Secrets revealed in this session:

To explore the principles of quantum machine learning models, their parameterisation and optimisation

QML workshop
QML team
QML and aims
Parameterised circuits
Data encoding
- *Angle encoding*
- *The good, the bad and the ugly*

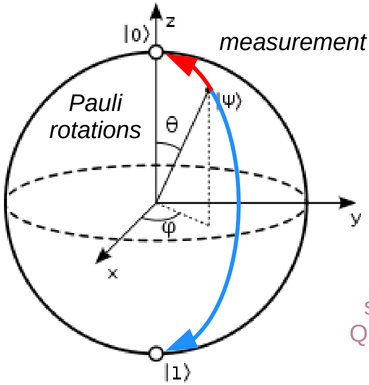
State measurement
Quantum model training
Parameters optimisation
Model geometry and gradients
QML readings
PennyLane demo
Summary

Quantum Machine Learning

Introduction

Jacob L. Cybulski

Enquanted, Melbourne, Australia



We will assume some knowledge of Quantum Computing ML and Python

Our QML Team

Sebastian Zając

Assistant Professor
SGH Warsaw School of Economics
LinkedIn

Dr Paweł Gora

Coordinator of QPoland *and*
CEO of Fundacja Quantum AI
LinkedIn

Jacob Cybulski
Founder, Researcher, Consultant
at Enquanted

and
Honorary A/Prof
In Quantum Computing
Deakin University
LinkedIn

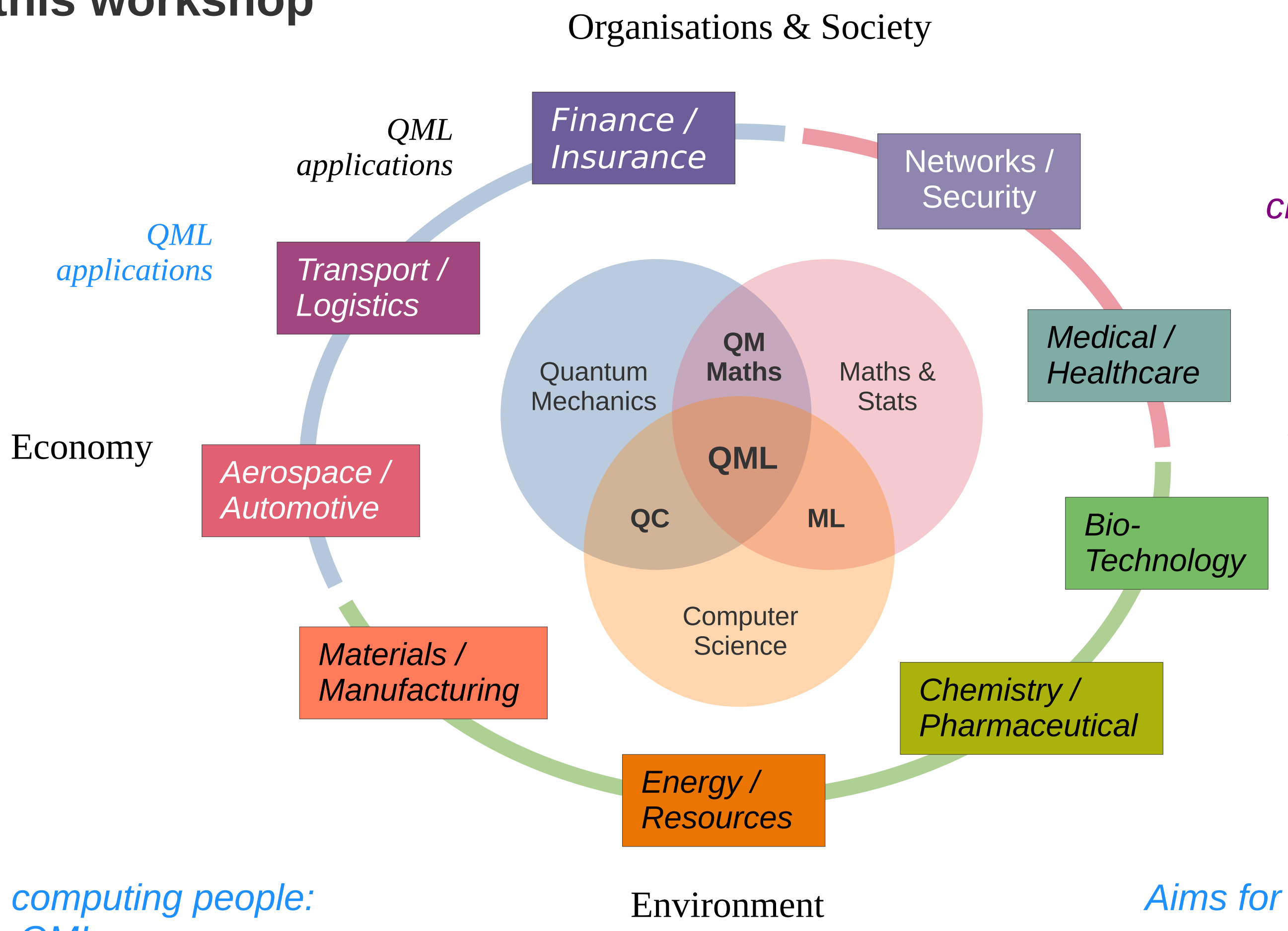
Tomek Rybotycki

QML Researcher
SRI PAS, NCAC PAS, CEAI AGH, KPLabs
LinkedIn

Quantum ML

aims of this workshop

this workshop aims at developing the understanding of and practical skills in the creation and application of QML models



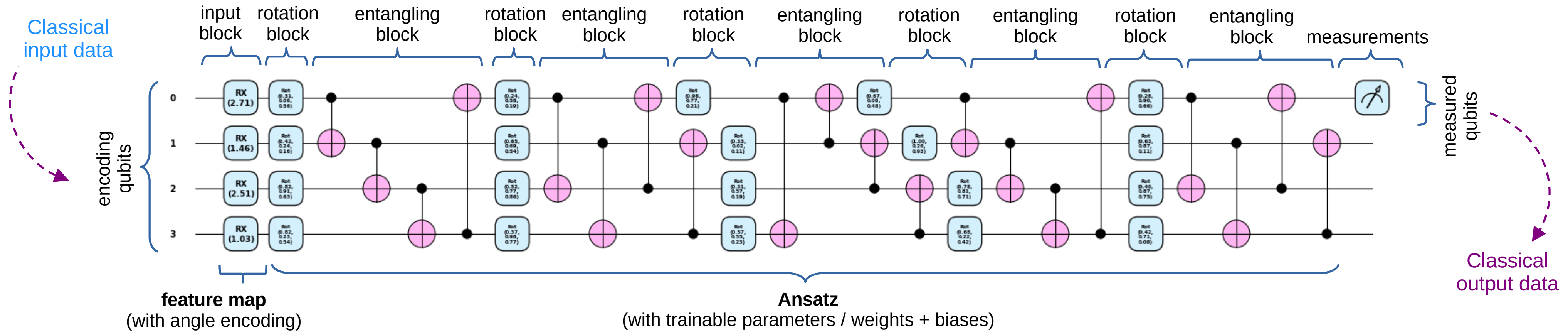
Aims for quantum computing people:
Learn about ML in QML

Aims for machine learning people:
Learn about Q in QML

Variational Quantum Models

= *Parameterised Quantum Circuits*

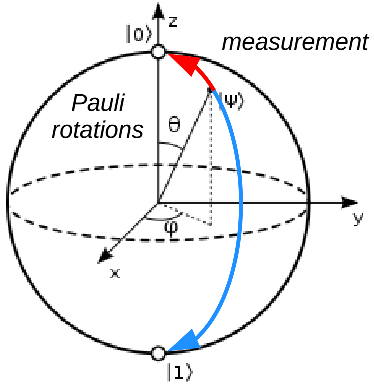
Variational quantum circuits are not executable!
They must first be instantiated, i.e. all of their input and weight parameters must be assigned values!
Ansatz parameters are trainable.



We can create a "variational" model = a circuit template with parameterised gates, e.g. P(a), Ry(a) or Rz(a), each allowing rotation of a qubit state in x, y or z axis (as per Bloch sphere).

Typically, but now always, the circuit consists of three blocks:

- a feature map (input)
- an ansatz (processing)
- measurements (output)



Classical input data is encoded into the feature map's parameters, setting the model's initial quantum state.

The quantum state is then altered by an ansatz, which consists of parameterised gates (operations), which alter the circuit state. Ansatz parameters are trainable. Qubits and parameters increase the model dimensionality.

The quantum state of the circuit is then measured and interpreted as the model's output in classical data form, e.g. as binary values, integer or real value, a single event's probability or the probability distribution.

Data encoding strategies

Data encoding

There are many methods of data embedding, such as: the *basis*, *angle*, *amplitude*, *QRAM*, ... encoding,

In this workshop we will rely on *angle encoding* realised as qubit state rotation by the angle defined by the data.

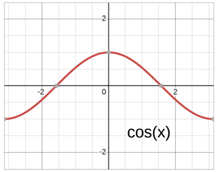
The rotation operators are always available in a quantum platform API (e.g. *Rx*, *Ry*, *Rz* or *Rxyz*).

Typically, the encoding rotation is performed around x or y axis, or both (allowing two values per qubit).

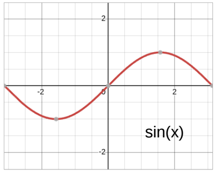
Rotations are *relative to a specific qubit state*, commonly starting at |0⟩ state, or (|0⟩+|1⟩)/sqrt(2), which require qubits to be initialised in these states.

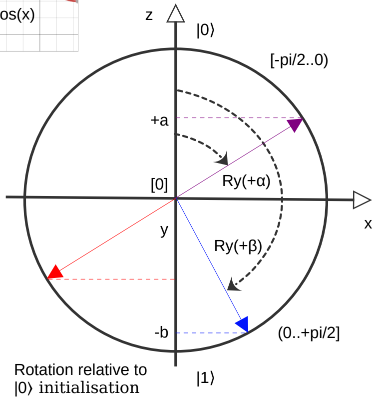
The encoded value could be represented either by the *angular rotation*, or the *amplitude* of the qubit projective measurement (Z).

In some cases, input data is repeatedly encoded and interspersed with ansatz layers, called *data reuploading*, which improves the model performance.

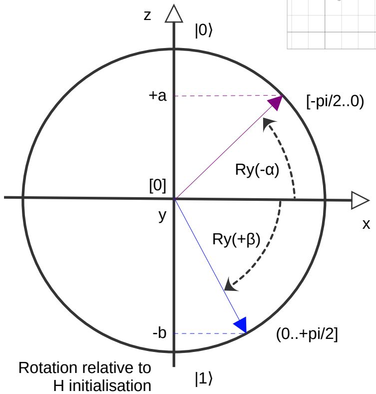


Note that training will place qubit states in areas x < 0 and arbitrarily around the z axis. Measurements of such states cannot distinguish them from "pure" x > 0 and z = 0.

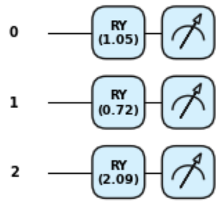


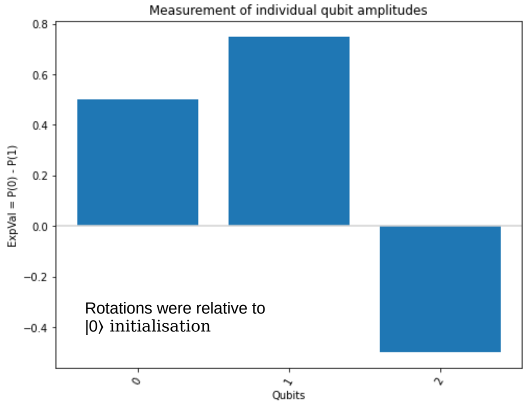


Rotation relative to
|0⟩ initialisation



Rotation relative to
H initialisation





Input

Values entered:	[np.arccos(0.5), np.arccos(0.75), np.pi-np.arccos(0.5)]
Ry angles used:	[1.047, 0.723, 2.094]

Measurements

Probabilities:	[[0.25, 0.75], [0.562, 0.438], [0.25, 0.75]]
Amplitudes:	[0.5, 0.75, -0.5]

Maria Schuld and Francesco Petruccione.
Machine Learning with Quantum Computers. 2nd ed. Springer, 2021.

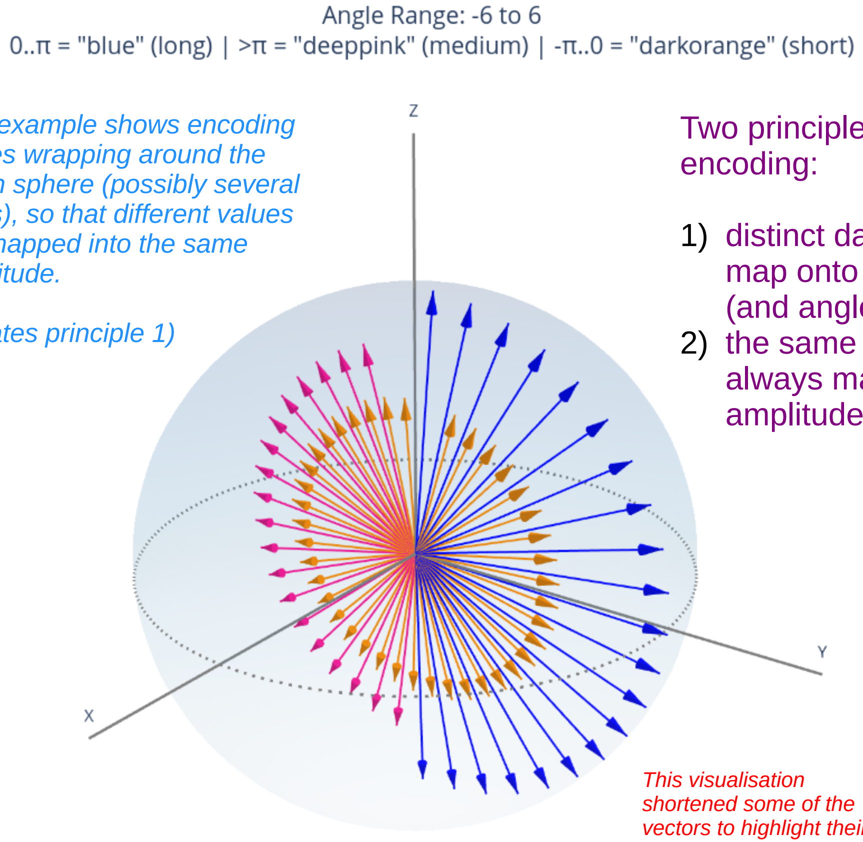
Angle encoding

The Good, the Bad and the Ugly

Special care must be taken in those cases where the values form some kind of symmetry or repetition. For example in a case where we were to encode angular values in the range of -2pi..2pi. In such a case values x and 2pi+x effectively represent the same angular position and their encoding should also be identical.

This example shows encoding values wrapping around the Bloch sphere (possibly several times), so that different values are mapped into the same amplitude.

(violates principle 1)



This visualisation shortened some of the vectors to highlight their difference, however, they are all of length 1.

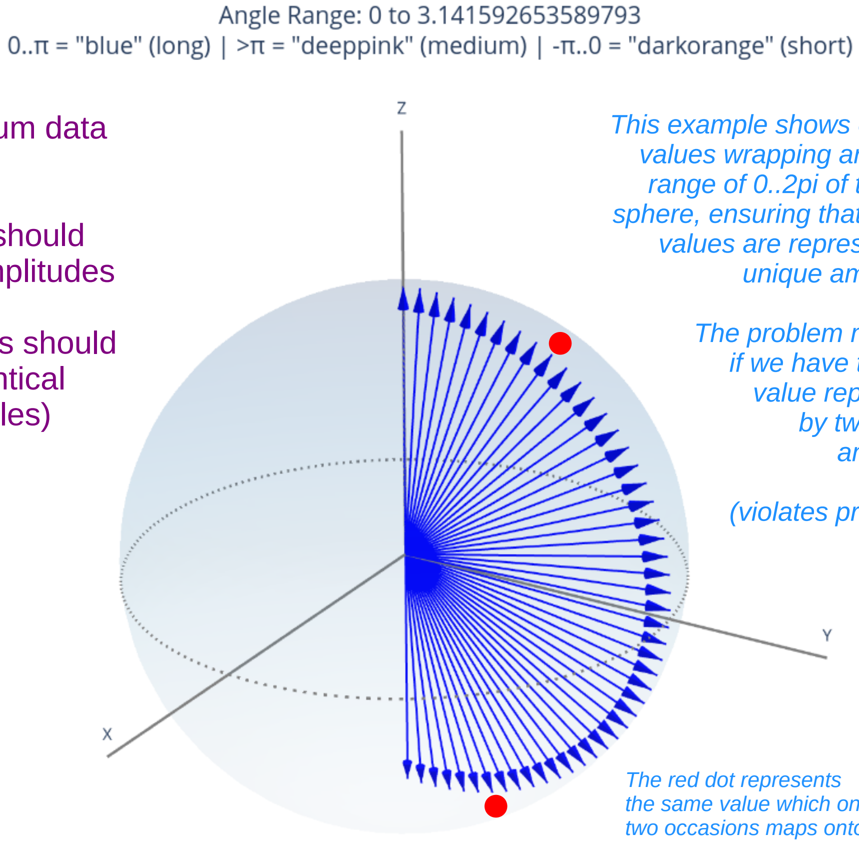
Two principles of quantum data encoding:

1) distinct data values should map onto distinct amplitudes (and angles)
2) the same data values should always map into identical amplitudes (and angles)

This example shows encoding values wrapping around the range of 0..2pi of the Bloch sphere, ensuring that different values are represented by unique amplitudes.

The problem may arise if we have the same value represented by two distinct amplitudes

(violates principle 2)



The red dot represents the same value which on two occasions maps onto two different angles and thus two amplitudes.

Commonly used
measurements and interpretation

Quantum circuits can be measured in many ways, e.g.

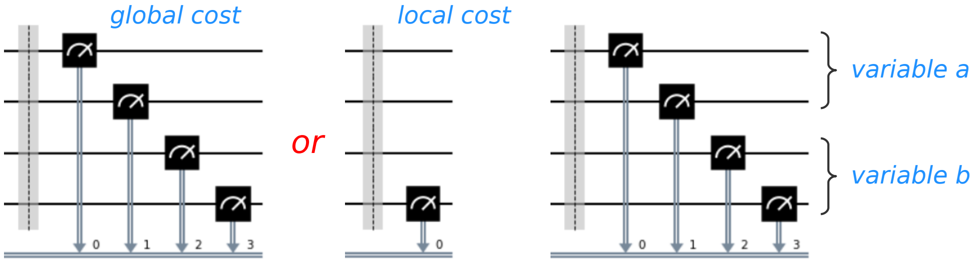
- all qubits (global cost / measurement)
- a few selected qubits (local cost / measurement)
- groups of qubits (each as a variable value)

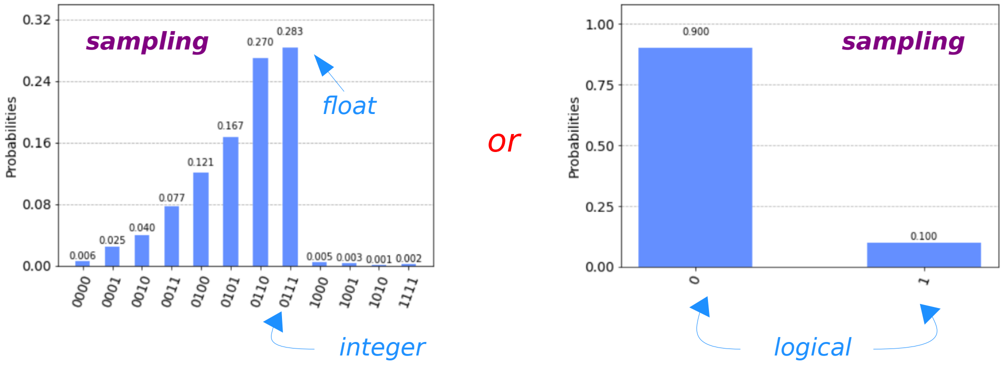
And received in many different formats, e.g.

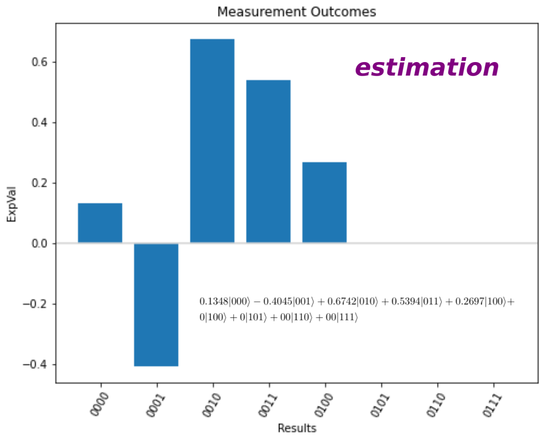
- as counts of outcomes (repeated measurements)
- as probabilities of outcomes (e.g. P(|0111⟩))
- as Pauli expectation values (i.e. of eigenvalues)
- as expectation of interpreted values (e.g. 0 to 15)
- as variance, etc.

Repeated measurement can be interpreted as outcomes of different types, e.g.

- as a probability distribution (as is)
- as a series of values (via expvals)
- as a binary outcome:
  single qubit measurement or parity of kets
- as an integer:
  most probable ket in multi-qubit measurement
- as a continuous variable:
  probability of the selected ket (e.g. $|0^n\rangle$)







Or we can measure expectation values of the circuit state and interpret them as a series of values in the range [-1..+1]

Beware that
adding 1 measurement → doubles the number of outcomes!

So... having ***n*** measurements leads to ***2^n*** outcomes

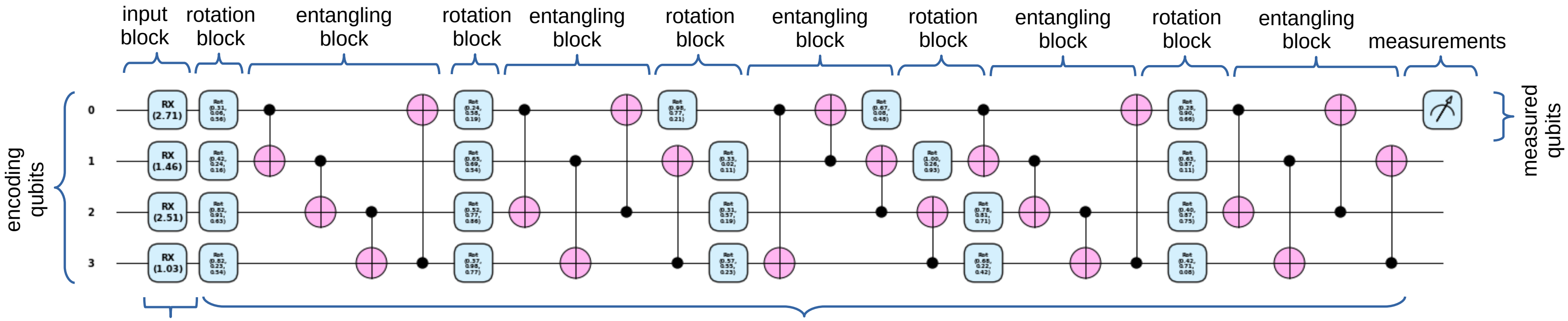
Maria Schuld and Francesco Petruccione.
Machine Learning with Quantum Computers. 2nd ed. Springer, 2021.

Ansatz design and training

A simple quantum classifier ...

Beware that
adding qubits adds
parameters and entanglements!

The number of states represented by the circuit **grows exponentially** with the number of qubits!



feature map
(with angle encoding)

Ansatz
(with trainable parameters / weights + biases)

*Data **reuploading** across circuit's width and depth*

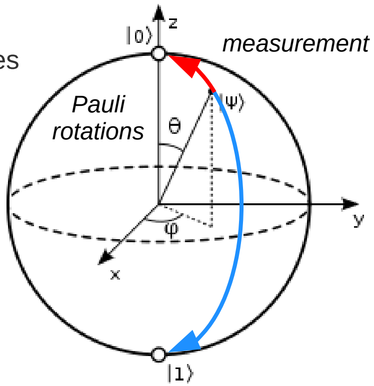
feature maps vary in:
structure and function

ansatze vary in:

- width (qubits #)
- depth (layers #)
- dimensions (param #)
- structure (e.g. funnelling)
- entangling (circular, linear, sca)

ansatz layers consist of:
rotation blocks and entangling blocks of R(x, y, z) and CNOT gates
(rotation) (entanglement)

rotation gates
alter qubit states around x, y, z axes



To execute a circuit we just apply it to input data and the optimum parameters

different cost functions:
R2, MAE, MSE, Huber, Poisson, cross-entropy, hinge-embedding, Kullback-Leibner divergence

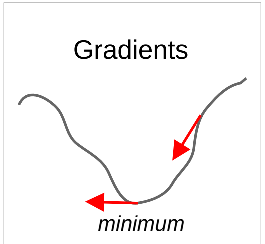
different optimisers:
gradient based (Adam, NAdam and SPSA)
linear approximation methods (COBYLA)
non-linear approximation methods (BFGS)
quantum natural gradient optimiser (QNG)

circuit execution on:
simulators (CPUs), accelerators (GPUs) and real quantum machines (QPUs)

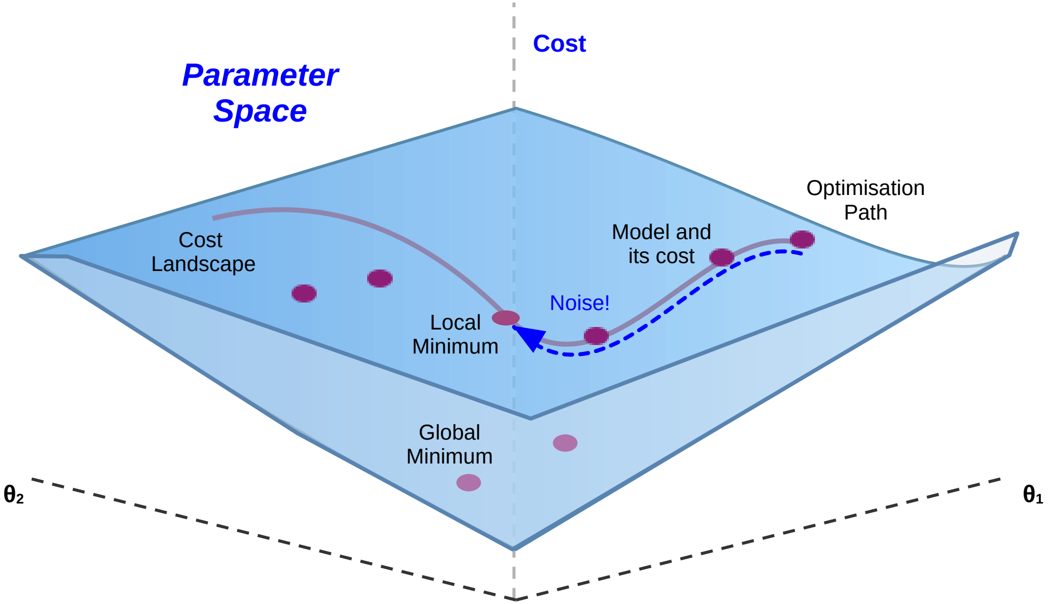
DL models
and their optimisation

- A Deep Learning model, such as a neural network, aims to represent some problem.
- The model takes *inputs* and calculates outputs via the layers of interconnected nodes.
- Each node has an activation, which is calculated as a *weighted* sum of nodes on its input, a *bias* added, and an *activation function* applied to produce its new value.
- All possible model parameterisations (weights and biases) form a multi-dimensional *parameter space*.
- The model quality is assessed by the *cost function*, where the lower the cost = the better the model.
- The costs of model parameterisations form a manifold over the parameter space - the *cost landscape*.
- The *optimisation process* relies on the shape of the landscape, which in turn is reflected in the *gradient* of points on the cost landscape.
- *Gradient descent* algorithms can assist in the identification of the model with the *minimum cost*.
- *Backpropagation* can also be used to efficiently re-calculate DL model's weights.

An optimiser uses gradients to recognise the shape of the cost landscape and to navigate it in search of such model parameters that produce the lowest cost

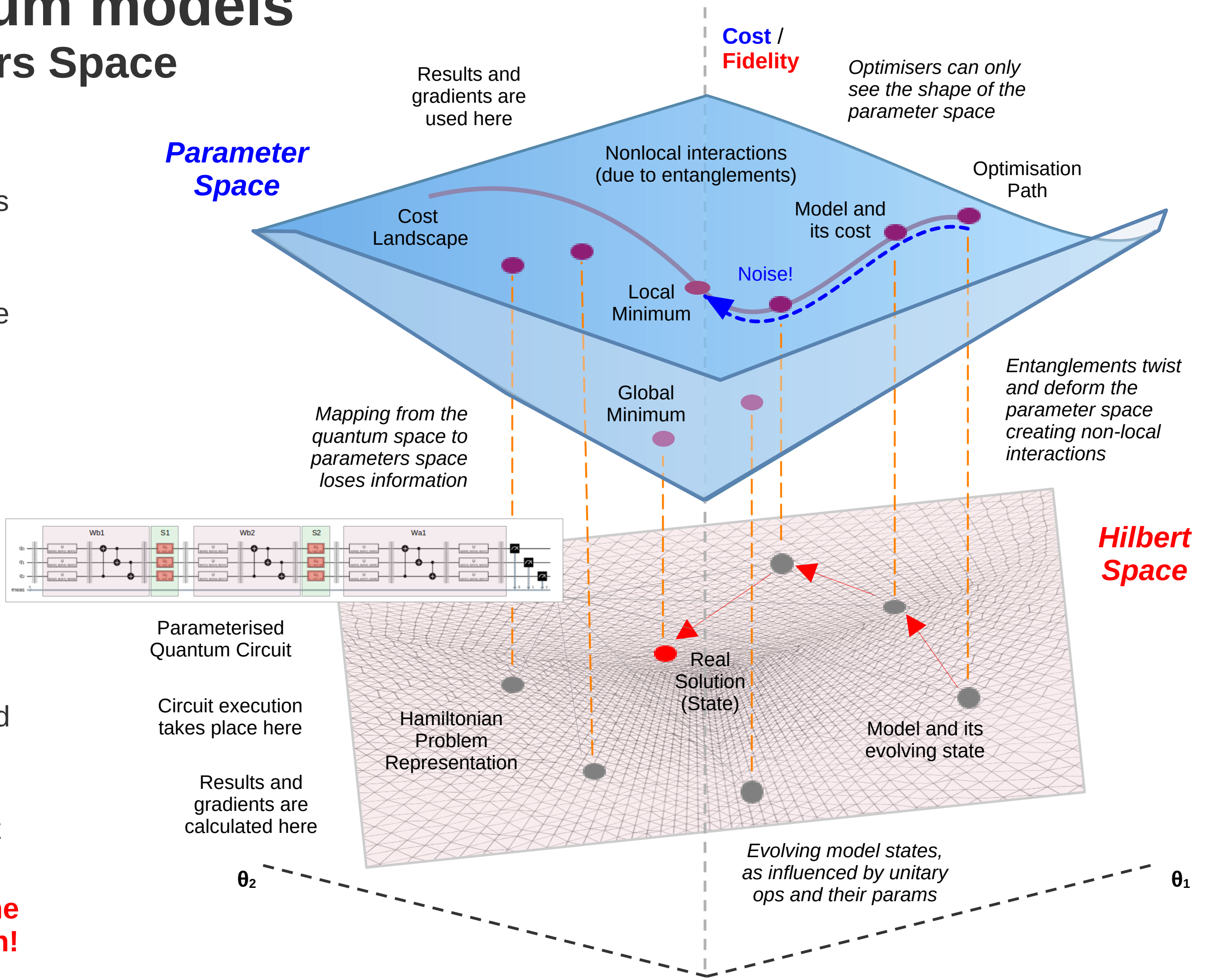


Gradients are local
i.e. during optimisation changes to the model cost influence gradients only in the immediate neighbourhood of the model



Working with quantum models
Hilbert Space vs Parameters Space

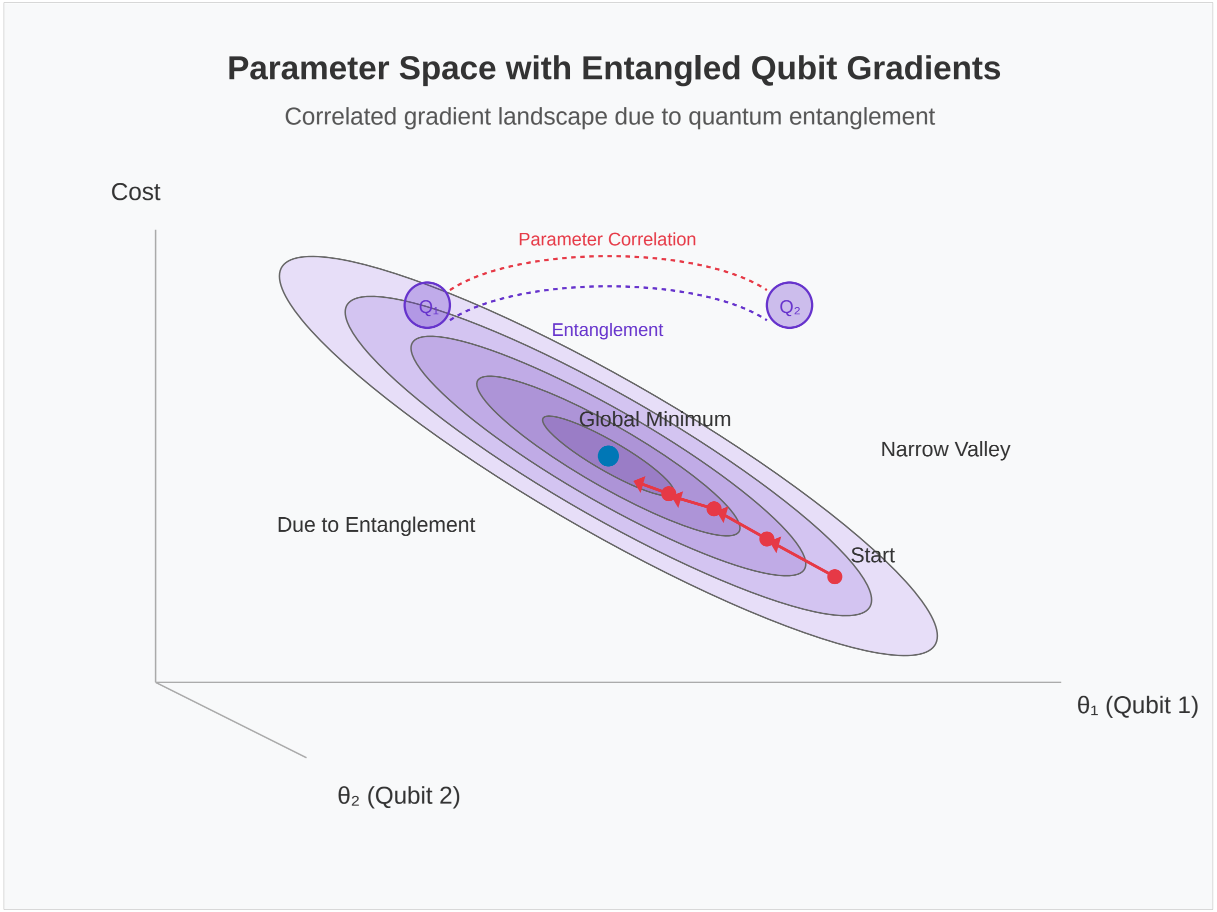
- **Hilbert state space** (dim = the number of qubits) is the quantum realm where the models and their states evolve in response to unitary operations as defined by the circuit gates
- **Data encoding** brings in classical data into the Hilbert space as unique and correlated quantum states during the model execution
- **Layers of circuit gates** determine the evolution of the quantum model's initial state into its final state during the circuit execution
- **Trainable parameter space** is a classical multi-dimensional space of circuit gate parameters, which the optimiser navigates
- **Entanglements** (defined by CNOTs) create and correlate non-separable qubit states, which alter the parameter space geometry, and also the cost landscape used by the optimiser
- **Measurement** of individual qubits collapses their states, consequently projecting the circuit state onto classical outcomes
- **The mapping from the quantum space to the classical parameter loses some information!**



Quantum model optimisation

- Optimisation of quantum model needs unique approaches due to the emergence of *non-local gradients*
- *Entangled qubits* result in *correlated parameters and gradients*, so the changes to one are reflected in the changes in distant others

- The cost landscape of highly entangled circuits commonly features *narrow valleys*
- Also, *backpropagation* cannot be used directly in training quantum circuits, as their state is not directly accessible and the measurement collapses the state
- *Gradient descend* can still be used with *global gradients*, i.e. those derived from the geometry of the cost landscape
- *Stochastic optimisation techniques* are highly effective when the cost landscape is smooth (no quantum moise)
- Other techniques are also available, such as *particle swarm optimisers*, these however are applicable to smaller models



PennyLane Demo
Everything is a function!

PennyLane (PL) …

- Supports *differentiable programming paradigm*
- Integrates seamlessly with the *Python*
- Has a range of operations for *state preparation*, *gates* and *measurements*
- Supports creation of flexible *quantum algorithms*
- Executes on *simulators* and *quantum hardware*
- Supports *error mitigation*
- Extends its *quantum gradients* with those from JAX, PyTorch, Keras, TensorFlow, or NumPy
- Supports *hybrid quantum-classical models*
- Allows training with *hardware-compatible gradients* and *higher-order derivatives*
- Provides numerous quantum models, such as: *QNNs*, *quantum kernels* and *Fourier models*
- Can be extended with models and optimisers from other SDKs, e.g. *PyTorch* and *TensorFlow*

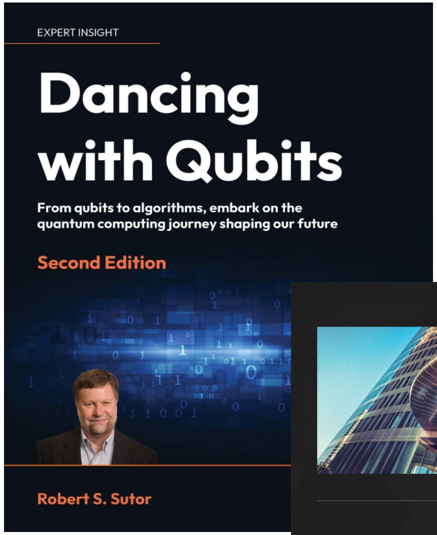
PennyLane Demo:

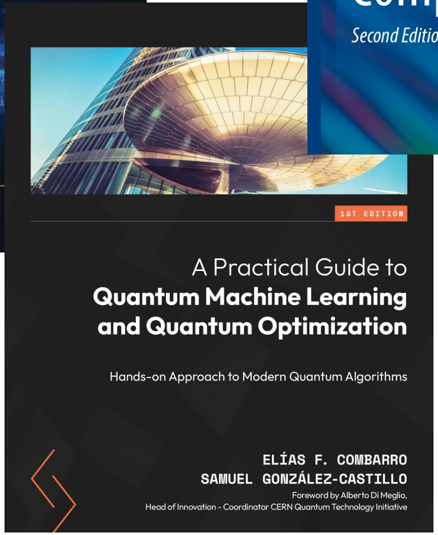
- Create a simple PL model to fit a simple function
- Learn to initialise model weights
- Explore the impact of ansatz structure on performance
- Create minimalistic quantum models
- Learn the interaction of data encoding and ansatz
- Investigate different types of entangling
- Apply the best solution to more complex data
- Learn about stamina and wisdom in QML development

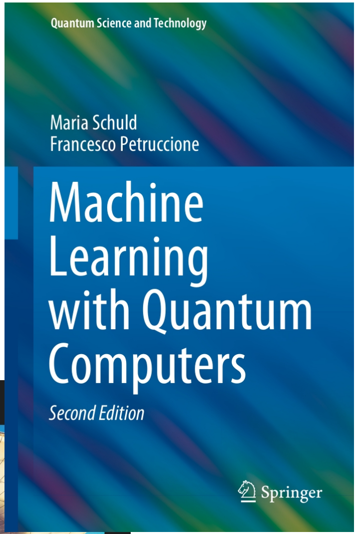
Key takeaways:

- Plan model development, tests and experiments
- Bad data encoding spoils the bunch!
- Strong entanglement improves the data fit
- More width and depth = the curse of dimensionality
- Carefully consider your quantum model initialisation
- Surprise - a single qubit model still works! (and well)
- More training does not solve the problems
- Data reuploading makes a huge difference!

Recommended reading
on QML







PennyLane: Automatic differentiation of hybrid quantum-classical computations

Ville Bergholm,[1] Josh Izaac,[1] Maria Schuld,[1] Christian Gogolin,[1] M. Sohaib Alam,[2] Shahnawaz Ahmed,[3] Juan Miguel Arrazola,[1] Carsten Blank,[4] Alain Delgado,[1] Soran Jahangiri,[1] Keri McKiernan,[2] Johannes Jakob Meyer,[5] Zeyue Niu,[1] Antal Száva,[1] and Nathan Killoran[1]

[1]Xanadu, 777 Bay Street, Toronto, Canada
[2]Rigetti Computing, 2919 Seventh Street, Berkeley, CA 94710
[3]Wallenberg Centre for Quantum Technology, Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden
[4]data cybernetics, Martin-Kolmsperger-Str 26, 86899 Landsberg, Germany
[5]Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany

14 Feb 2020

Modern applications of machine learning in quantum sciences

Anna Dawid[1,2*], Julian Arnold[3†], Borja Requena[2†], Alexander Gresch[4†], Marcin Płodzień[2], Kaelan Donatella[5], Kim A. Nicoli[6,7], Paolo Stornati[2], Rouven Koch[8], Miriam Büttner[9], Robert Okuła[10,11], Gorka Muñoz-Gil[12], Rodrigo A. Vargas-Hernández[13,14], Alba Cervera-Lierta[15], Juan Carrasquilla[14], Vedran Dunjko[16], Marylou Gabrié[17], Patrick Huembeli[18,19], Evert van Nieuwenburg[16,20], Filippo Vicentini[18], Lei Wang[21,22], Sebastian J. Wetzel[23], Giuseppe Carleo[18], Eliška Greplová[24], Roman Krems[25], Florian Marquardt[26,27], Michał Tomza[1], Maciej Lewenstein[2,28] and Alexandre Dauphin[2*]

1 Faculty of Physics, University of Warsaw, Poland
2 ICFO - Institut de Ciències Fotòniques, The Barcelona Institute of Science and Technology, 08860 Castelldefels (Barcelona), Spain
3 Department of Physics, University of Basel, Switzerland
4 Quantum Technology Research Group, Heinrich-Heine-Universität Düsseldorf, Germany
5 Université de Paris, CNRS, Laboratoire Matériaux et Phénomènes Quantiques, France
6 Machine Learning Group, Technische Universität Berlin, Germany
7 BIFOLD, Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany
8 Department of Applied Physics, Aalto University, Espoo, Finland
9 Institute of Physics, Albert-Ludwig University of Freiburg, Germany
10 International Centre for Theory of Quantum Technologies, University of Gdańsk, Poland
11 Department of Algorithms and System Modeling, Faculty of Electronics, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Poland
12 Institute for Theoretical Physics, University of Innsbruck, Austria
13 Department of Chemistry, University of Toronto, Canada
14 Vector Institute for Artificial Intelligence, MaRS Centre, Toronto, Canada
15 Barcelona Supercomputing Center, Spain
16 LIACS, Leiden University, The Netherlands
17 CMAP, École Polytechnique, France
18 Institute of Physics, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
19 Menten AI, Inc., Palo Alto, California, United States of America
20 Niels Bohr Institute, Copenhagen, Denmark
21 Beijing National Lab for Condensed Matter Physics and Institute of Physics, Chinese Academy of Sciences, Beijing, China
22 Songshan Lake Materials Laboratory, Dongguan, China
23 Perimeter Institute for Theoretical Physics, Waterloo, Canada
24 Kavli Institute of Nanoscience, Delft University of Technology, NL-2600 GA Delft, The Netherlands
25 Department of Chemistry, University of British Columbia, Vancouver, Canada
26 Max Planck Institute for the Science of Light, Erlangen, Germany
27 Department of Physics, Friedrich-Alexander Universität Erlangen-Nürnberg, Germany
28 ICREA, Pg. Lluís Companys 23, 08010 Barcelona, Spain
† These authors contributed equally.

* Anna.Dawid@fuw.edu.pl, Alexandre.Dauphin@icfo.eu

June 23, 2022

Abstract

In these Lecture Notes, we provide a comprehensive introduction to the most recent advances in the application of machine learning methods in quantum sciences. We cover the use of deep learning and kernel methods in supervised, unsupervised, and reinforcement learning algorithms for phase classification, representation of many-body quantum states, quantum feedback control, and quantum circuits optimization. Moreover, we introduce and discuss more specialized topics such as differentiable programming, generative models, statistical approach to machine learning, and quantum machine learning.

1

Thank you!

Any questions?

Photos from Unsplash
Enquanted is being somewhere in-between Enchanted and Entangled