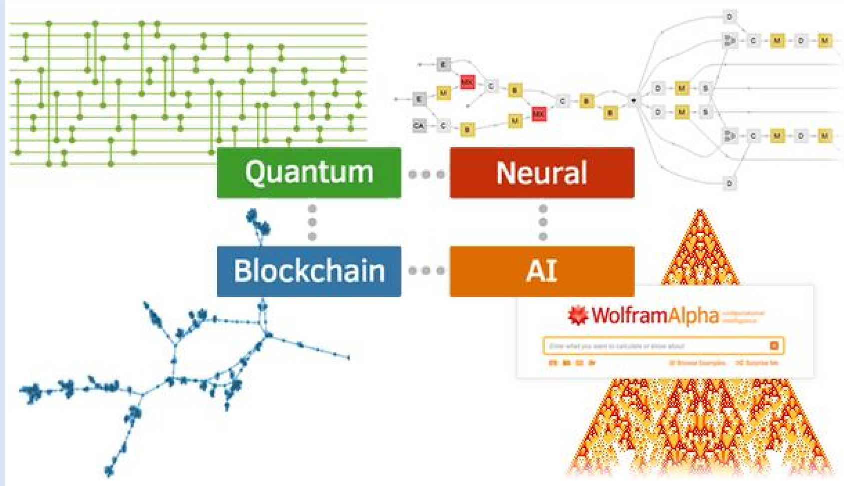
Quantumize Everything Everywhere All at Once

Qi Cao
Affiliation, University of California Berkeley

1. Introduction: The Rise of Quantum Neural Network



Source: https://writings.stephenwolfram.com/2018/04/buzzword-convergence-making-sense-of-quantum-neural-blockchain-ai/

➢**Why quantum neural network :**

Why do we use quantum neural network (QNN)? I believe one of the most important reasons is that the neural network and quantum variational algorithm exhibit good compatibility.

Although many quantum algorithms exhibit exponential speedup to certain classes of problems in theory, the quantum circuits of many these methods are too large to be executed on these noisy intermediate-scale quantum (NISQ) devices [1] due to the lack of quantum error correction.

QNN is based on the method of variational quantum circuit, which has already shown potentially good adaptation to NISQ systems [2]. This means that QNN can potentially be used in real-life system with quantum advantages. **QNN can be everything everywhere all at once!**

2. Ability: Quantum Neural Network Can Be Everything

➢**Universal approximation for Neural Network:**

- ❑ Neural network is the most popular machine learning model now, which shows strong reprsentative ability and mysterious inner mechanism.
- ❑ Cybenko Theorem: Two-layer neural network can approximate any function.

Theorem: Cybenko (1989)

If $\sigma(z)$ is a non-constant sigmoid, then for any continuous function $f(x)$ on $[0,1]^p$, there exist parameter values $H, \alpha_h \in \mathbb{R}, w_h \in \mathbb{R}^n, w_0 \in \mathbb{R}$, such that the two-layer network

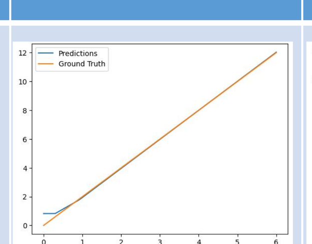
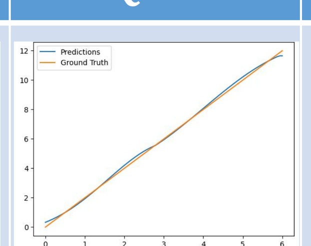
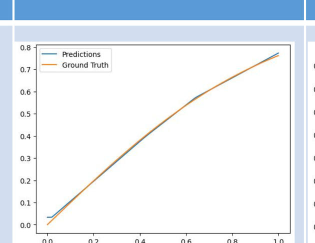
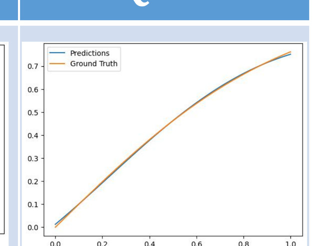
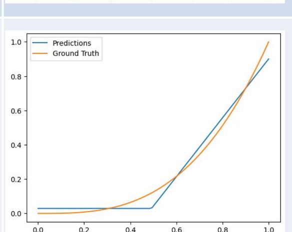
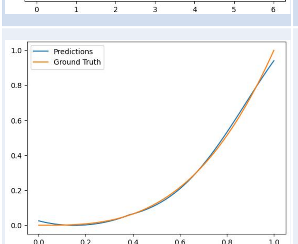
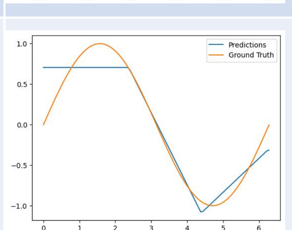
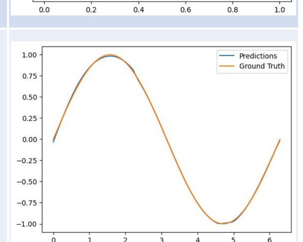
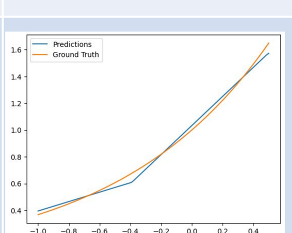
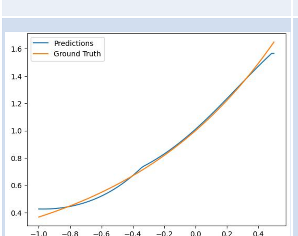
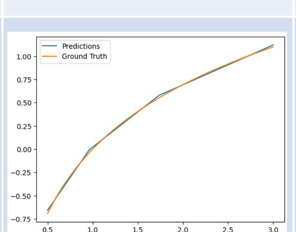
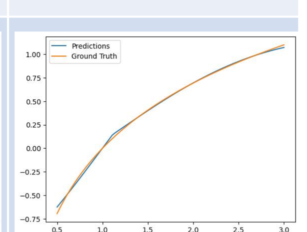
$$\hat{f}(x) = \sum_{h=1}^{H} \alpha_h \sigma(\langle x, w_h \rangle - w_0)$$

approximates $f(x)$ with arbitrary pre-defined accuracy ε:

$$|\hat{f}(x) - f(x)| < \varepsilon, \text{ for all } x \in [0,1]^p.$$

➢**Universal QNN experiment:**

- ❑ Can VQC-based QNN approximate any function?
- ❑ I designed simulation experiment to test QNN's capability, and compare it with traditional neural network.
- ❑ Simulate with the help of Pennylane and Pytorch.

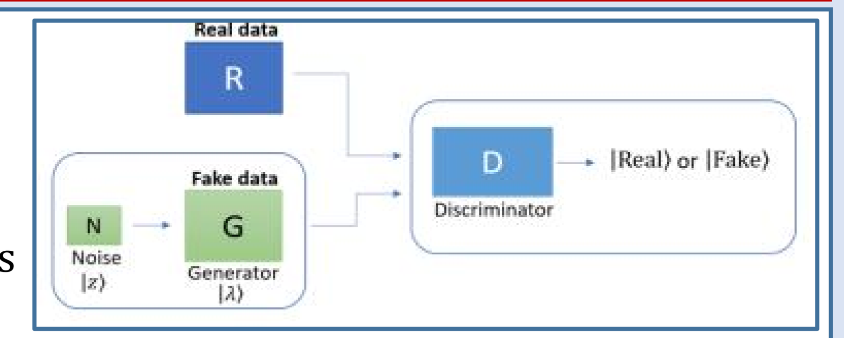
Function	Traditional	QNN	Function	Traditional	QNN
f(x) = 2x			f(x) = tanh(x)		
f(x) = x³			f(x) = sin(x)		
f(x) = exp(x)			f(x) = log(x)		

- ❑ Both QNN and NN are trained 500 epoches with the same structure and dataset.
- ❑ QNN has even better reprsentative ability than traditional neural network.
- ❑ QNN is very good at approximating nonlinear function, the lines are smoother.

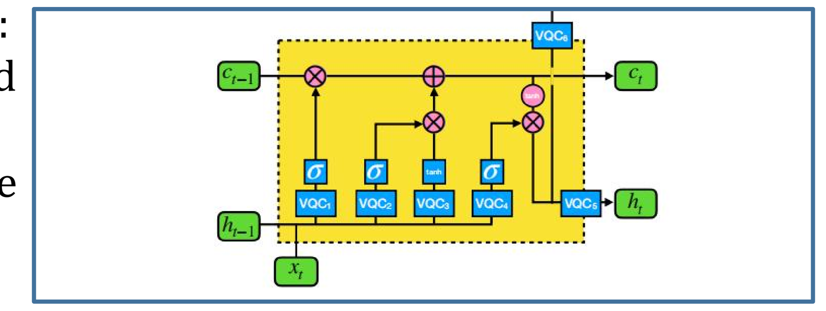
3. Application: Quantum Neural Network Can Be Everywhere

➢**Many applications of QNN:**

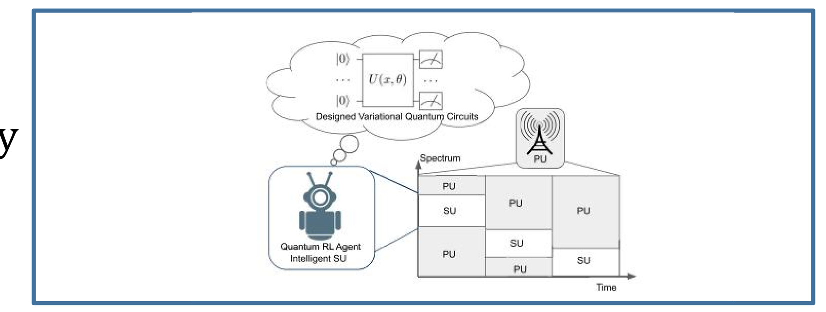
- ❑ Generative models: [3] gives an example of a simple practical circuit ansatz and perform a simple numerical experiment to demonstrate that quantum generative adversarial networks can be trained successfully.



- ❑ Quantum NLP (Natural Language Processing): LSTM (Long-short time memory) is a standard model for NLP. In [4], QLSTM is proposed and shown to be more effective than LSTM in three tasks, both in the convergence and parameter number.



- ❑ Quantum Reinforcement Learning: [5] uses VQC for deep Q-value function of reinforcement learning with experience replay and target network. Besides, their variational quantum circuits can be deployed in many near-term NISQ machines.



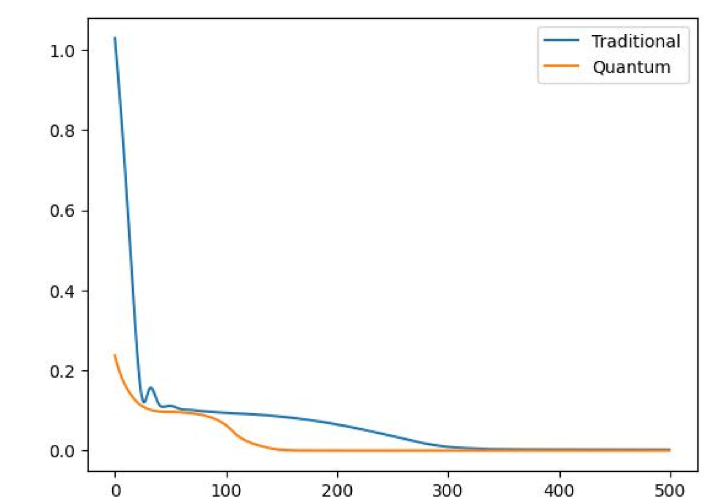
4. Efficientcy: Quantum Neural Network Can Be All At Once

➢ **Parameter Comparasion**

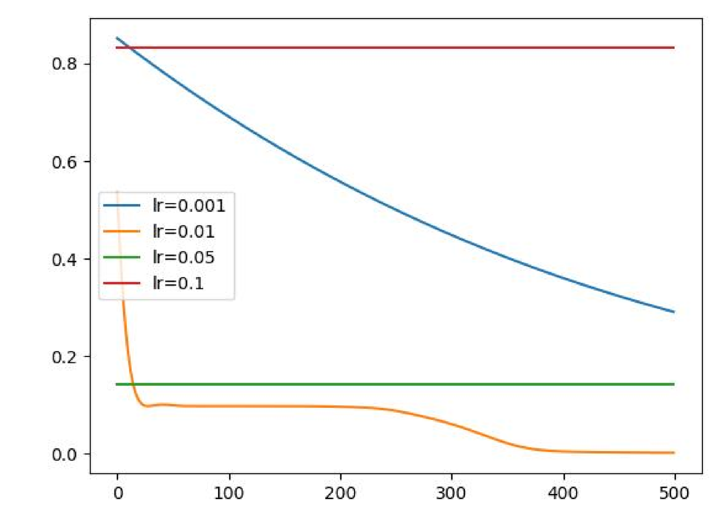
- ❑ Traditional Neural Network in the experiment have 54 parameters.
- ❑ QNN have less parameters (i.e. 32), but shows greater representation ability.

➢ **Training Comparasion**

- ❑ I compared the loss curve with traditional, and I tried different learning rate of QNN.



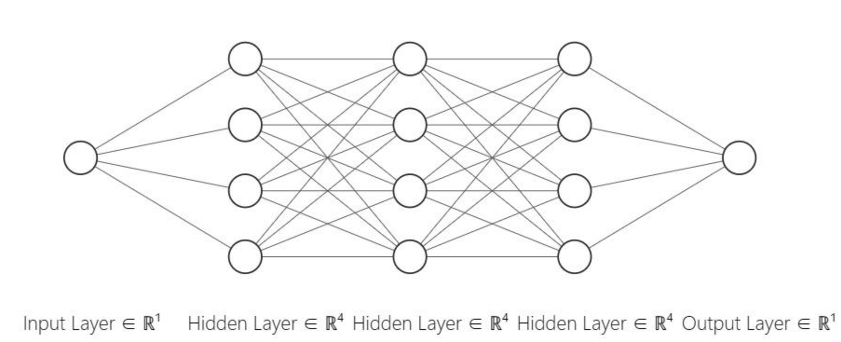
Loss Curve



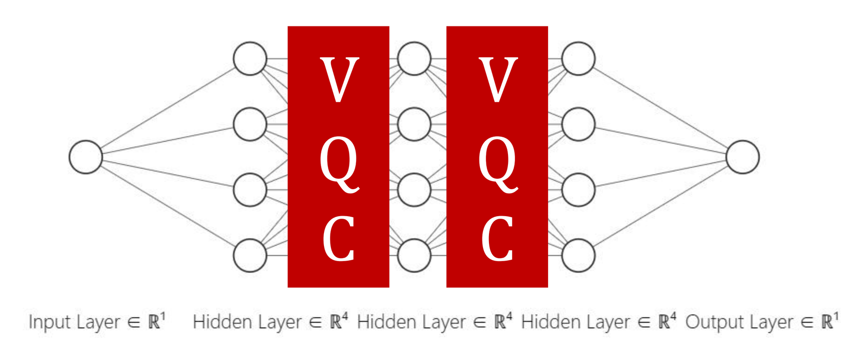
Learning Rate Tuning

5. Method: My QNN and Quantum Gradient Decent

➢ **My QNN Architecture**



Traditional Neural Network



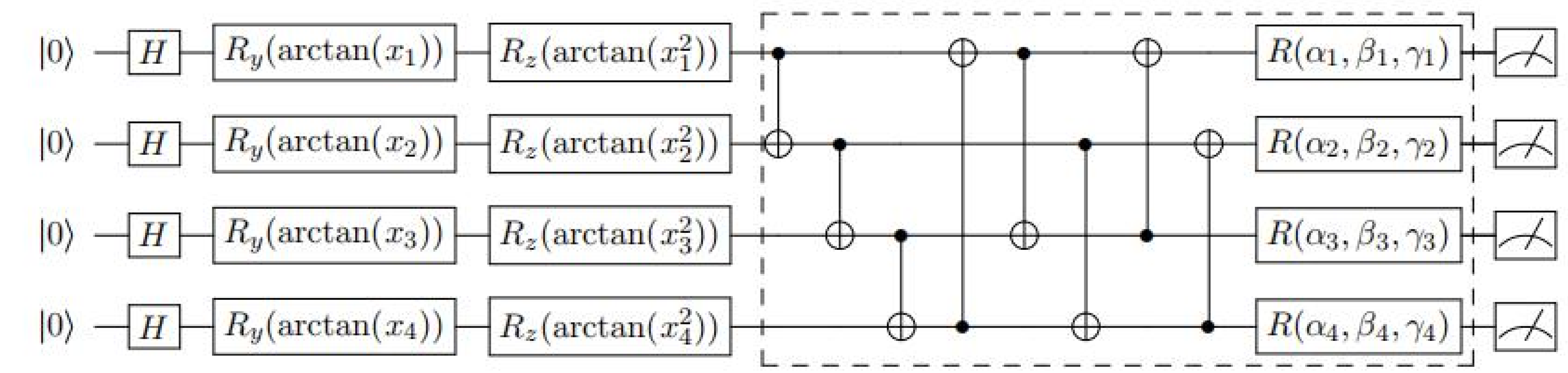
Quantum Hybrid Neural Network

➢ **Quantum Optimization and Gradient Decent**

- ❑ Training Neural Network should do gradient decent to optimize the parameters.
- ❑ How to compute the gradient?
- ❑ Parameter shift method [6] is used to derive the gradient of QNN block. According to [7], the analytical gradient of QNN block is:

$$\nabla_{\theta_i} f(x;\theta_i) = \frac{1}{2}\left[f\left(x;\theta_i + \frac{\pi}{2}\right) - f\left(x;\theta_i - \frac{\pi}{2}\right)\right]$$

7. Theoretic Background: Variational Quantum Circuit



➢ **Data Encoding Layer:**

- ❑ Any classical data to be processed with a quantum circuit needs to be encoded into its quantum state. A general N-qubit quantum state can be represented as:

$$|\psi\rangle = \sum_{(q_1,q_2,\cdots,q_N)\in\{0,1\}} c_{q_1,q_2,\cdots,q_N} |q_1\rangle \otimes |q_2\rangle \otimes \cdots \otimes |q_N\rangle$$

- ❑ Input x is encoded into rotation block.

➢ **Variational Layer**

- ❑ The encoded classical data, which is now a quantum state, will then go through a series of unitary operations.
- ❑ The 3 rotation angles {αi, βi, γi} along the axes x, y, and z, respectively, in the single-qubit rotation gates {Ri = R(αi, βi, γi)} are not fixed in advance; rather, they are to be updated in the iterative optimization process based on a gradient descent method.

➢ **Quantum Measurement Layer**

- ❑ The end of every VQC block is a quantum measurement layer.
- ❑ We consider the expectation values of every qubit by measuring in the computational basis.
- ❑ With quantum simulation software such as PennyLane [8] and IBM Qiskit [9], it can be calculated numerically on a classical computer, whereas with real quantum computers, such values are statistically estimated through repeated measurements, which should be in theory close to the value obtained from simulation in the zero-noise limit.

8. Discussion

➢**Will QNN surpass Neural Network in the future?**

- ❑ QNNs are expected to excel in certain types of computations, such as solving specific optimization problems or simulating quantum systems. Classical NNs, on the other hand, are well-suited for a wider range of tasks, including image recognition and so on.
- ❑ The success of QNNs might depend on the specific problem or task at hand. Some problems may benefit from quantum processing, while others may not see as much improvement.

References:

[1] J. Preskill, "Quantum computing in the nisq era and beyond," Quantum, vol. 2, p. 79, 2018.
[2] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, "The expressive power of parameterized quantum circuits," arXiv preprint arXiv:1810.11922, 2018.
[3] Dallaire-Demers, Pierre-Luc, and Nathan Killoran. "Quantum generative adversarial networks." Physical Review A 98.1 (2018): 012324.
[4] Chen, Samuel Yen-Chi, Shinjae Yoo, and Yao-Lung L. Fang. "Quantum long short-term memory." ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022.
[5] Chen, Samuel Yen-Chi, et al. "Variational quantum circuits for deep reinforcement learning." IEEE Access 8 (2020): 141007-141024.
[6] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," Physical Review A, vol. 99, no. 3, p. 032331, 2019.
[7] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," Physical Review A, vol. 98, no. 3, p. 032309, 2018.
[8] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, C. Blank, K. McKiernan, and N. Killoran, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," arXiv preprint arXiv:1811.04968, 2018.
[9] H. Abraham et al., "Qiskit: An open-source framework for quantum computing," 2019.