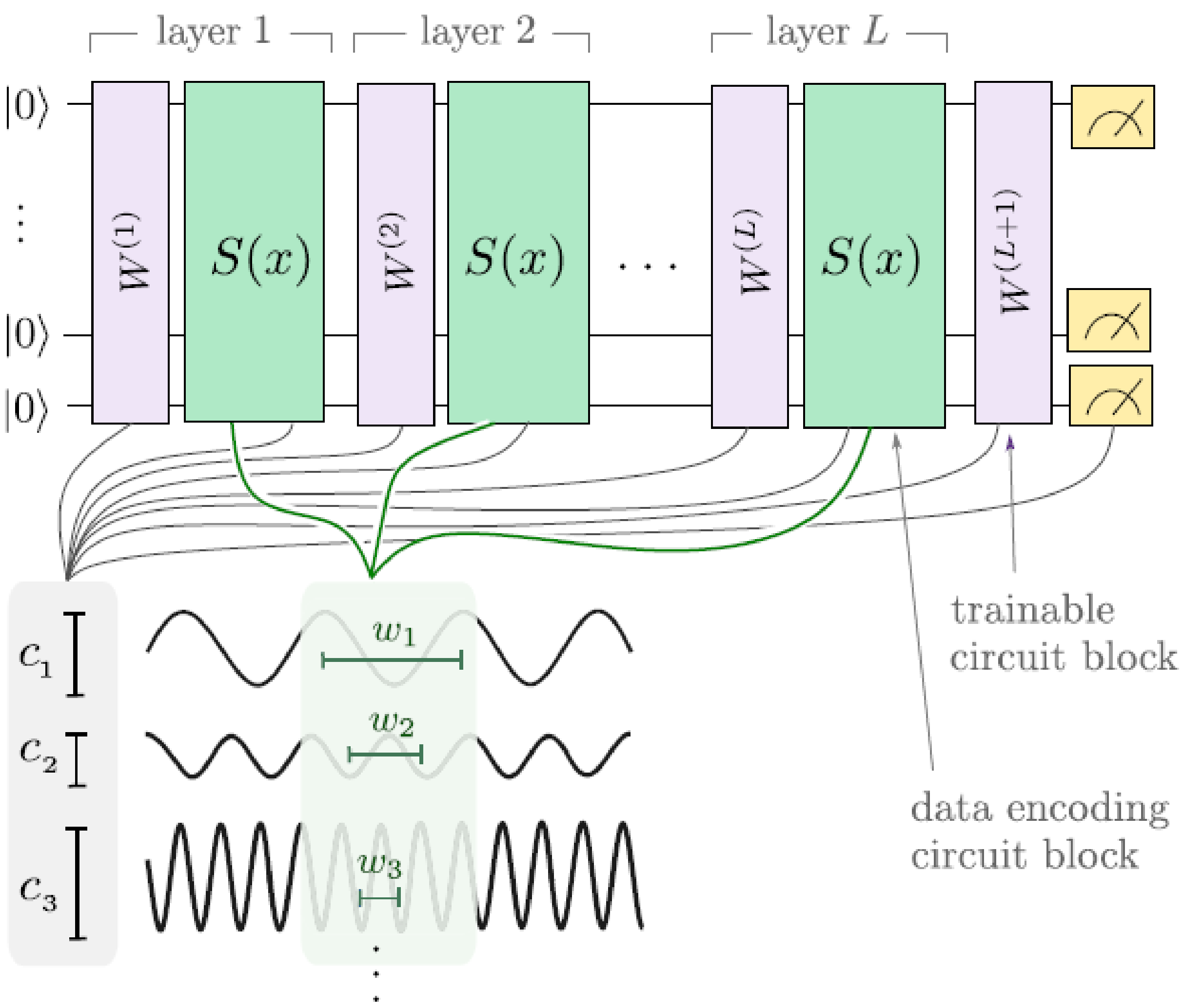
Effect of data encoding on the expressive power of variational quantum-machine-learning models

Maria Schuld,[1] Ryan Sweke,[2] and Johannes Jakob Meyer[2]
[1] Xanadu, Toronto, Ontario, Canada M5G 2C8
[2] Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany

(Received 22 September 2020; revised 11 February 2021; accepted 3 March 2021; published 24 March 2021)

$$f(x)_\theta = \sum_{\omega \in \Omega \subset \mathbb{R}^N} c_\omega(\theta) e^{i\omega x}$$

- Expressivity influenced by both frequency spectrum and trainable parameters.
- Data Encoding controls the Expressivity of quantum models



$$U(x,\theta) = W_{N+1}(\theta) \prod_{i=1}^{N} S_i(x_i) W_i(\theta_i)$$

$$S_i(x_i) = e^{-i x_i G_i}$$

$$f(x) = \langle 0 | U(x)^{\dagger} \mathcal{M} U(x) | 0 \rangle$$

$$f(x)_{\theta} = \sum_{\omega \in \Omega \subset \mathbb{R}^N} c_{\omega}(\theta) e^{i\omega x}$$

- If the encoding is not rich enough, we may end up with very limited model classes that variational circuits can express, and learn, even if the variational circuit is arbitrarily deep and wide
- Expressivity of the coefficients, controlled by the model

Function class of quantum model

Theorem

- $\mathcal{X} = \mathbb{R}^N$ *input domain*, $\mathcal{Y} = \mathbb{R}$ *output domain*
- $f_\theta: \mathcal{X} \rightarrow \mathcal{Y}$ deterministic quantum model
- Circuit $U(x, \theta) = W_{N+1}(\theta) \prod_{i=1}^{N} S_i(x_i) W_i(\theta_i)$ where $S_i(x_i) = e^{-i x_i G_i}$
- G_i is a diagonal operator diag($\lambda_1^i, \ldots, \lambda_d^i$), d is Hilbert space dimension

Then:

$$f(x)_\theta = \sum_{\omega \in \Omega \subset \mathbb{R}^N} c_\omega(\theta) e^{i\omega x}$$

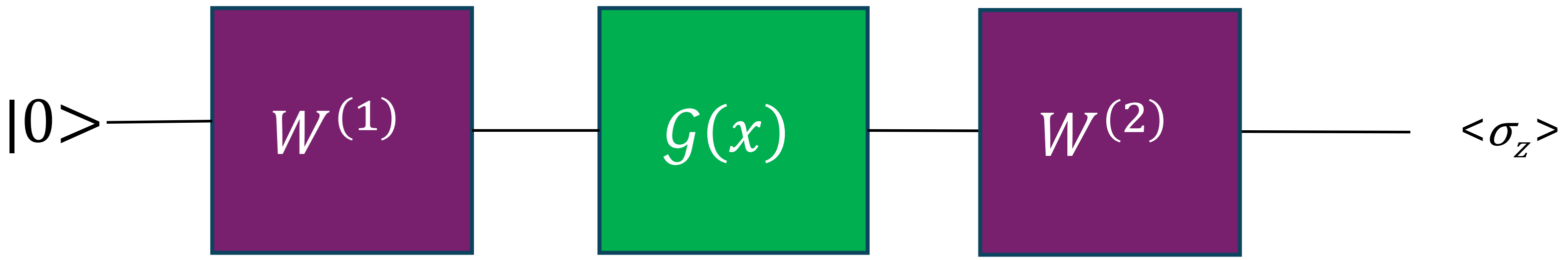
Real valued

- $0 \in \Omega$
- $\omega \in \Omega, -\omega \in \Omega \rightarrow c_\omega = c_{-\omega}^*$
- K=(| Ω |-1)/2 size of the spectrum

Example: single Pauli rotation encoding

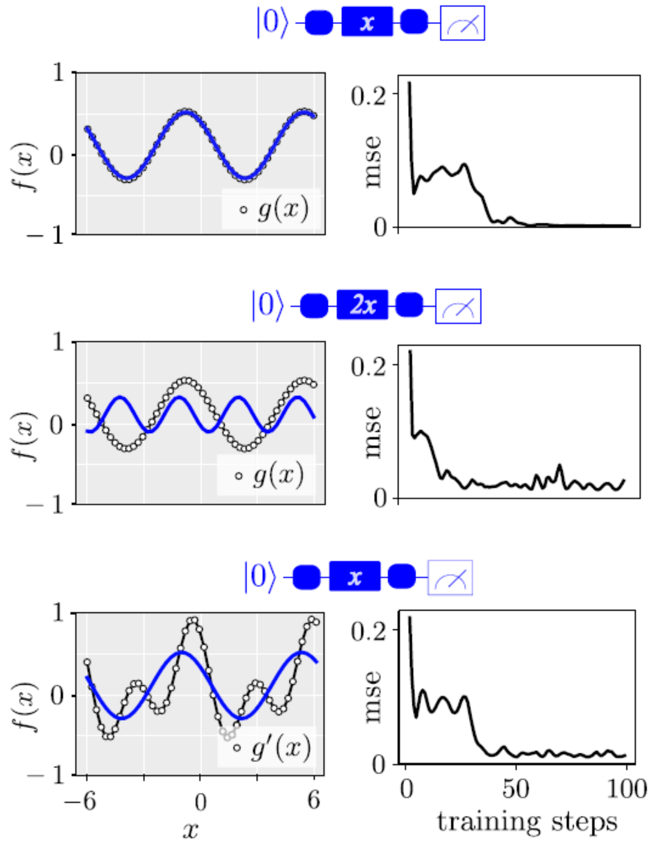
- L=1
- $\mathcal{G}(x) = e^{-ixH}$
- $U(x) = W^{(2)}\mathcal{G}(x)W^{(1)}$ $\rightarrow f(x) = A sin(2\gamma x + B) + C$
- H has two eigenvalues $(\lambda_1, \lambda_2) \rightarrow (-\gamma, \gamma)$
- $H = \frac{\sigma}{2}$ in Pauli basis $\rightarrow \gamma = \frac{1}{2}$

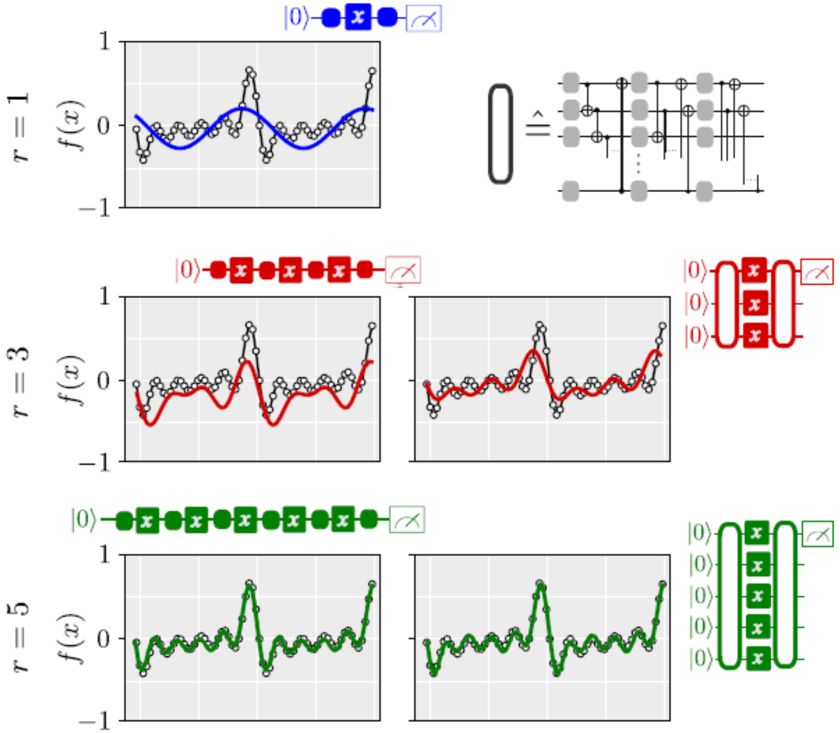
$\Omega = \{-2,0,2\}$

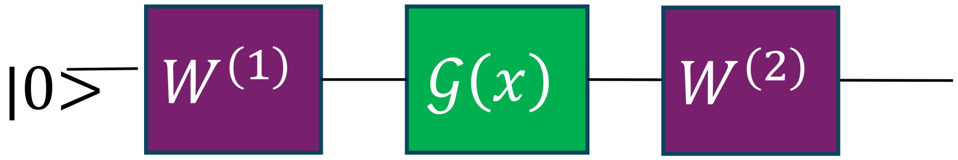


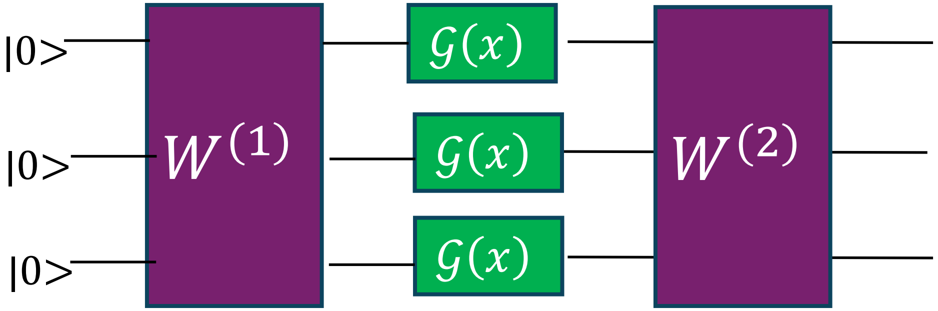
Numerical results

$$\Omega = \{-r, 1-r, \dots, 0, \dots r-1, r\}$$









Limits of expressivity

- Quantify the number of frequencies a model has acces to:
- $\Omega = \{\left(\lambda_{j_1} + \cdots + \lambda_{j_L}\right) - (\lambda_{k_1} + \cdots + \lambda_{k_L})\}$
- 2L terms
- d^{2L} total values of frequencies possible
- $K \leq \frac{d^{2L}}{2} - 1$

Conclusion

- VQC represented by truncated Fourier series
- Frequencies are determined by data-embedding
- Coefficient determined by unitaries and observable
- Replicating the embedding (parallel or serial) extends frequency spectrum linearly
- A single layer VQC with a sufficient large Hilbert space is a universal function approximator

Trainability

Variational quantum circuits

- How can we determine the optimal models that minimize the cost functions derived from learning problems?

Cost function -> Training

- should not be efficiently computable with a classical computer
- should be “operationally meaningful” (smaller cost values indicate a better solution quality)
- must be trainable (it should be possible to efficiently optimize the parameters)

Training Variational Quantum Models

- Goal: Find the parameters which minimize a data-dependent cost function $C(\theta)$
- Automatic differentiation (chain rule)

Automatic differentiation

- Programming paradigm in which for a programmatic implementation of a differentiable function f_θ, methods to compute partial derivatives of the form $\partial_\mu f_\theta$ are automatically provided.
- Neural network

$$\partial_\mu C(\theta) = \frac{\partial C}{\partial f_\theta} \frac{\partial f_\theta}{\partial \mu}$$

Classical Results of quantum computation

Parameter-shift rule

- Family of rules that express the partial derivative of a quantum expectation with respect to a gate parameter as a *linear combination* of the same expectation, but with the parameter "shifted"

Parameter-shift rule

- The quantum model only depends on a single parameter μ which only affect a single gate $\mathcal{G}(\mu)$
- Variational circuit of he model f_θ: V$\mathcal{G}(\mu)$W
- $|\psi\rangle = W|0\rangle$
- $\mathcal{B} = V^\dagger \mathcal{M} V$
- Deterministic quantum model:
- $f_\theta = \langle\psi|\mathcal{G}^\dagger(\mu)\mathcal{B}\mathcal{G}(\mu)|\psi\rangle, \mu \in \theta$
- By linearity of the expectation the partial derivative:
- $\partial_\mu f_\theta = \langle\psi|\mathcal{G}^\dagger\mathcal{B}(\partial_\mu\mathcal{G})|\psi\rangle + \langle\psi|(\partial_\mu\mathcal{G})^\dagger\mathcal{B}\mathcal{G}|\psi\rangle$

Parameter-shift rule

$\partial_\mu f_\theta$=$\langle\psi|\mathcal{G}^\dagger\mathcal{B}(\partial_\mu\mathcal{G})|\psi\rangle + \langle\psi|(\partial_\mu\mathcal{G})^\dagger\mathcal{B}\mathcal{G}|\psi\rangle$

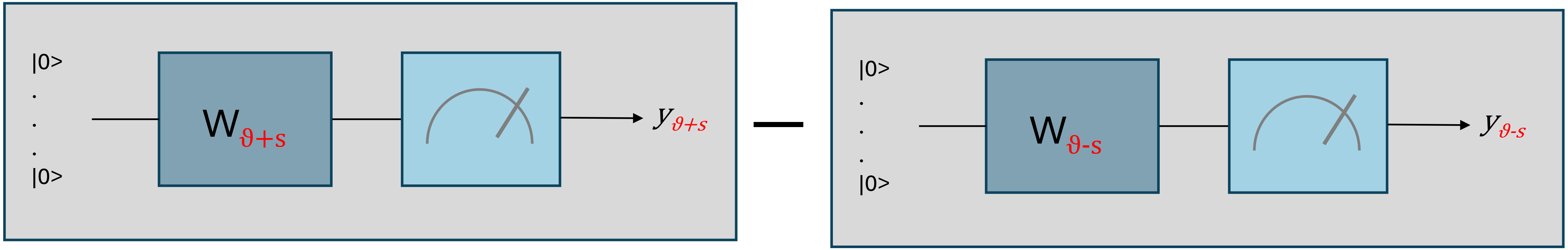
- Each term is not a quantum expectation value
- $\partial_\mu\mathcal{G}$ is unitary?
- How to compute the $\partial_\mu f_\theta$ using quantum computation?

Parameter shift rule

- Let f_μ = $<M>_\mu$ be a quantum expectation value that depends on a classical parameter μ.
- A parameter-shift rule is an identity of the form
- $\partial_\mu f_\mu = \sum_i a_i f_{\mu+s_i}$
- where $\{a_i\}$ and $\{s_i\}$ are real scalar values.

Parameter shift rule

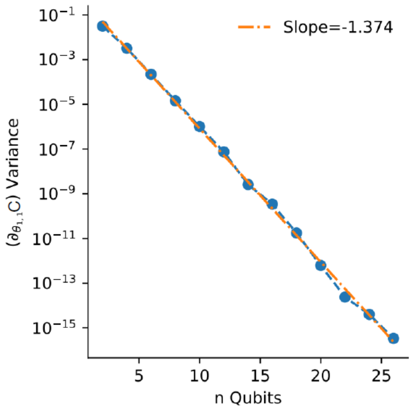
- $\nabla_j f_\mu = \frac{f(\mu+s)-f(\mu-s)}{2 sin s}$
- Estimation of analytical gradient (unbiased)
- Finite-difference rule: estimation of approximate gradient (biased)



Challenges in training VQA for ML

- Inherently stochastic environment due to finite budget for measurement
- Hardware noise
- Barren plateaus

$$Var[\partial_\vartheta f] = \langle(\partial_\vartheta f)^2\rangle_W - \langle\partial_\vartheta f\rangle_w{}^2$$



- The variance of the gradient of the loss function vanishes
- Gradients become concentrated around zero

Training Quantum models

- Find the optimal measurements of quantum models for typical machine learning cost functions only have relatively few degrees of freedom.
- The process of finding these optimal models (i.e., training over the space of all possible quantum models) can be formulated as a *low-dimensional optimization problem*.
- *Kernel-based approach*

Training

From a learning theory perspective, training can be phrased as
Regularized empirical risk minimization problem

Regularized empirical risk minimization of quantum models

- Let $\mathcal{X}, \mathcal{Y}$ be data input and output domains
- *p* a probability distribution (unknown) on $\mathcal{X}$ from which data is drawn
- $L: \mathcal{X} x \mathcal{Y} x \mathbb{R} \rightarrow [0, \infty]$ a loss function that quantifies the quality of the prediction of a quantum model
- $f(x) = \ tr\ [\rho(x)\mathcal{M}]$
- $R_L(f) = \int L(x, y, f(x) dp(x, y)$ expected loss
- $\hat{R}_L(f) = \frac{1}{M} \sum_{m=1}^{M} L(x^m, y, f(x^m))$
- $inf_{\mathcal{M} \epsilon \mathcal{F}} \lambda \|\mathcal{M}\|_{\mathcal{F}}^2 + \hat{R}_L(tr\{\rho(x)\mathcal{M}\}), \lambda \epsilon \mathbb{R}^+$

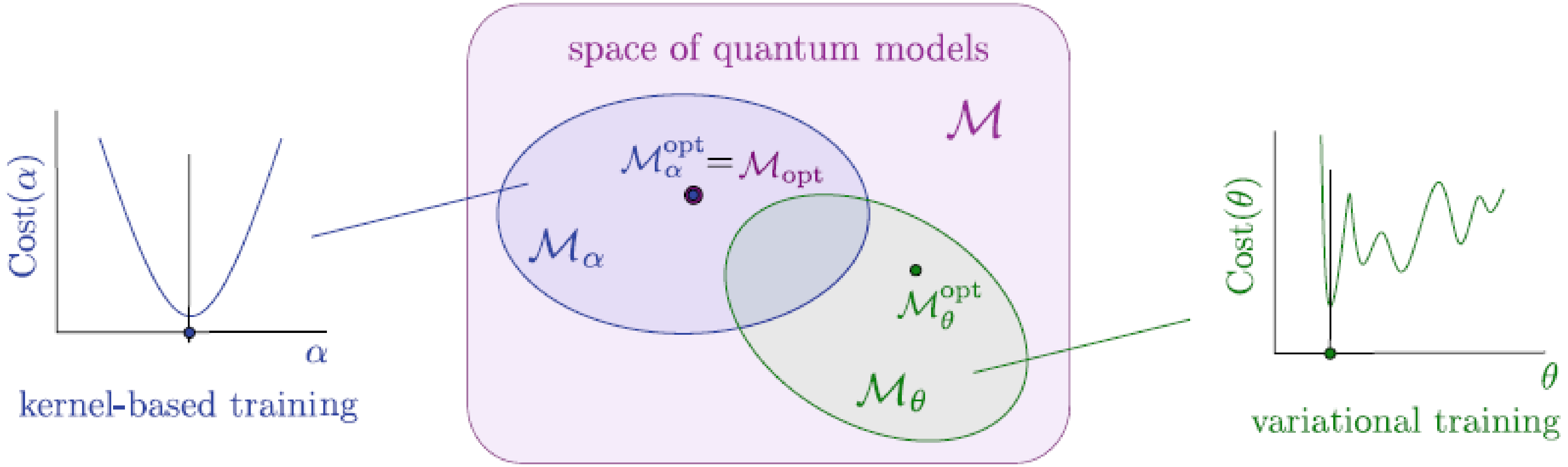
The Representer Theorem

- $\kappa: \mathcal{X}x\mathcal{X} \rightarrow \mathbb{R}$, F RKHS
- $\mathcal{D} = \{(x^1, y^1), \ldots, (x^M, y^M)\} \in \mathcal{X}x\mathcal{Y}$
- $g: [0, \infty) \rightarrow \mathbb{R}$ strictly monotonic increasing regularization function
- $L: \mathcal{X}x\mathcal{Y}x\mathbb{R} \rightarrow [0, \infty]$
- Any minimizer of the regularized empirical risk
- $f_{opt} = argmin_{f \epsilon \mathcal{F}}\{g\|f\|_{\mathcal{F}}^2 + \hat{R}_L(f)\}$
- Admit a representation of the form
- $f_{opt}(x) = \sum_{m=1}^{M} \alpha_m \kappa(x^m, x)$

Optimal measurement: $\mathcal{M}_{opt} = \sum \alpha_m \rho(x^m), x^m \in \mathcal{X}$

$$
\begin{aligned}
f_{\mathrm{opt}}(x) &= \sum_{m=1}^{M} \alpha_m \operatorname{tr}\{\rho(x)\rho(x^m)\} \\
&= \operatorname{tr}\{\rho(x) \sum_{m=1}^{M} \alpha_m \rho(x^m)\} \\
&= \operatorname{tr}\{\rho(x)\mathcal{M}_{\mathrm{opt}}\}
\end{aligned}
$$

Kernel-based training versus variational training



Training quantum models can be formulated as a finite-dimensional convex program

Quantum Machine Learning models

Deterministic

Probabilistic

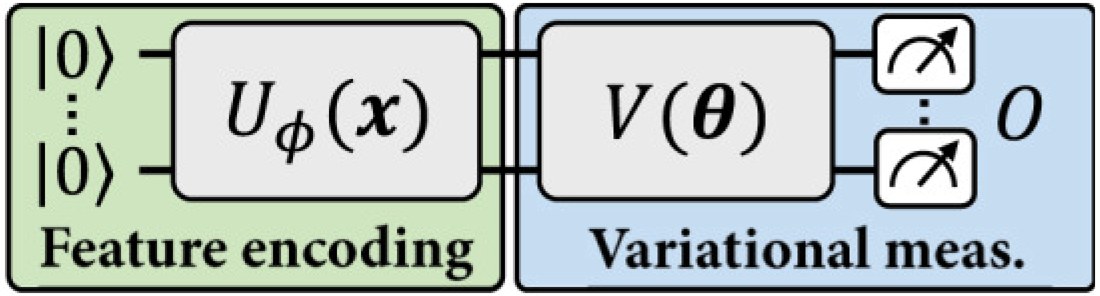
Implicit

Explicit

Data re-uploading

Quantum Machine Learning models

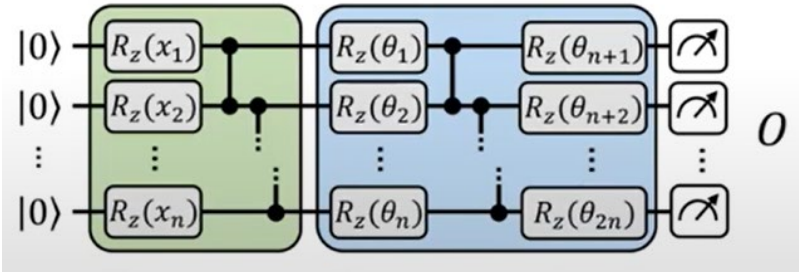
Explicit



$$
\begin{aligned}
f_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \langle \psi(\boldsymbol{x}) | V^\dagger(\boldsymbol{\theta}) O V(\boldsymbol{\theta}) | \psi(\boldsymbol{x}) \rangle \\
&= \mathrm{Tr}[\rho(\boldsymbol{x}) O_{\boldsymbol{\theta}}] \\
&= \langle \phi(\boldsymbol{x}), \boldsymbol{w}_{\boldsymbol{\theta}} \rangle_{\mathcal{H}}
\end{aligned}
$$

$$
\phi(\boldsymbol{x}) = \rho(\boldsymbol{x}) = |\psi(\boldsymbol{x})\rangle\langle\psi(\boldsymbol{x})|
$$

$$
w_{\boldsymbol{\theta}} = O_{\boldsymbol{\theta}} = V^\dagger(\boldsymbol{\theta}) O V(\boldsymbol{\theta})
$$

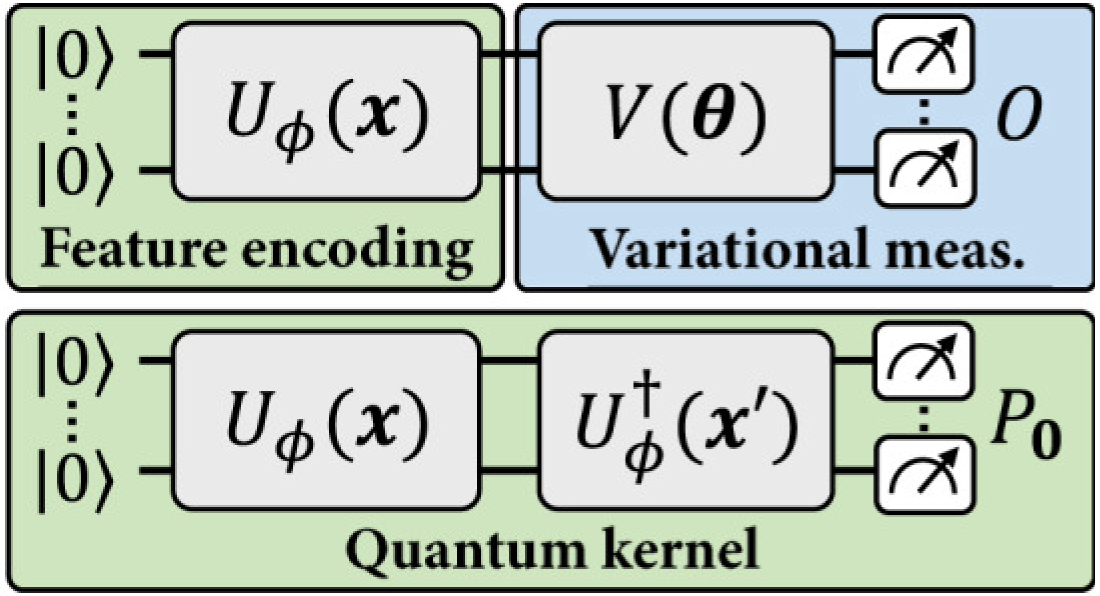


Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods.** Nat. Comm. (2023)

Quantum Machine Learning models

Explicit

Implicit



$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{m=1}^{M} \alpha_m \underbrace{\left|\langle\psi(\boldsymbol{x})|\psi(\boldsymbol{x}^{(m)})\rangle\right|^2}_{k(\boldsymbol{x},\boldsymbol{x}^{(m)})} = \sum_{m=1}^{M} \alpha_m \overbrace{\mathrm{Tr}\left[\rho(\boldsymbol{x})\rho(\boldsymbol{x}^{(m)})\right]}$$

$$O_{\boldsymbol{\alpha},\mathcal{D}} = \sum_{m=1}^{M} \alpha_m \rho(\boldsymbol{x}^{(m)})$$

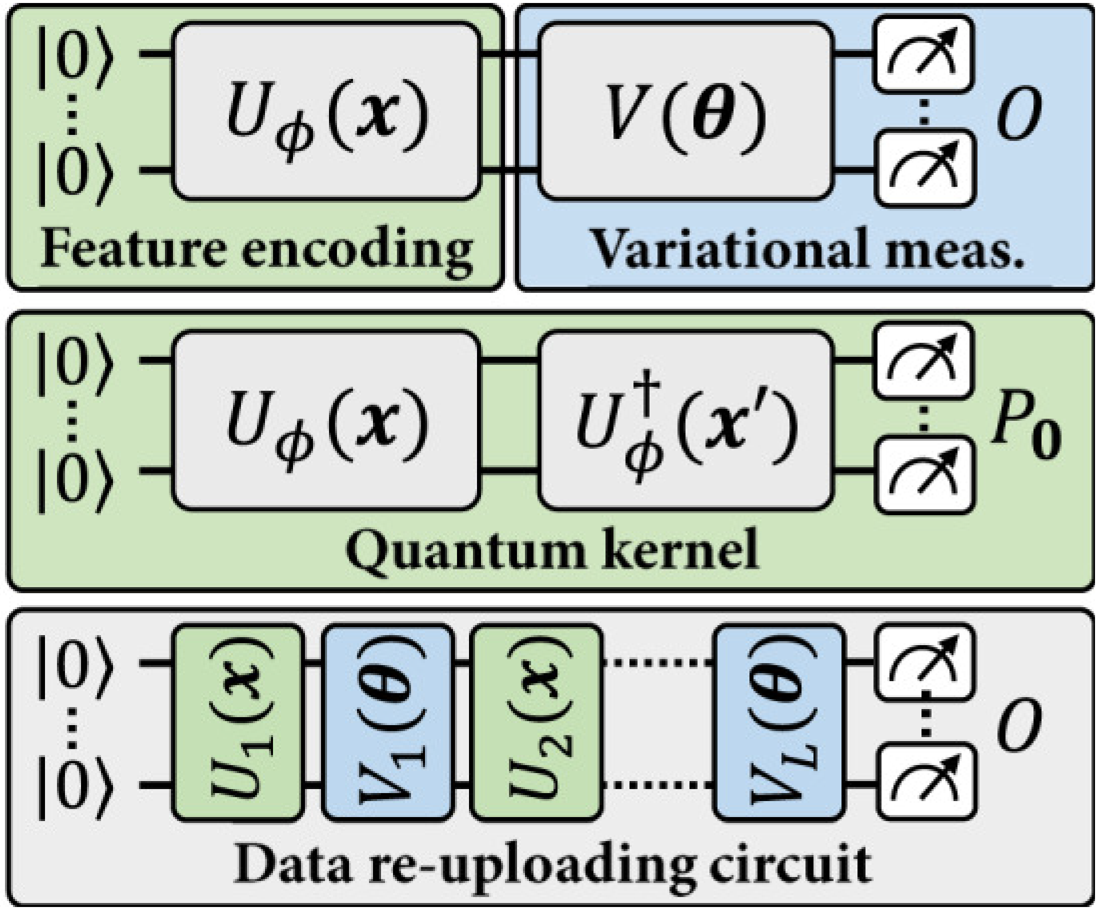
Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods.** Nat. Comm. (2023)

Quantum Machine Learning models

Explicit

Implicit

Data re-uploading



$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathrm{Tr}[\rho(\boldsymbol{x})O(\boldsymbol{\theta})]$$

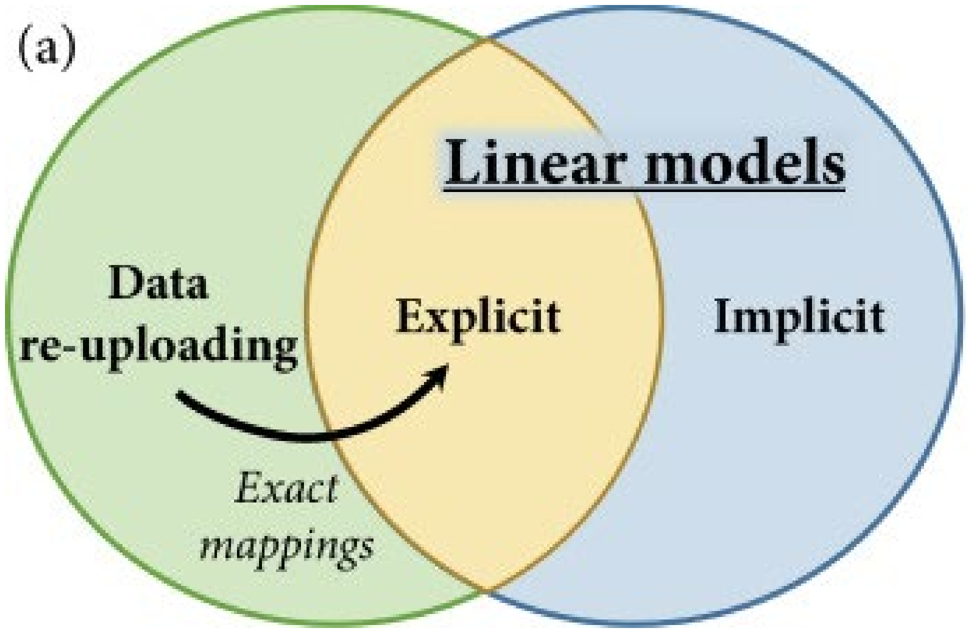
$$\mathrm{Tr}[\rho(\boldsymbol{x})O_{\boldsymbol{\alpha}}]$$

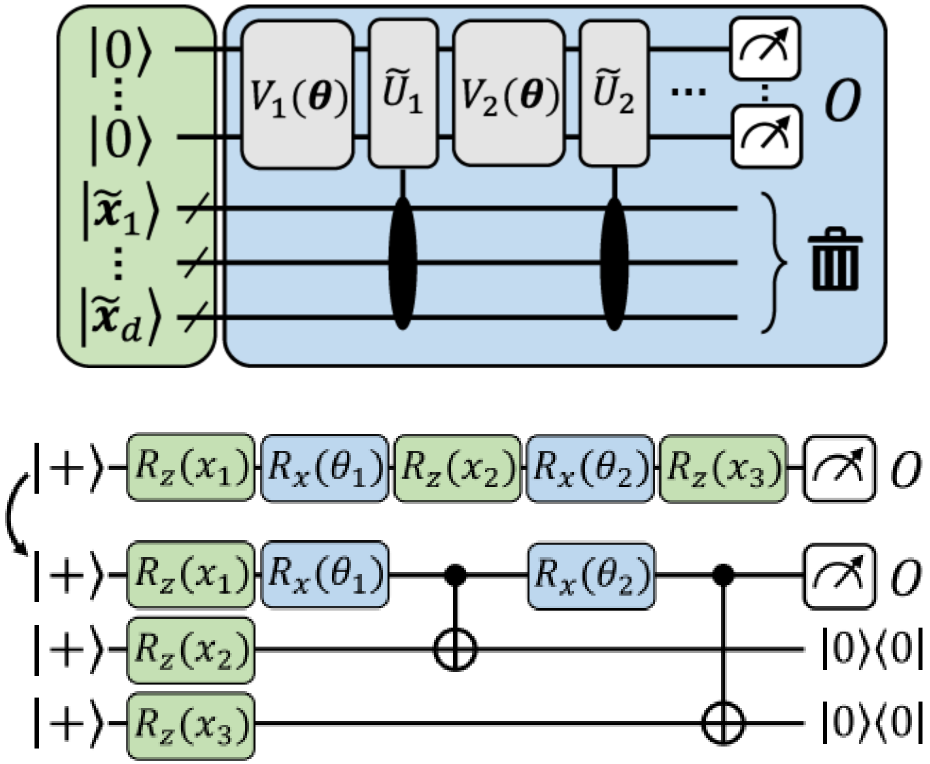
$$\textstyle\sum_m \alpha_m \rho(\boldsymbol{x}^{(m)})$$

$$\mathrm{Tr}[\rho(\boldsymbol{x},\boldsymbol{\theta})O(\boldsymbol{\theta})]$$

Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods.** Nat. Comm. (2023)

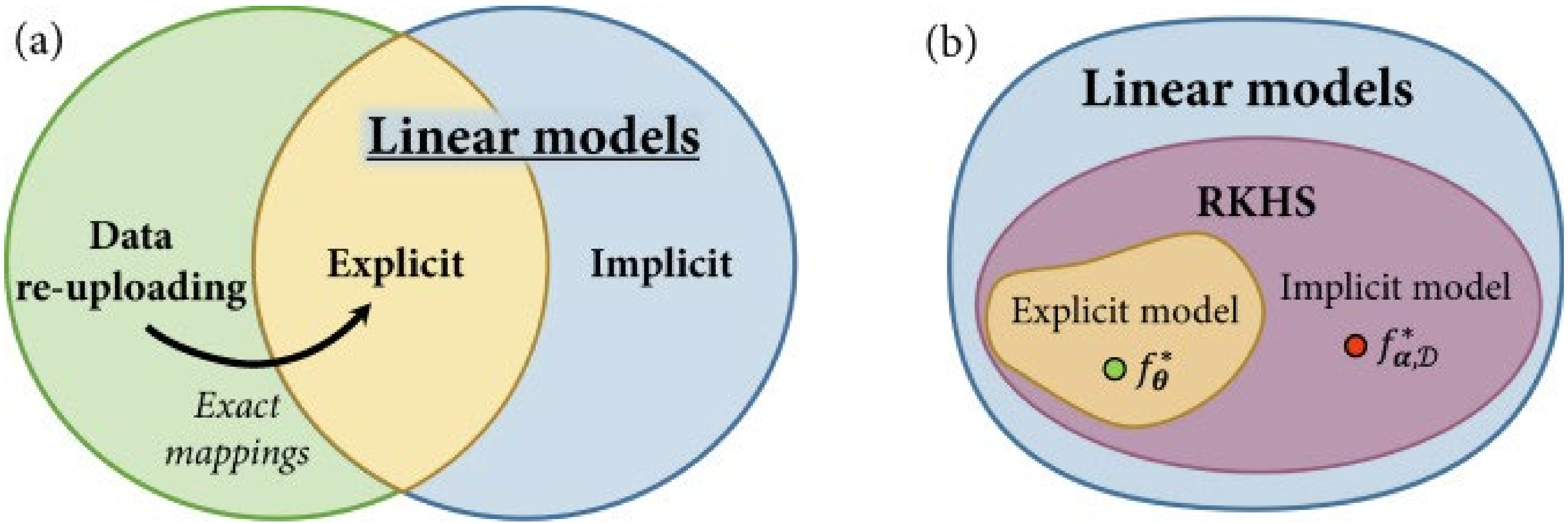
Quantum Machine Learning models





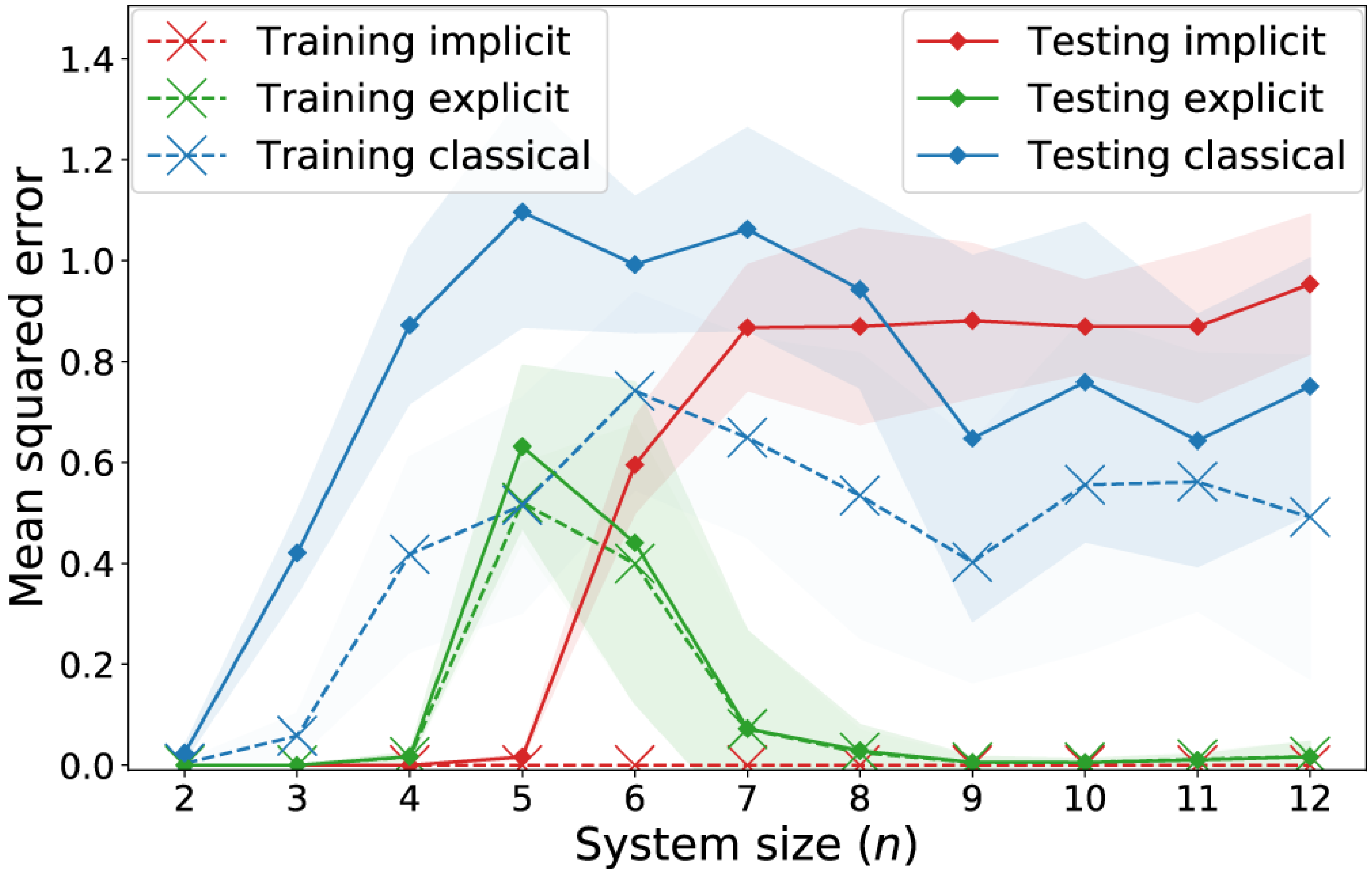
Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods.** Nat. Comm. (2023)

Quantum Machine Learning models



Jerbi, Fiderer, Poulsen Nautrup, Kübler, Briegel & Dunjko, **Quantum machine learning beyond kernel methods.** Nat. Comm. (2023)

Performance evaluation



Software

Qiskit

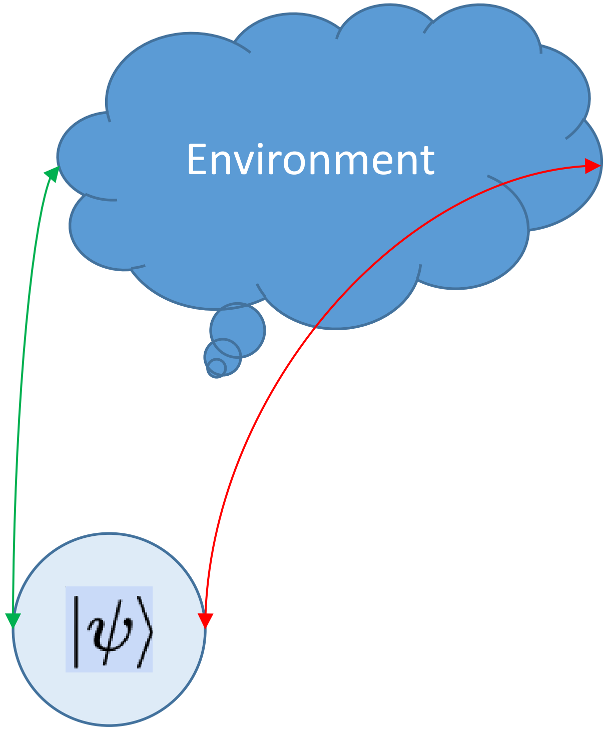
XANADU

PENNYLANE

Cirq

PyTorch

Quantum Noise



Dissipative Quantum Machine Learning

Benefits of Open Quantum Systems for Quantum Machine Learning

*María Laura Olivera-Atencio Lucas Lamata Jesús Casado-Pascual**

María Laura Olivera-Atencio
Física Teórica, Universidad de Sevilla, Apartado de Correos 1065, Sevilla 41080, Spain
Lucas Lamata
Departamento de Física Atómica, Molecular y Nuclear, Universidad de Sevilla, 41080 Sevilla, Spain
Instituto Carlos I de Física Teórica y Computacional, 18071 Granada, Spain
Jesús Casado-Pascual
Física Teórica, Universidad de Sevilla, Apartado de Correos 1065, Sevilla 41080, Spain

Keywords: *Quantum Machine Learning, Open Quantum Systems, Noise, Dissipation.*

Quantum machine learning is a discipline that holds the promise of revolutionizing data processing and problem-solving. However, dissipation and noise arising from the coupling with the environment are commonly perceived as major obstacles to its practical ex-

Practical: Classification

What to do:

Forms 4 groups, each need to present a small pitch on sunday

Challenge, improve, modify of:

Quantum Information Processing (2024) 23:339
https://doi.org/10.1007/s11128-024-04526-3

Quantum classifier based on open quantum systems with amplitude information loading

Eduardo Barreto Brito[1] · Fernando M. de Paula Neto[1] · Nadja Kolb Bernardes[2]

Received: 10 February 2024 / Accepted: 29 August 2024 / Published online: 8 October 2024

Questions?
Email me ☺
Thanks!