Machine Learning with Quantum Computers

Maria Schuld

Xanadu and University of KwaZulu-Natal

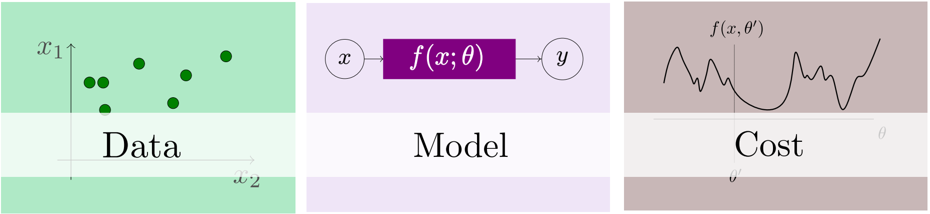
SLAC Seminar, December 2020

Agenda

- Some context on QML
- Attempts to unite QML and HEP
- A critical comment

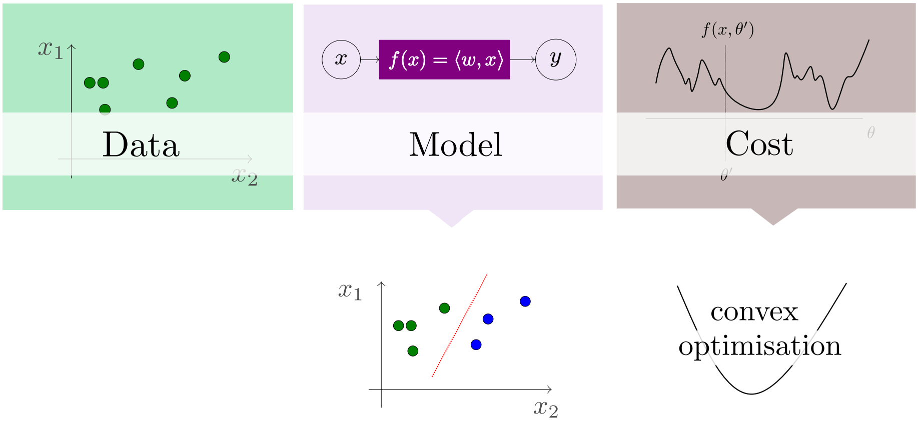
QML

Machine Learning

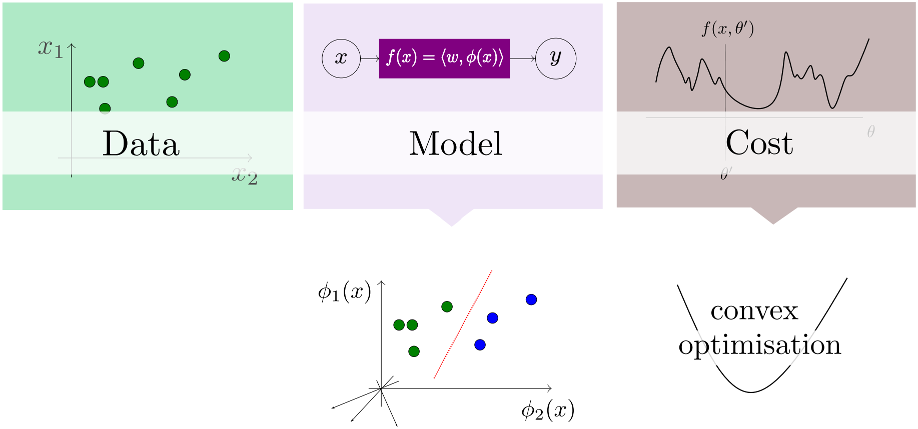


Use data samples
to construct model
that minimises cost
on unseen data.

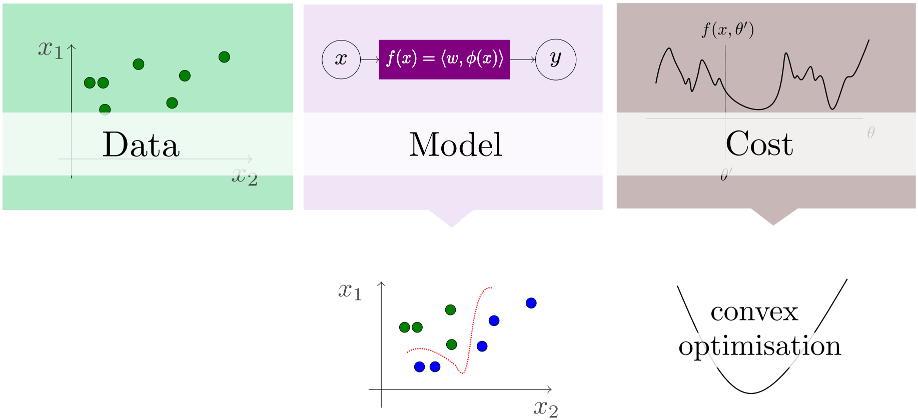
Linear models



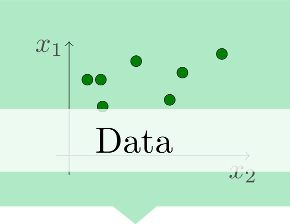
Kernel methods

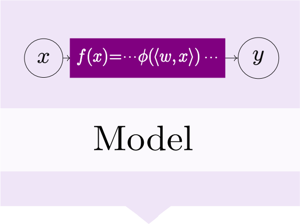


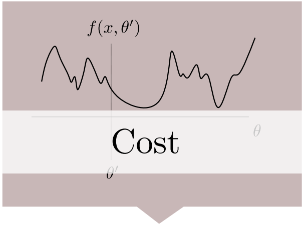
Kernel methods



Deep learning

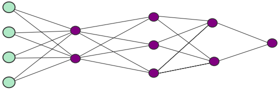






Big

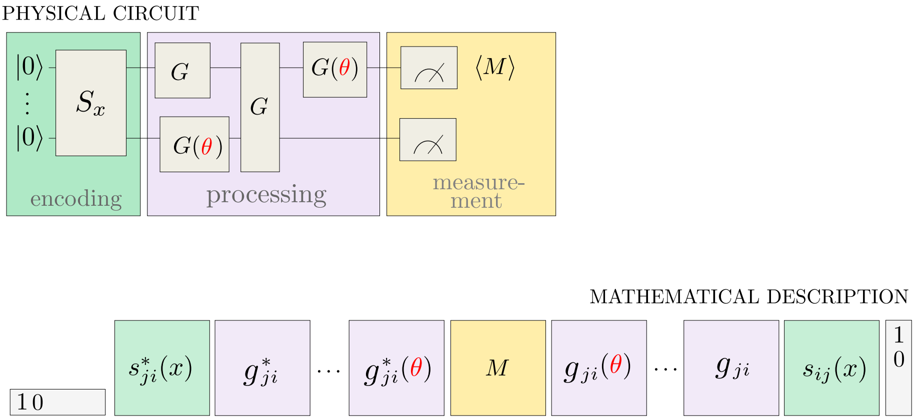
trainable,
composable
& differentiable



nonconvex
optimisation

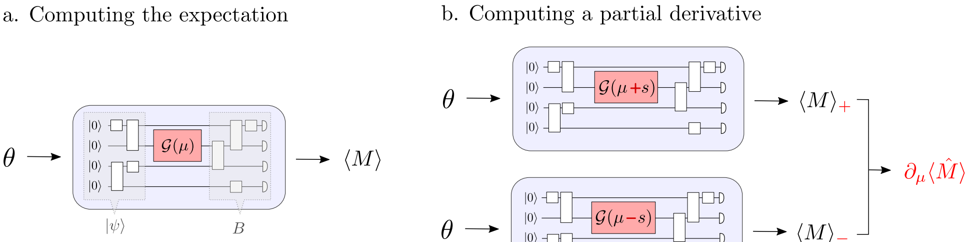
- gradient descent
- high performance hardware
- special purpose software

Qcircuits as trainable, composable & differentiable models.



Farhi & Neven 1802.06002, Schuld et al. 1804.00633

Qcircuits as trainable, composable & differentiable models.



Guerreschi & Smelyanskiy 1701.01450, Mitarai et al. 1803.00745, Schuld et al. 1811.11184

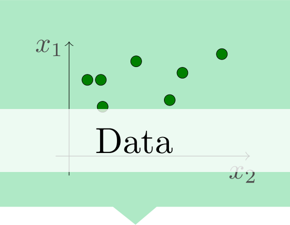
Qcircuits as trainable, composable & differentiable models.

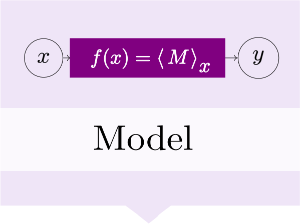
```python
import torch
from torch.autograd import Variable

data = torch.tensor([(0., 0.), (0.1, 0.1), (0.2, 0.2)])

def model(phi, x=None):
    return x*phi

def loss(a, b):
    return torch.abs(a - b) ** 2

def av_loss(phi):
    c = 0
    for x, y in data:
        c += loss(model(phi, x=x), y)
    return c

phi_ = Variable(torch.tensor(0.1), requires_grad=True)
opt = torch.optim.Adam([phi_], lr=0.02)

for i in range(5):
    l = av_loss(phi_)
    l.backward()
    opt.step()
```

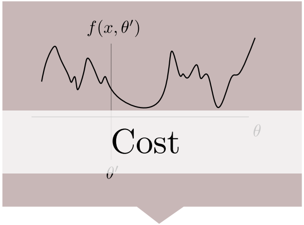
```python
from pennylane import *
import torch
from torch.autograd import Variable

data = [(0., 0.), (0.1, 0.1), (0.2, 0.2)]

dev = device('default.qubit', wires=2)

@qnode(dev, interface='torch')
def circuit(phi, x=None):
    templates.AngleEmbedding(features=[x], wires=[0])
    templates.BasicEntanglerLayers(weights=phi, wires=[0, 1])
    return expval(PauliZ(wires=[1]))

def loss(a, b):
    return torch.abs(a - b) ** 2

def av_loss(phi):
    c = 0
    for x, y in data:
        c += loss(circuit(phi, x=x), y)
    return c

phi_ = Variable(torch.tensor([[0.1, 0.2],[-0.5, 0.1]]), requires_grad=True)
opt = torch.optim.Adam([phi_], lr=0.02)

for i in range(5):
    l = av_loss(phi_)
    l.backward()
    opt.step()
```

pennylane.ai

We can train quantum circuits like neural nets.

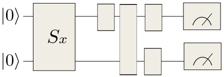






Big

trainable,
composable
& differentiable

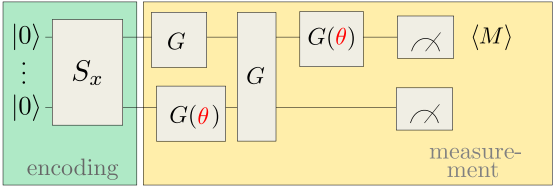


nonconvex optimisation

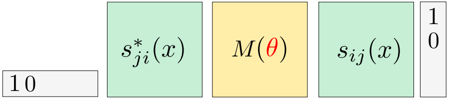
- gradient descent
- high performance hardware
- special purpose software

Quantum circuits are kernel methods.

PHYSICAL CIRCUIT

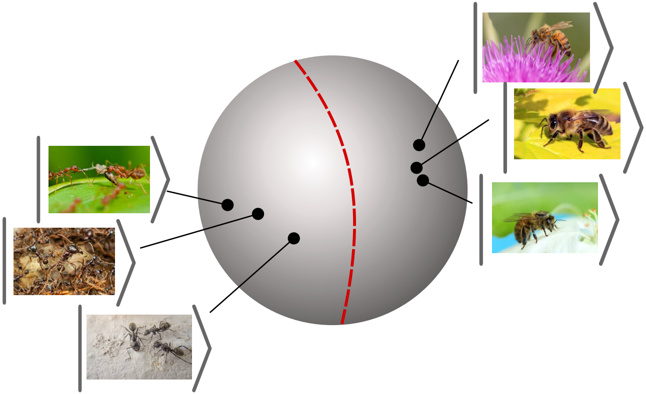


MATHEMATICAL DESCRIPTION



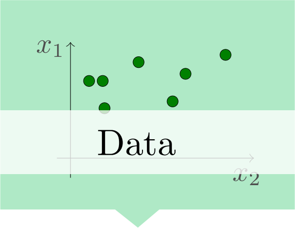
Schuld & Killoran 1803.07128, Havlicek et al. 1804.11326, Lloyd et al. 2001.03622

Quantum circuits are kernel methods.

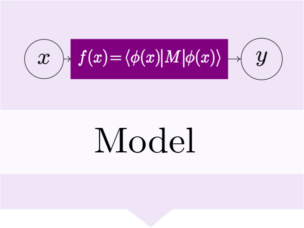


Lloyd et al. 2001.03622

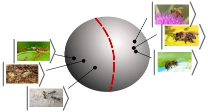
Quantum circuits are kernel methods.

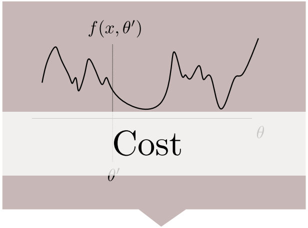


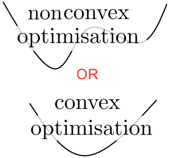
Big



trainable,
composable
& differentiable

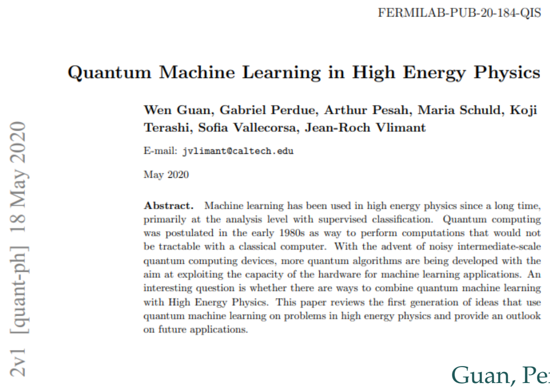






QML and HEP

A flavour of current work



FERMILAB-PUB-20-184-QIS

2v1 [quant-ph] 18 May 2020

Quantum Machine Learning in High Energy Physics

Wen Guan, Gabriel Perdue, Arthur Pesah, Maria Schuld, Koji Terashi, Sofia Vallecorsa, Jean-Roch Vlimant

E-mail: `jvlimant@caltech.edu`

May 2020

Abstract. Machine learning has been used in high energy physics since a long time, primarily at the analysis level with supervised classification. Quantum computing was postulated in the early 1980s as way to perform computations that would not be tractable with a classical computer. With the advent of noisy intermediate-scale quantum computing devices, more quantum algorithms are being developed with the aim at exploiting the capacity of the hardware for machine learning applications. An interesting question is whether there are ways to combine quantum machine learning with High Energy Physics. This paper reviews the first generation of ideas that use quantum machine learning on problems in high energy physics and provide an outlook on future applications.

Guan, Perdue, Pesah, Schuld, Terashi, Vallecorsa, Vlimant, *Quantum Machine Learning in High Energy Physics*, arxiv:2005.08582

A flavour of current work

Task: *Distinguish pair of photons created by Higgs decay from uncorrelated background events*
Features: *8 measurements taken on the di-photon system*
Quantum technology: *Quantum annealer (hardware)*
Quantum algorithm: *Use QUBO to find best (0/1) weights to combine 36 simple ML models ("weak learners")*

Mott, Job, Vlimant, Lidar, & Spiropulu (2017), Nature, 550(7676), 375-379

A flavour of current work

Task: *Particle track reconstruction*
Features: *Locations of hits + corresponding particles (TrackML challenge)*
Quantum technology: *Qubit-based quantum circuits (simulator)*
Quantum algorithm: *Represent hits as "tree-tensor network" quantum circuit and train gates in the network*

Tüysüz, Carminati, Demirköz, Dobos, Fracas, Novotny, ... & Vlimant (2020), arXiv:2003.08126

A flavour of current work

Task: *Higgs coupling to top quark pairs (ttH)*
Features: *45 input events (+ PCA)*
Quantum technology: *Qubit-based quantum circuits (simulator + hardware)*
Quantum algorithm: *Variational circuit (SVM interpretation)*

Chan, Guan, Sun, Wang, Wu, Zhou, ... & Di Meglio (2019), PoS, LeptonPhoton2019, 49

A flavour of current work

Task: *Classification of signal predicted in Supersymmetry*
Features: *SUSY data set in the UC Irvine Machine Learning Repositiory*
Quantum technology: *Qubit-based quantum circuits (simulator + hardware)*
Quantum algorithm: *Variational circuit (NN interpretation)*

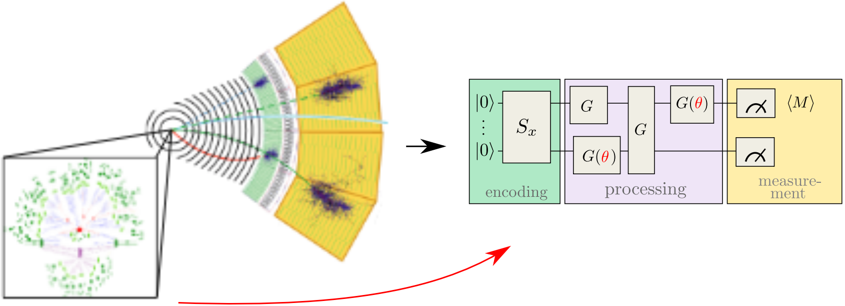
Terashi, Kaneda, Kishimoto, Saito, Sawada, & Tanaka (2020), arXiv:2002.09935

A critical comment

Why would you use QML in HEP?

- In ca 40 years' time you want to solve a linear algebra problem
- You have Fourier signals somewhere
- You can do information processing on your physical objects directly

Why would you do that?



Thank you!

www.pennylane.ai
www.xanadu.ai
@XanaduAI