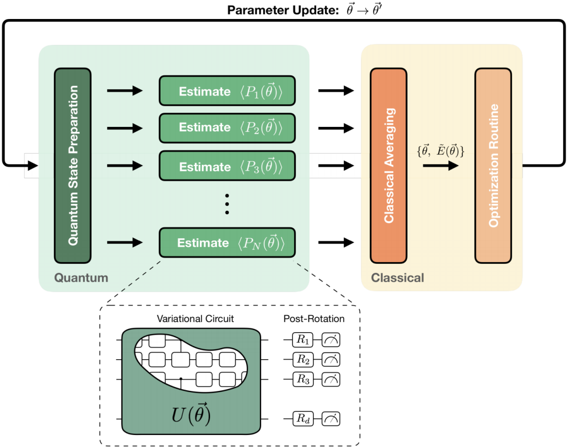
Near-term Algorithms for Quantum Machine Learning

Matthias Degroote
Work with Jhonathan Romero, Jonathan Olson, Lasse Bjørn Kristensen, Peter Wittek, Nikolaj Zinner, Alán Aspuru-Guzik

Conference on Quantum Information and Quantum Control-VIII 29/08/2019

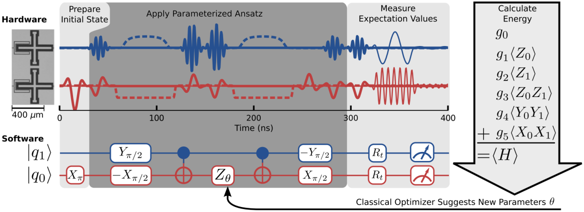


10.1038/ncomms5213
arXiv:1812.09976

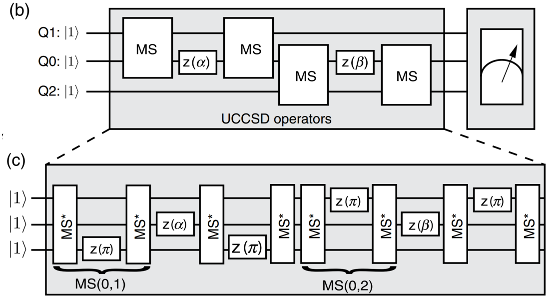
Comparison

- less operations
- more measurements
- optmization on classical computer

PhysRevX.6.031007



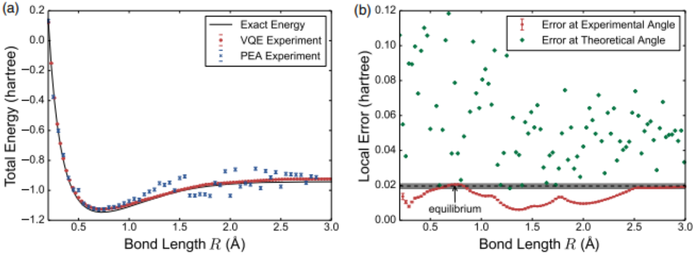
PhysRevX.8.031022



Transmon qubits

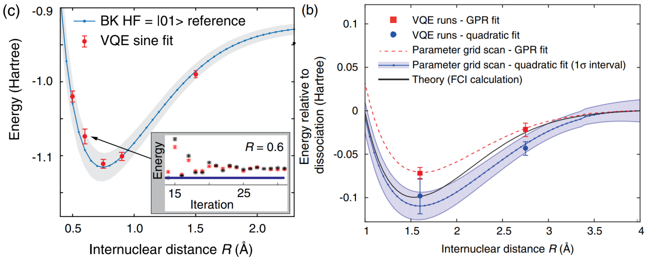
Trapped ions

PhysRevX.6.031007



H_2

PhysRevX.8.031022



H_2, LiH

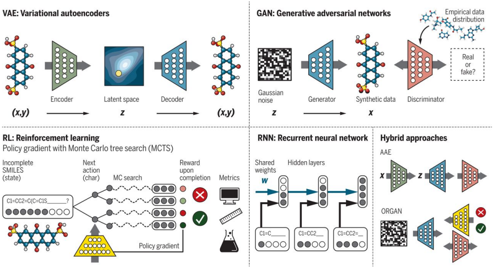
Some remarks

- What can you do with only the energy?
- How large to be useful?
- Which systems are interesting?

⇒ Inverse design

Challenge 1. The Designer Challenge. While the mission of the 20th century was related to providing answers to questions pertaining to properties of specific chemical structures, the questions of the 21st century revolve around the *inverse design* problem:[88−94] finding the best chemical structures that are associated with desired and requested properties. A potential solution for this challenge is the use of invertible models from machine learning such as generative models (GANs, autoencoders, ...)[48,89] or inverting molecules from families of Hamiltonians.[90−93]

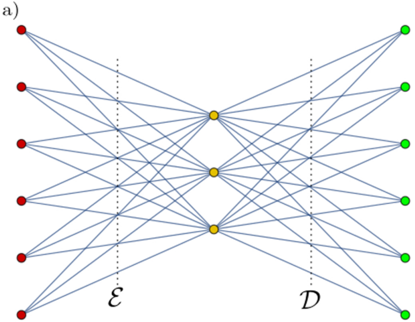
10.1021/acscentsci.7b00550



10.1021/acscentsci.7b00572
10.1126/science.aat2663

Can we reformulate this machine learning toolbox for near-term quantum devices?

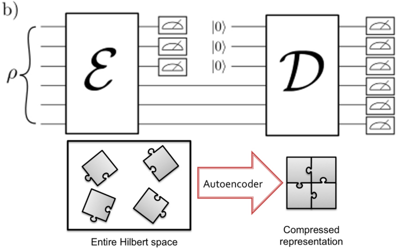
Classical Autoencoder

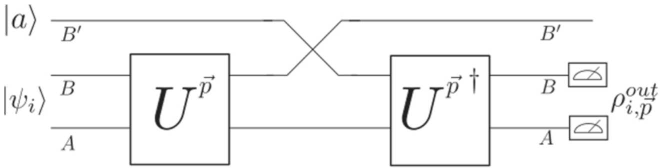


Quantum Variational Autoencoder

Jhonathan Romero

10.1088/2058-9565/aa8072





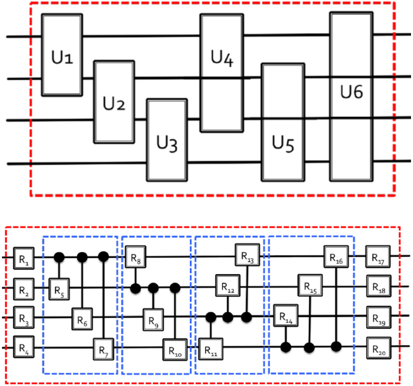
Fidelities

$$C_1 = \sum p_i F\left(|\psi_i\rangle_{AB}, \rho_{i,\bar{p}}^{out}\right)$$

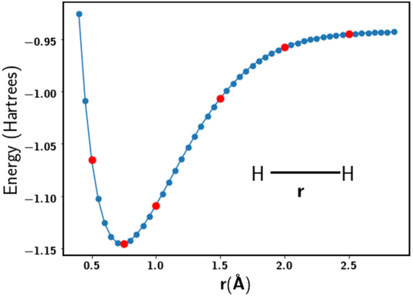
$$C_2 = \sum p_i F\left(\mathrm{Tr}_A\left[|\psi_i'\rangle\langle\psi_i'|_{AB}\right], |a\rangle_B\right)$$

- no input in swap
- guaranteed $C_1 \leq C_2$

Parametrized Unitaries

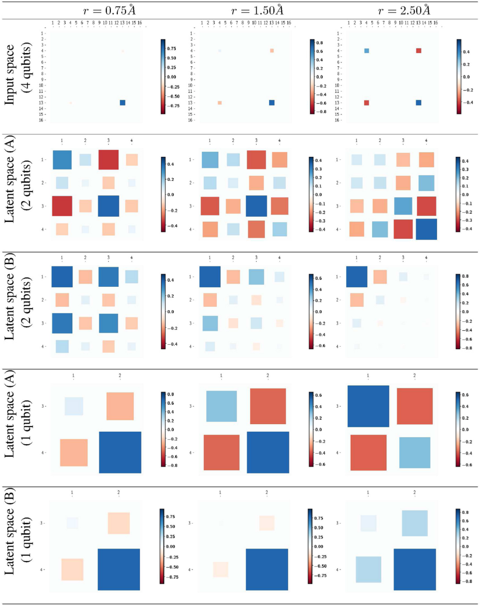


Training vs. Test



Circuit	Final size (# qubits)	Set	$-\log_{10}(1 - \mathcal{F})$ MAE[a]	$-\log_{10}$ Energy MAE[a] (Hartrees)
Model A	2	Training	6.96(6.82–7.17)	6.64(6.27–7.06)
	2	Testing	6.99(6.81–7.21)	6.76(6.18–7.10)
	1	Training	6.92(6.80–7.07)	6.60(6.23–7.05)
	1	Testing	6.96(6.77–7.08)	6.72(6.15–7.05)
Model B	2	Training	6.11(5.94–6.21)	6.00(5.78–6.21)
	2	Testing	6.07(5.91–6.21)	6.03(5.70–6.21)
	1	Training	3.95(3.53–5.24)	3.74(3.38–4.57)
	1	Testing	3.81(3.50–5.38)	3.62(3.35–4.65)

[a] MAE: Mean absolute error. Log chemical accuracy in Hartrees ≈ -2.80.


$r = 0.75 \mathring{A}$
$r = 1.50 \mathring{A}$
$r = 2.50 \mathring{A}$
Input space (4 qubits)
Latent space (A) (2 qubits)
Latent space (B) (2 qubits)
Latent space (A) (1 qubit)
Latent space (B) (1 qubit)

Latent Space exploration

- Identify information $\Rightarrow$ **Douglas Mendoza**
- State preparation
- Operations on reduced space

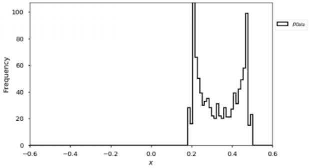
Quantum information applications

- Lossless compression as proxy for entanglement entropy
- Error correction scheme $\Rightarrow$ **Smik Patel**

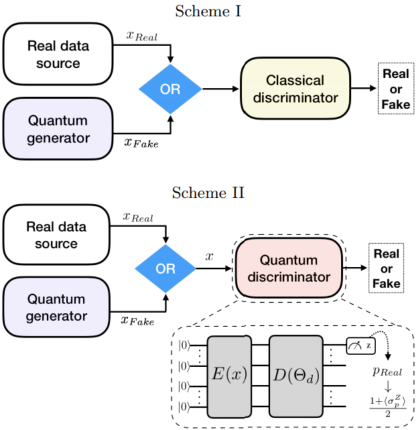
Jhonathan Romero
arXiv:1901.00848

- replicate classical distribution

Real data source:

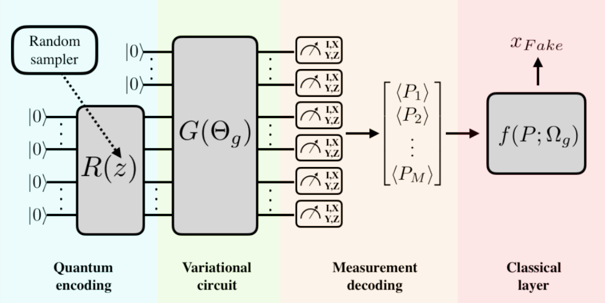


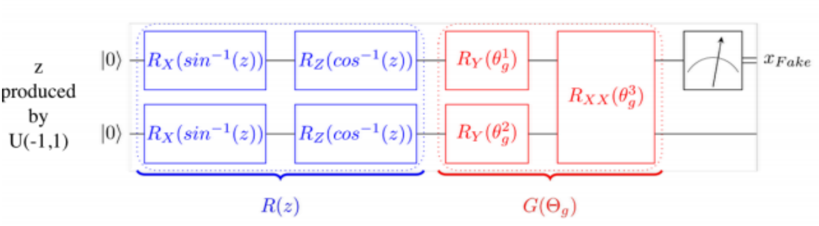
- adversial training

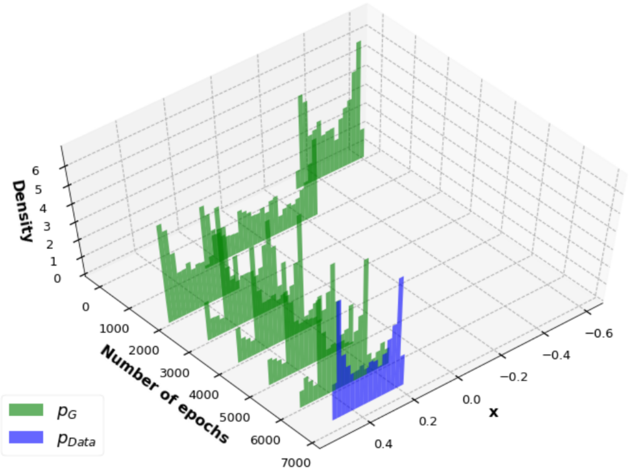


$$C = -\frac{1}{2}\mathbb{E}\left[F_D\left(x;\Theta_d\right)\right] -\frac{1}{2}\mathbb{E}\left[\log\left[1 - F_D\left(\left(F_G\left(z;\Theta_g,\Omega_g\right);\Theta_d\right)\right)\right]\right]$$

Generator circuit





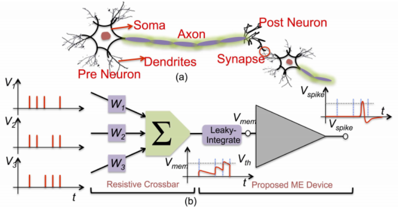


Currently looking at noisy simulations and implementations on a superconducting device ⇒ **Abhinav Anand**

Classical Spiking Neuron

- sparse connectivity
- spikes transfer information
- thresholding behavior
- temporal character

⇒ extremely efficient

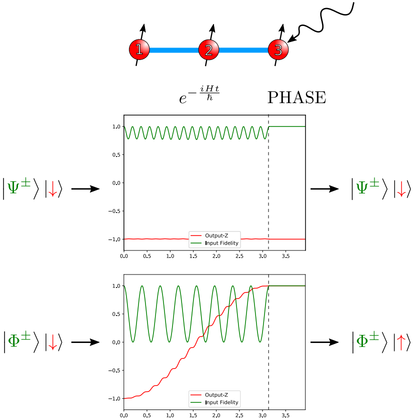


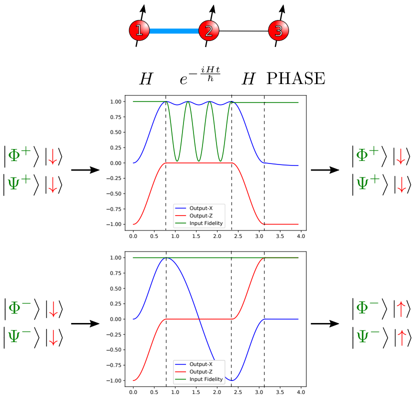
Quantum Spiking Neuron

Lasse Bjørn Kristensen
(Aarhus University, DK)
arXiv:1907.06269

- quantum neuron with temporal character
- not gate based
- discriminate Bell states without changing

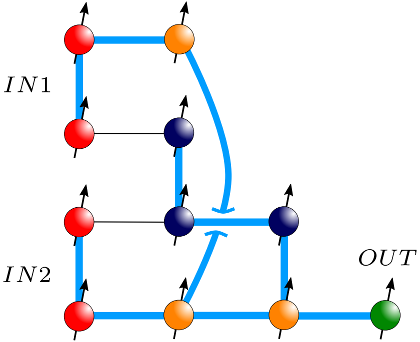
$$|\Phi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$$
$$|\Psi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)$$



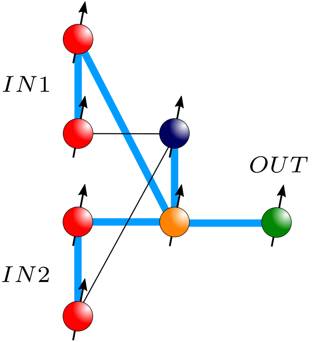


$$H = \frac{J}{2}\vec{\sigma_1}\cdot\vec{\sigma_2} + \beta\sigma_2^z\sigma_3^z + A\cos\left(\frac{2\beta}{\hbar}t\right)\sigma_3^x$$

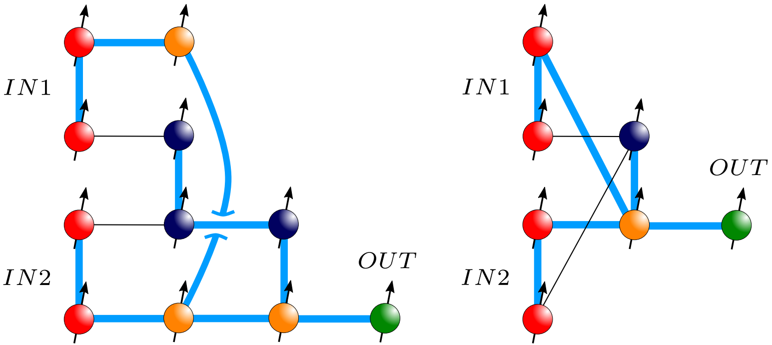
$$H = \frac{J}{2}\vec{\sigma_1}\cdot\vec{\sigma_2} + \delta\sigma_2^x\sigma_3^x + B\sigma_3^z$$



- 7 qubit overhead
- feed forward



- 3 qubit overhead
- not strictly feed forward
- complication in readout



- Measurement back-action $\Rightarrow$ non-trivial correlations in network
- Defines an expected likelihood kernel $\Rightarrow$ quantum kernel learning methods

$$P_{|\uparrow\rangle} = \sum_{0}^{3} |a_i|^2 \, |b_i|^2 \sim k\left(p, p'\right) = \int p(x)p'(x)\mathrm{d}x$$

Done:

- First demonstrations of non-trivial machine-learning functions

To do:

- Expand machine learning functions
- Apply concepts to new problems
- From simulation to experimental demonstration

Thank you for your attention!

Questions are welcome

Slides: `https://mfdgroot.github.io/`