Kwantowe sieci neuronowe

dr Sebastian Zając

Seminarium
Wojskowa Akademia Techniczna

06.03.2025

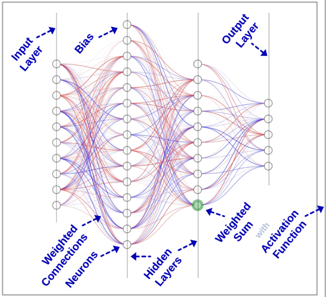
Neural Networks

A class of complex ML models

- *Multi-Layer Perceptrons* (MLP) take numerical input and produce numerical output
- MLPs are structured into layers
- *Layers* consist of neurons
- *Neurons* hold activation - value in range [-1,+1]
- *Weighted links* connect neurons of adj. layers
- *Activation* is a weighted sum of activations of neurons from the previous layer
- *Bias* is a value added to the sum
- *Activation function* is applied to the sum to scale the result back to the interval [-1, 1]
- *Optimisation* is the process to identify optimum weights and biases, it is commonly iterative
- *Optimisation aims* to reduce cost or aggregated loss, a distance between the calculated and expected results

MLP = a simple neural network

capable of learning any "smooth" function
learns to associate inputs with outputs



- This process can be accelerated by using specialised hardware, e.g. GPUs or TPUs
- Other NNs: CNN, AE, GAN, LSTM + QNNs

Name	Plot	Function, $g(x)$
Identity		x
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$
Logistic, sigmoid, or soft step		$\sigma(x) \doteq \frac{1}{1+e^{-x}}$
Hyperbolic tangent (tanh)		$\tanh(x) \doteq \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Soboleva modified hyperbolic tangent (smht)		$\operatorname{smht}(x) \doteq \frac{e^{ax} - e^{-bx}}{e^{cx} + e^{-dx}}$
Rectified linear unit (ReLU)[10]		$(x)^+ \doteq \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} = \max(0, x) = x\mathbf{1}_{x>0}$
Gaussian Error Linear Unit (GELU)[2]		$\frac{1}{2}x\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right) = x\Phi(x)$
Softplus[11]		$\ln(1 + e^x)$
Exponential linear unit (ELU)[12]		$\begin{cases} \alpha(e^x - 1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α
Scaled exponential linear unit (SELU)[13]		$\lambda\begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameters $\lambda = 1.0507$ and $\alpha = 1.67326$
Leaky rectified linear unit (Leaky ReLU)[14]		$\begin{cases} 0.01x & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$
Parametric rectified linear unit (PReLU)[15]		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter α
Sigmoid linear unit (SiLU,[2] Sigmoid shrinkage,[16] SiL,[17] or Swish-1[18])		$\frac{x}{1+e^{-x}}$
Gaussian		e^{-x^2}

Wikipedia

Deep learning algorithm flow

Model

model with trainable parameters $f(x;\theta) = \sigma(Wx + b)$ where $\theta = \{W, b\}$

Cost function

$C = \sum_{x,y}(f(x;\theta) - y)^2$

Gradient descent

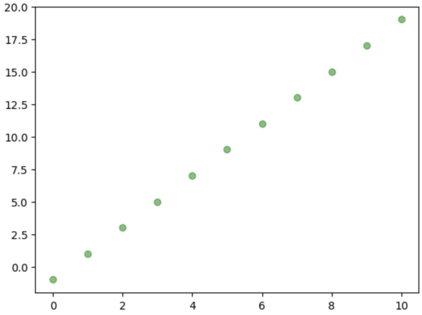
- Compute gradient of cost w.r.t parameters
- Update parameters $\theta^{t+1} = \theta_t - \eta\nabla_\theta C$

ML models as a special case of NN

Wygenerujemy niezaszumione dane na podstawie wzoru $y = 2x - 1$. Na podstawie zbioru danych postaramy się oszacować nieznane parametry czyli wyraz przy x ($\alpha_1 = 2$) i wyraz wolny ($\alpha_0 = -1$).

```
import numpy as np
import matplotlib.pyplot as plt

# zbior danych
x = range(11)
y = [2*xi - 1 for xi in x]
plt.plot(x, y, 'go', label='True data', alpha=0.5)
```



Model regresji liniowej dla jednej zmiennej można zrealizować jako prostą jednowarstwową sieć neuronową. Cały proces można zrealizować za pomocą obiektu `torch.nn.Linear`

```
import torch

class LinearRegression(torch.nn.Module):

    def __init__(self, inputSize, outputSize):
        super(LinearRegression, self).__init__()
        self.layers = torch.nn.Sequential(
            torch.nn.Linear(inputSize, outputSize)
        )

    def forward(self, x):
        return self.layers(x)
```

Aby nasze dane mogłybyć przeliczane przez bibliotekę PyTorch musimy je przetworzyć na tensory - czyli obiekty z biblioteki PyTorch.

```
# dostosowanie do pytorch
x = np.array(x, dtype=np.float32)
y = np.array(y, dtype=np.float32)

X_train = torch.from_numpy(x).view(-1,1)
y_train = torch.from_numpy(y).view(-1,1)
```

Uwaga - ponieważ mamy jedną zmienną zawierającą 10 przypadków - potrzebujemy listy składającej się z 10 list jednoelementowych.

Możemy utworzyć model i wybrać optymalizator z funkcją kosztu.

```
# obiekt liniowej regresji w wersji sieci nn
lr_model = LinearRegression(1,1)

criterion = torch.nn.MSELoss()
optimizer = torch.optim.SGD(lr_model.parameters(), lr=0.01)
```

Czy zadziała?

Obliczenia kwantowe

Obliczenia kwantowe to nowy paradygmat przetwarzania informacji wykorzystujący własności mechaniki kwantowej - *kubit, superpozycja, splątanie.*

Modele obliczeń kwantowych

- Quantum Circuits - bramkowy model obliczeń kwantowych
- adiabatyczne obliczenia kwantowe,
- topologiczne komputery kwantowe

jeszcze jedna zasada

Dekoherencja - oddziaływanie „niszczące” stan kwantowy
ERA Noisy Intermediate-Scale Quantum (NISQ)

What is
Quantum Technology

***Quantum computing** and*
Quantum information science
study of the information processing tasks that can be accomplished using quantum mechanical systems (Nielsen and Chuang, 2010)

$$d_{\gamma,n}(\mathcal{M}_\Theta) = \frac{2\log\left(\frac{1}{V_\Theta}\int_\Theta \sqrt{\det\left(\mathrm{id}_d + \frac{\gamma n}{2\pi\log(n)}\hat{F}(\theta)\right)}\,\mathrm{d}\theta\right)}{\log\left(\frac{\gamma n}{2\pi\log n}\right)}$$

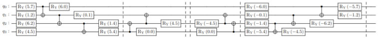
$$\hat{F}_{i,j}(\theta) = d\frac{V_\Theta}{\int_\Theta \mathrm{tr}(F(\theta))\mathrm{d}\theta}F_{i,j}(\theta),$$

Not an easy path

$$V_\Theta = \int_\Theta \mathrm{d}\theta$$

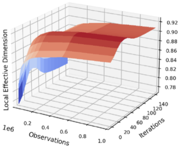
Quantum finance
Quantum chemistry
Quantum optimisation
Quantum machine learning
problem-solving with quantum technology

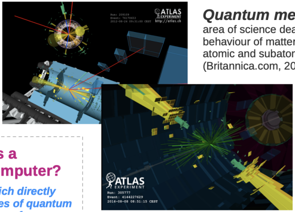
Cybulski & Nguyen
Barren Plateaus
in QNNs and VQAs



What is a Quantum Computer?

It is a device which directly applies the principles of quantum mechanics to perform computational tasks

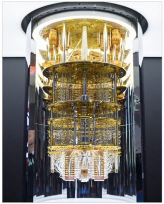




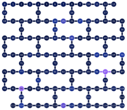
Quantum mechanics
area of science dealing with the behaviour of matter and light on the atomic and subatomic scale (Britannica.com, 2020)

CERN
Atlas particle detector
Large Hadron Collider

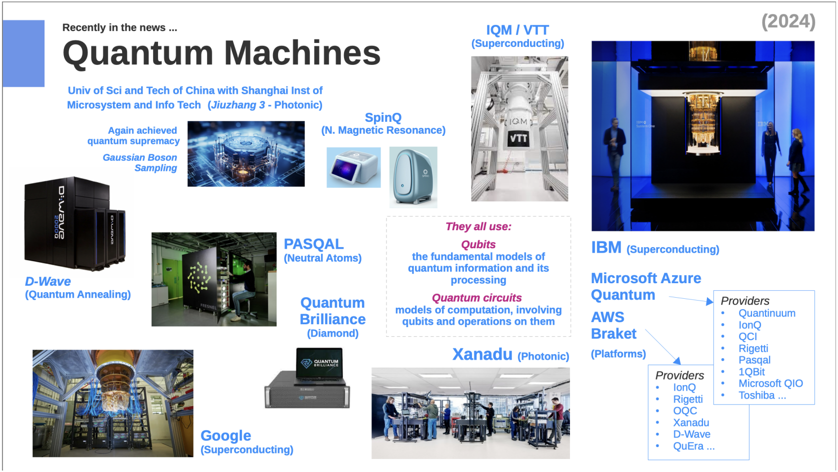
There are other quantum technologies e.g. quantum sensing



IBM superconducting quantum machine (127 qubits on cloud)



Quantum engineering
building quantum devices



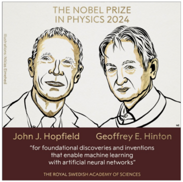
(2024)
Recently in the news ...
Quantum Machines
Univ of Sci and Tech of China with Shanghai Inst of Microsystem and Info Tech (Jiuzhang 3 - Photonic)
Again achieved quantum supremacy
Gaussian Boson Sampling
SpinQ
(N. Magnetic Resonance)
IQM / VTT
(Superconducting)
IQM
VTT
D-Wave
(Quantum Annealing)
PASQAL
(Neutral Atoms)
Quantum Brilliance
(Diamond)
QUANTUM BRILLIANCE
They all use:
Qubits
the fundamental models of quantum information and its processing
Quantum circuits
models of computation, involving qubits and operations on them
IBM (Superconducting)
Microsoft Azure Quantum
AWS Braket
(Platforms)
Providers
Quantinuum
IonQ
QCI
Rigetti
Pasqal
1QBit
Microsoft QIO
Toshiba ...
Providers
IonQ
Rigetti
OQC
Xanadu
D-Wave
QuEra ...
Xanadu (Photonic)
Google
(Superconducting)

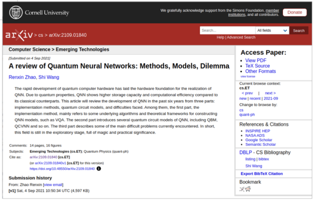
Recommended reading
on QNN + Deep Learning

Classical Neural Networks / Deep Learning

Chapter 10: QNN

Neural Networks Fathers





Computer Science > Emerging Technologies

[Submitted on 4 Sep 2021]

A review of Quantum Neural Networks: Methods, Models, Dilemma

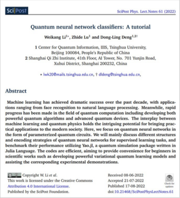
Renxin Zhao, Shi Wang

The rapid development of quantum computer hardware has laid the hardware foundation for the realization of QNN. Due to quantum properties, QNN shows higher storage capacity and computational efficiency compared to its classical counterparts. This article will review the development of QNN in the past six years from three parts: implementation methods, quantum circuit models, and difficulties faced. Among them, the first part, the implementation method, mainly refers to some underlying algorithms and theoretical frameworks for constructing QNN models, such as VQA. The second part introduces several quantum circuit models of QNN, including QBM, QCVNN and so on. The third part describes some of the main difficult problems currently encountered. In short, this field is still in the exploratory stage, full of magic and practical significance.

Comments: 14 pages, 16 figures
Subjects: Emerging Technologies (cs.ET); Quantum Physics (quant-ph)
Cite as: arXiv:2109.01840 [cs.ET] (or arXiv:2109.01840v1 [cs.ET] for this version) https://doi.org/10.48550/arXiv.2109.01840

Submission history
From: Zhao Renxin [view email]
[v1] Sat, 4 Sep 2021 10:50:34 UTC (4,597 KB)

Variety of QNN Types



SciPost Phys. Lect.Notes 61 (2022)

Quantum neural network classifiers: A tutorial

Weikang Li[1*], Zhide Lu[1] and Dong-Ling Deng[1,2†]

1 Center for Quantum Information, IIIS, Tsinghua University, Beijing 100084, People's Republic of China
2 Shanghai Qi Zhi Institute, 41th Floor, AI Tower, No. 701 Yunjin Road, Xuhui District, Shanghai 200232, China

* lwk20@mails.tsinghua.edu.cn, † dldeng@tsinghua.edu.cn,

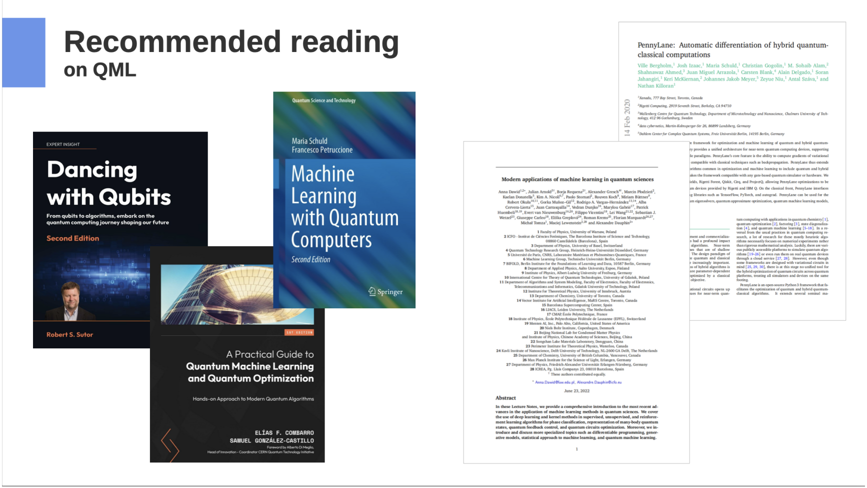
Abstract

Machine learning has achieved dramatic success over the past decade, with applications ranging from face recognition to natural language processing. Meanwhile, rapid progress has been made in the field of quantum computation including developing both powerful quantum algorithms and advanced quantum devices. The interplay between machine learning and quantum physics holds the intriguing potential for bringing practical applications to the modern society. Here, we focus on quantum neural networks in the form of parameterized quantum circuits. We will mainly discuss different structures and encoding strategies of quantum neural networks for supervised learning tasks, and benchmark their performance utilizing Yao.jl, a quantum simulation package written in Julia Language. The codes are efficient, aiming to provide convenience for beginners in scientific works such as developing powerful variational quantum learning models and assisting the corresponding experimental demonstrations.

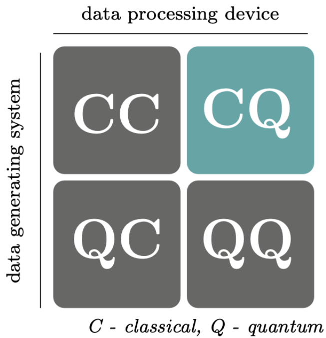
Copyright W. Li et al. This work is licensed under the Creative Commons Attribution 4.0 International License. Published by the SciPost Foundation.

Received 08-06-2022
Accepted 21-07-2022
Published 17-08-2022
doi:10.21468/SciPostPhysLectNotes.61

QNN in Julia


Recommended reading
on QML
EXPERT INSIGHT
Dancing with Qubits
From qubits to algorithms, embark on the quantum computing journey shaping our future
Second Edition
Robert S. Sutor
Quantum Science and Technology
Maria Schuld
Francesco Petruccione
Machine Learning with Quantum Computers
Second Edition
Springer
1ST EDITION
A Practical Guide to Quantum Machine Learning and Quantum Optimization
Hands-on Approach to Modern Quantum Algorithms
ELÍAS F. COMBARRO
SAMUEL GONZÁLEZ-CASTILLO
PennyLane: Automatic differentiation of hybrid quantum-classical computations
14 Feb 2020
Modern applications of machine learning in quantum sciences
Abstract

Kwantowe uczenie maszynowe

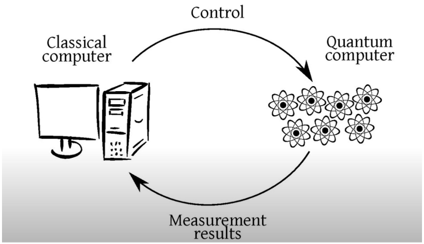


$$|f(x;\theta)\rangle$$

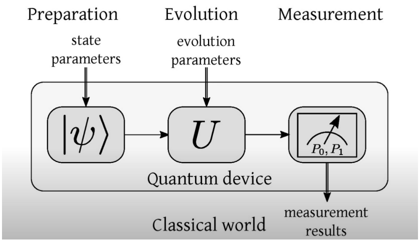
Czy możemy zamienić „model” wykorzystując obliczenia kwantowe?

Proces obliczeń kwantowych

Quantum computation control loop



Computation as experiment

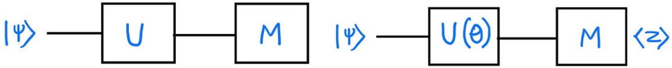


Przykład (deterministyczny): Algorytm Deutsch-Jozsa

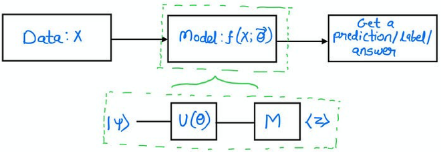
Wariacyjne modele kwantowe

Variational Quantum Circuits

Trenujmy nasze komputery kwantowe tak jak robimy to w sieciach neuronowych.



Variational Circuit as Classifier:



Variational quantum eigensolver, Variational quantum classifier, Quantum Support Vector Machine, ...

Qubits

in scientific terms

In practice qubits involve *elementary particles*, such as *ions*, *photons*, single *atoms*, *electrons*, even *defects in diamonds*; and, their behaviour is governed by Physics (Nature / Universe)

A qubit *represents a state* of such a particle, e.g. an electron spin, which can be up or down, (written formally as $|\uparrow\rangle$ and $|\downarrow\rangle$ or $|0\rangle$ and $|1\rangle$), which are called the *basis states*

It is possible to change the state of a qubit with certain predetermined *operations*, such as rotation or reversing the position of the qubit state

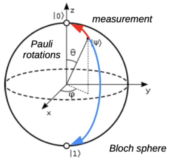
Afterwards, the qubit is in a state of *superposition*, or a combination, of its basis states up and down

The superposition state is the actual state of elementary particles, not its math description

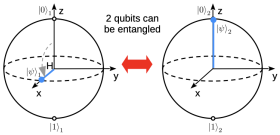
It is impossible to determine the qubit's state without its *measurement*

Qubit measurement returns only the basis state that is "likely" to be closest to its superposition state, which also destroys the qubit state

The outcome of measurement is precise but probabilistic



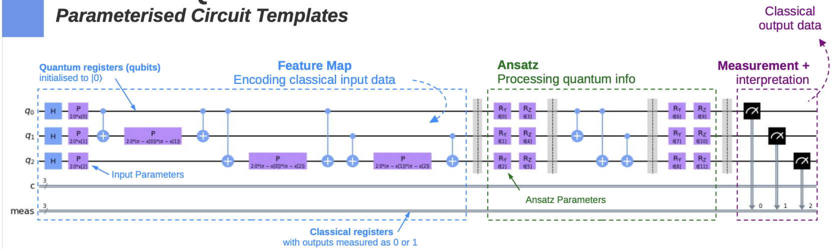
Qubits can be *entangled*, then they start behaving as a unit with a common complex state, until they are measured or until some external factor (*noise*) destroys their entanglement



What makes quantum computers special?

Qubit *superposition* (parallel choices) and *entanglement* (exponential combination of choices and their filtering), as well as *measurement* (collapse of choices and randomness), is what gives quantum computers their immense computational power allowing some problems to be solved in minutes rather than 1000s of years!

Variational Quantum Models =
Parameterised Circuit Templates



Quantum circuits are static

Data and operations are hard-coded

New data / operation params → new circuit

Typically, the circuit consists of three blocks:

- a feature map (input)
  encodes classical data as circuit state
- an ansatz (processing)
  alters circuit state
- measurements (output)
  measures circuit state into classical data

Variational Quantum Algorithm

VQA is an ***iterative process***

VQA uses ***cost/loss function*** and ***optimiser***

VQA has ***difficulties***:

- The problem at hand
- Large circuits with many parameters
- Complex measurement strategy
- Unsupervised learning
- Emergence of barren plateaus

A typical VQA process

The ansatz parameters are initialised to some values, e.g. zero or random

The feature map parameters are bound to the new input data

The parameter values are used to create a new circuit

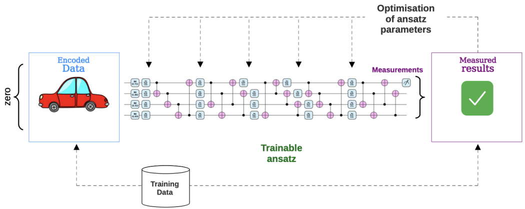
The circuit is executed

The circuit quantum state is then measured

Cost function is applied to measurement results and expected values

The cost of difference is calculated

Based on the difference and previous parameters the new parameters values are proposed



9 / 16

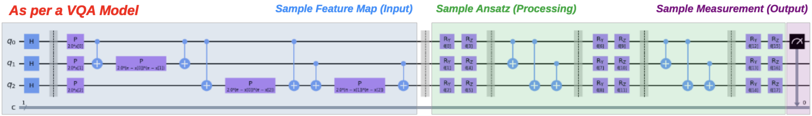
Quantum Neural Networks
Specifically Quantum MLPs

Abbas, Amira, David Sutter, Christa Zoufal, Aurelien Lucchi, Alessio Figalli, and Stefan Woerner. "The Power of Quantum Neural Networks." Nature Computational Science 1, no. 6 (June 2021): 403–9.
Schreiber, Amelie. "Quantum Neural Networks for FinTech." Medium, May 8, 2020.

- The QNN variational model is typically represented by a quantum circuit of three components, i.e.
  - *feature map* encoding QNN's classical input data and preparing the circuit's quantum state
  - *ansatz* consisting of several layers of trainable parameters (Pauli rotations), responsible for quantum state processing and transformation
  - *measurement* of the circuit's quantum state, which can subsequently be interpreted as QNN' classical output
- QNNs can be trained with variational quantum algorithms and a wide range of classical optimisers.
- Pure quantum training strategies are also possible.

QNNs

- can deal with highly complex computation (as QM)
- can deal with large volume of data (as NN)
- can process entire probabilistic distributions of values (superposition) and utilise parameters space of exponential size (entanglement)
- require repeated execution to produce output
- are missing some efficiency of classical NNs (non-linear activations and regularisation strategies)
- are difficult to process on quantum simulators of limited computational capacity
- need experimentation and extensive data preparation – there is no magic in quantum tech!

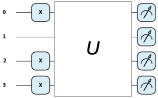


Data embedding:
Basis encoding and decoding

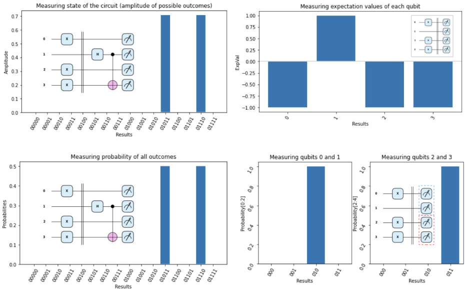
There are many different approaches to quantum data encoding and decoding that are suitable for QNN, e.g. basis, angle and amplitude embedding.

Basis embedding is the commonly used strategy for quantum encoding and decoding of integer numbers, where:

- qubits act as bits of the encoded numbers
- circuit state can be interpreted as bits of the numeric value on output
- application needs a single (or very few) integer value on input and output



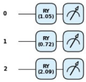
Encoding and measuring number 11

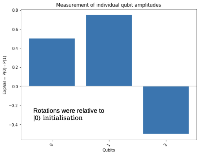


Maria Schuld and Francesco Petruccione.
Machine Learning with Quantum Computers. 2nd ed. Springer, 2021.
http://link.springer.com/book/10.1007/978-3-030-83098-4.

QNN model

Data embedding:
Angle encoding and decoding





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]

Angle embedding represents numeric values as properties of qubit state rotation (angle, amplitude or probability)

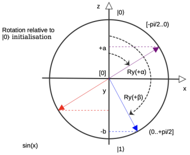
The rotation operators are the basic quantum operation

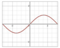
Encoding rotations are performed around x, y, z axes of the Bloch sphere (multiple values per qubit are possible)

Rotations are relative to a specific qubit state, e.g. |0⟩

Input encoding can be repeated across the circuit, called reuploading, which improves the model performance

As training will place qubit states in areas x < 0 and around the z axis, measurements may not distinguish these states from “pure” x > 0 and z = 0.

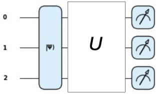


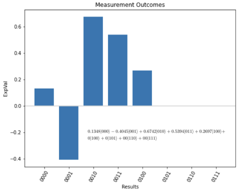


Data embedding:
Amplitude encoding and decoding

Sutor, R.S., 2024. Dancing with Qubits, Second Edition. Packt Pub.

Maria Schuld and Francesco Petruccione. *Machine Learning with Quantum Computers. 2nd ed.* Springer, 2021. http://link.springer.com/book/10.1007/978-3-030-83098-4.





There are many other methods of data encoding, e.g. QRAM, time-evolution, or dense-angle, or Hamiltonian encoding.

Many of these methods are offered as ready-made feature maps.

Qiskit feature maps include:

- ZfeatureMap
- ZZFeatureMap
- PauliFeatureMap

Amplitude embedding is one of the most useful encoding / decoding strategies

Unless supported by the quantum platform, it is considered the most difficult (see Sutor 2024)

In amplitude encoding, each data point is encoded as expectation value of multi-qubit measurement of all qubits' states

This way, we can embed 2^{qubits} numbers into a circuit!

Consider a vector:

v = [0.1, -0.3, 0.5, 0.4, 0.2],

which needs to be normalised by the vector length:

sqrt(0.1^2+$(-0.3)^2$+0.5^2+0.4^2+0.2^2],

which results in a new vector (approximately):

v' ≈ [0.13484 -0.40452 0.6742 0.53936 0.26968].

To encode 5 amplitudes in a quantum circuit, we need at least 3 qubits. Thus, resulting in the following encoding:

$$0.1348|000\rangle - 0.4045|001\rangle + 0.6742|010\rangle + 0.5394|011\rangle + 0.2697|100\rangle + 0|100\rangle + 0|101\rangle + 00|110\rangle + 00|111\rangle$$

We will rely on PennyLane and Qiskit to generate quantum gates for this circuit ...

Commonly used
measurements and interpretation

There are many ways of obtaining the outcome of a circuit execution, e.g. we can measure:

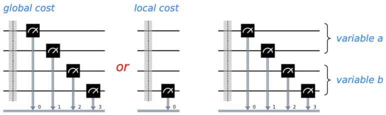
- all qubits (global cost)
- a few qubits (local cost)
- groups of qubits
- as counts of repeated measurements
- as probabilities of |0⟩ and |1⟩
- as expectation values, P(0)-P(1)
- as variance, etc.

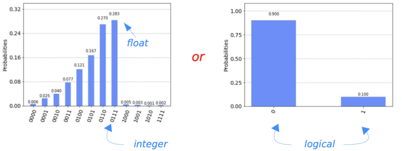
Repeated circuit measurement can be interpreted as outcomes of different types, e.g.

- as a binary outcome:
  single qubit measurement
- as an integer:
  multi-qubit measurement
- as a continuous variable:
  expectation value of a specific outcome

Circuit state measurement has an impact on the calculation of the loss/cost function

Model measurement and interpretation of results share their fundamental concepts and methods with data encoding.

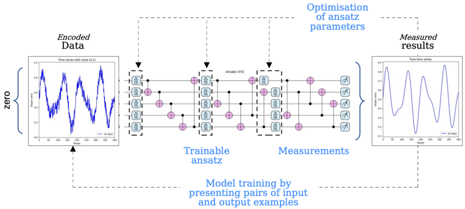




Probability distribution of measurements can be further interpreted, e.g. we could check parity of the probability kets (e.g. |110⟩ is even, while |111⟩ is odd), add all even and odd probabilities respectively, and treat the result as a logical measurement.

Training of Quantum Neural Networks

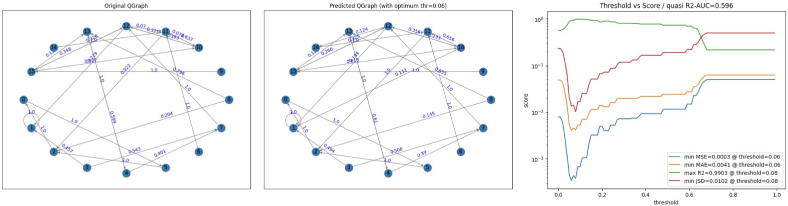
- QNN training needs a loss / cost function and an optimiser of the model parameters
- The loss function and optimiser can either be pure quantum or hybrid
- Pure quantum approach often relies on quantum adiabatic or quantum annealing optimisation, and Grover-like amplitude amplification
- A hybrid approach uses variational quantum algorithms (VQA), and relies on the QNN execution on a quantum machine, and its parameters optimisation conducted on a classical machine
- Hybrid training of QNNs is identical to training classical NN models



Current Work

Neurons hum, entrained,
Quantum states in circuits chained—
Truth blooms, unrestrained.

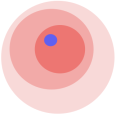
Quantum Haiku by DeepSeek



Quantum Graphs

Development of concepts and formalisms related to "quantisation" of classical data structures, such as time series, signals and graphs.

Quantum graphs for instance will assist highly efficient representation and processing of very large interconnected structures, e.g. when assisting management of social networks, identification of sub-graph communities and detection of anomalies in graphs.



Enquanted is being somewhere in-between Enchanted and Entangled

Dziękuję za uwagę!

sebastian.zajac@sgh.waw.pl