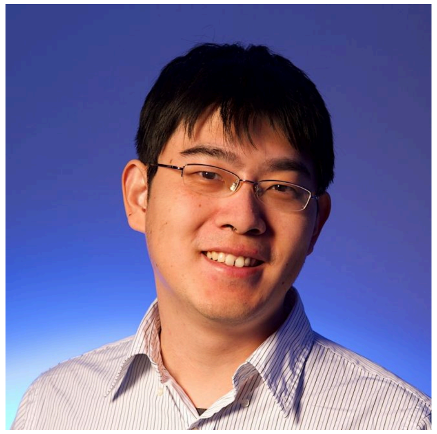
Quantum Variational Methods for Quantum Applications



Shouvanik Chakrabarti   Xuchen You   **Xiaodi Wu**

ICCAD 2021: Special Session on Quantum Machine Learning

based on results published in NeurIPS 19, PLDI 20, ICML 21

UNIVERSITY OF MARYLAND

JOINT CENTER FOR QUANTUM INFORMATION AND COMPUTER SCIENCE

Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: ~ classically *parameterized* quantum circuits

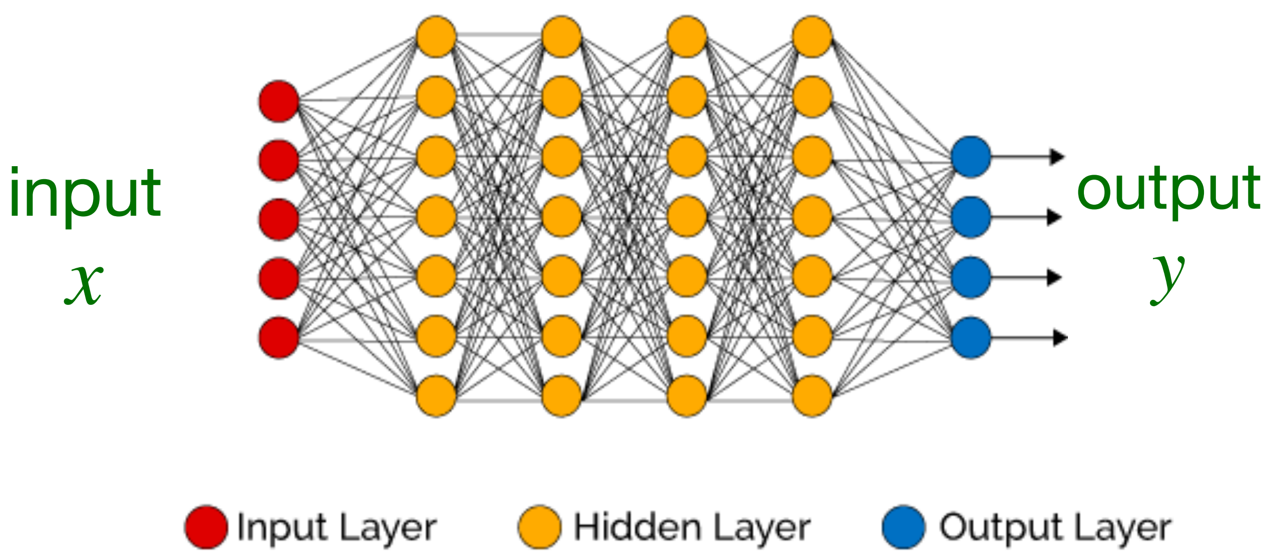
- share a lot of similarities w/ classical NNs, also with quantum features

Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: ~ classically *parameterized* quantum circuits
- share a lot of similarities w/ classical NNs, also with quantum features

Classical Neural Networks (CNNs)



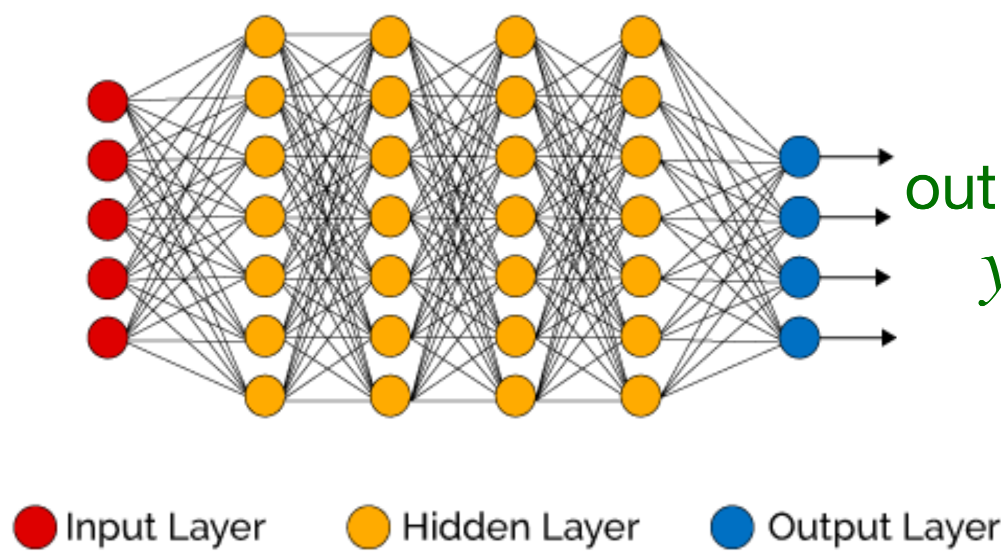
Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: ~ classically *parameterized* quantum circuits
- share a lot of similarities w/ classical NNs, also with quantum features

Classical Neural Networks (CNNs)

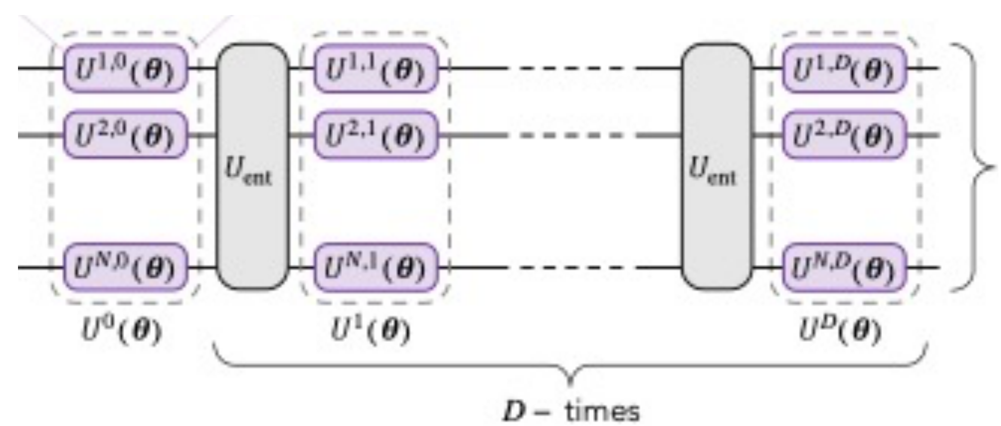
input x



output y

input x

quantum ρ_x



output y

Measurement

Variational Quantum Circuits (VQCs)

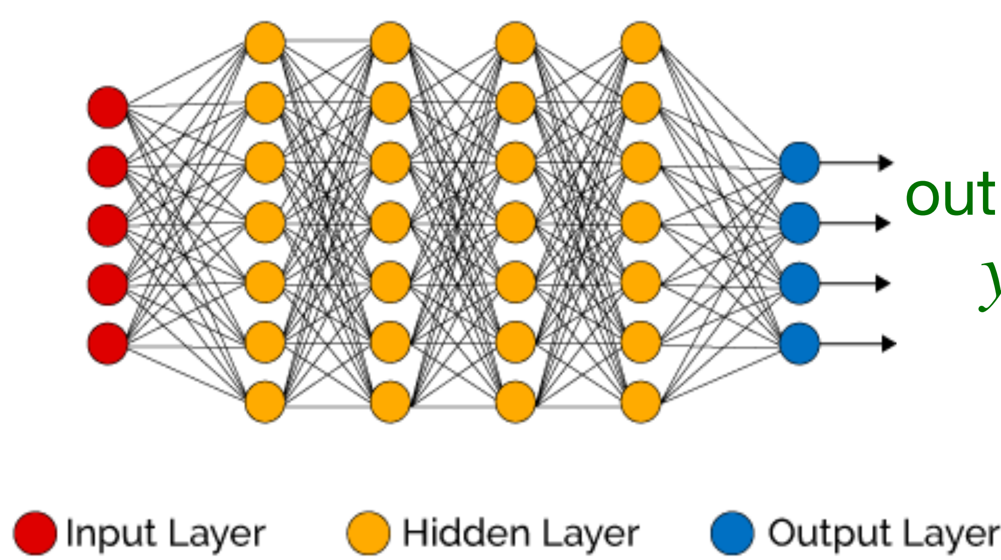
Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: ~ classically *parameterized* quantum circuits
- share a lot of similarities w/ classical NNs, also with quantum features

Classical Neural Networks (CNNs)

input x



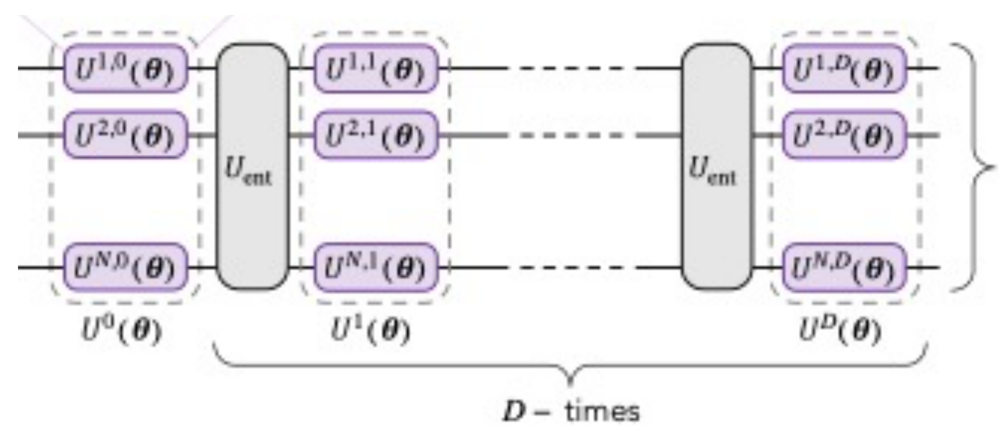
output y

Replace $x \longrightarrow y$ (classical) by

$x \rightarrow \rho_x \rightarrow y$ (quantum) w/ potential speedups

input x

quantum ρ_x



output y

Measurement

Variational Quantum Circuits (VQCs)

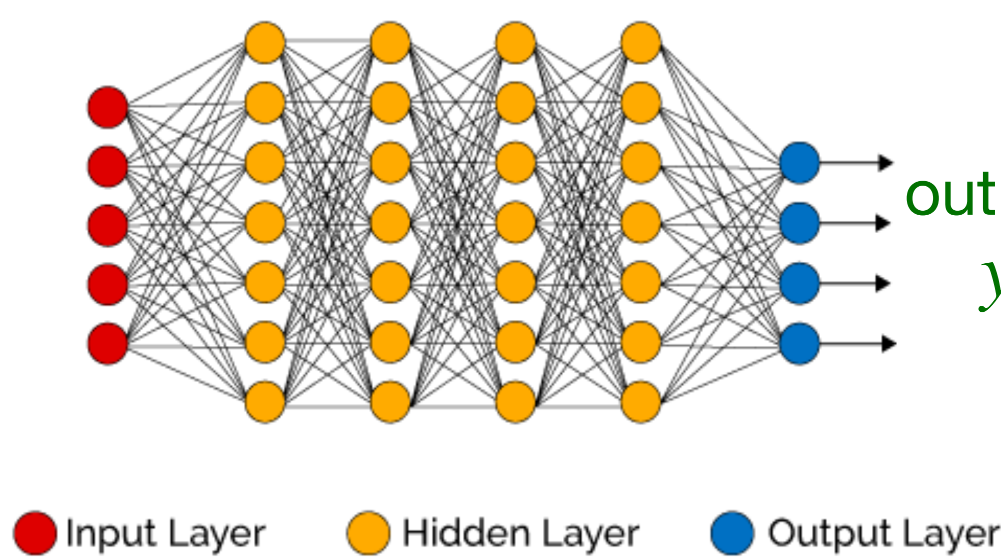
Quantum Variational Methods: Short & Long Term

Motivation: (*short-term*) NISQ with **resource** constraints
or (*long-term*) consider the **paradigm shift** made by the deep learning

Quantum Variational Methods: ~ classically *parameterized* quantum circuits
- share a lot of similarities w/ classical NNs, also with quantum features

Classical Neural Networks (CNNs)

input
x

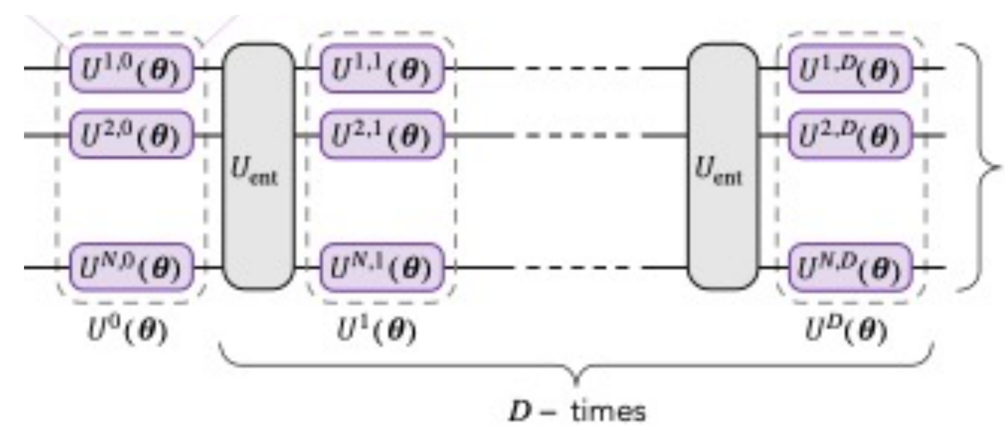


output
y

Replace $x \longrightarrow y$ (classical) by

$x \rightarrow \rho_x \rightarrow y$ (quantum) w/ potential speedups

input
x
quantum
ρ_x



output
y
Measurement

Variational Quantum Circuits (VQCs)

Promising for:
(1) quantum physics related problems
(2) computational tasks with structures that can be leveraged by quantum mechanics

Quantum Variational Methods: Theory-guided Empirical Study

Unlike classical NNs, **empirical study** of q. variational method is limited:

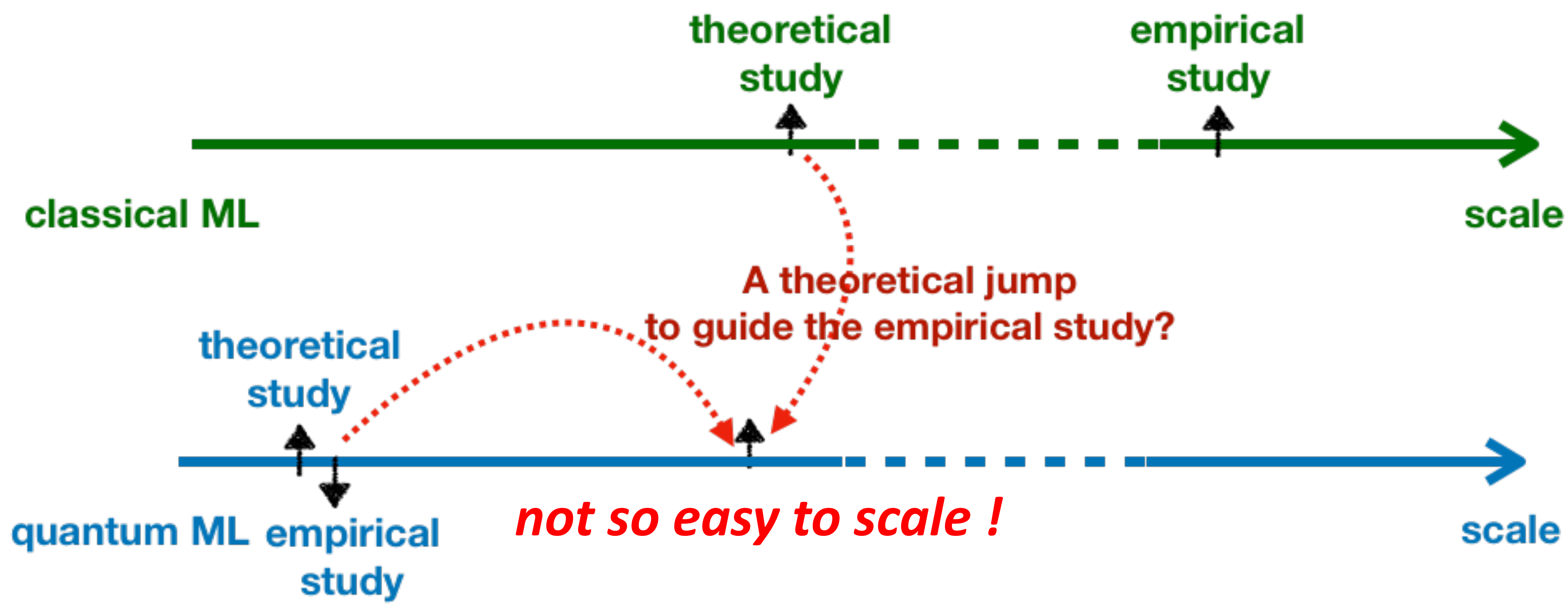
- due to **exponential** cost in classical simulation of parameterized q. Circuits
- due to **noisy** and **size-limited** available quantum machines (NISQ)

Quantum Variational Methods: Theory-guided Empirical Study

Unlike classical NNs, **empirical study** of q. variational method is limited:

- due to **exponential** cost in classical simulation of parameterized q. Circuits
- due to **noisy** and **size-limited** available quantum machines (NISQ)

Unboxing Techniques from Machine Learning

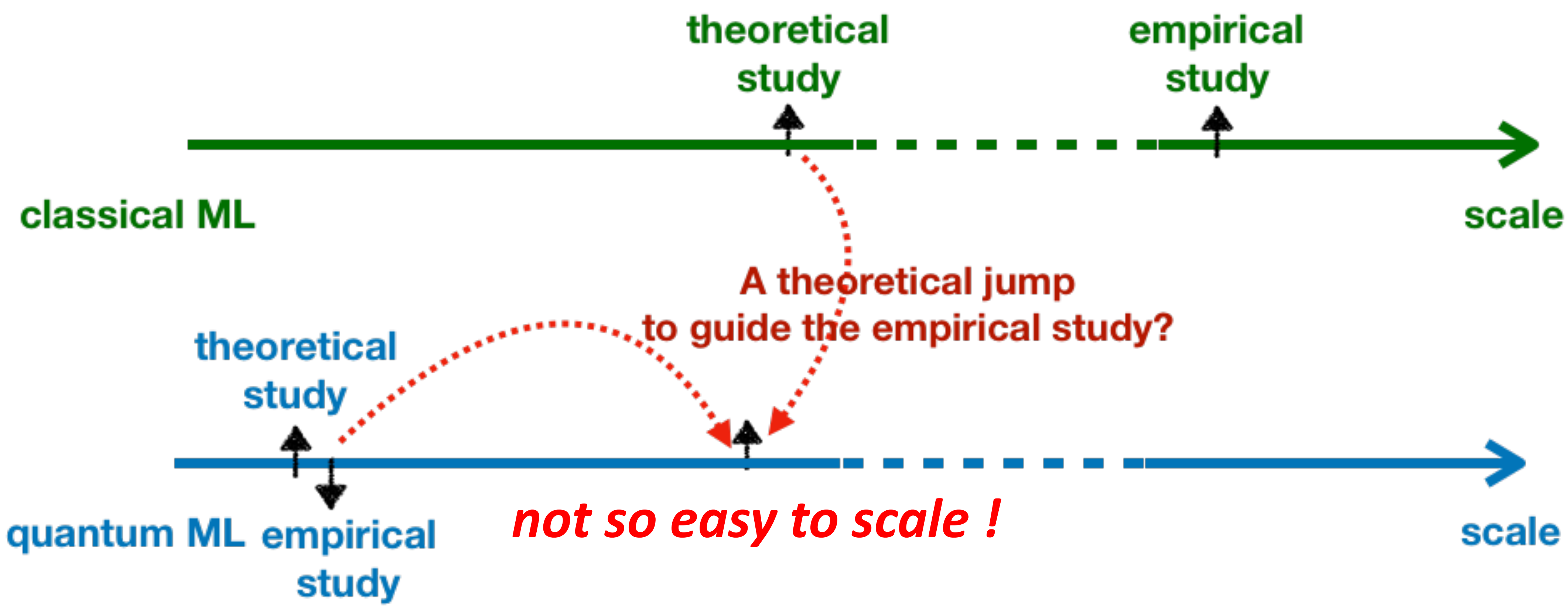


Quantum Variational Methods: Theory-guided Empirical Study

Unlike classical NNs, **empirical study** of q. variational method is limited:

- due to **exponential** cost in classical simulation of parameterized q. Circuits
- due to **noisy** and **size-limited** available quantum machines (NISQ)

Unboxing Techniques from Machine Learning



In particular:

We focus on how to **train** quantum variational models efficiently !!

loss function design + *variational model* design

Landscape in Training Quantum Variational Methods

VQCs could be very hard in training:

(a) random initialization => zero gradients for slightly larger VQCs [McClean et al., Nat Com, 9(1):4812, 2018]

(b) bad numerical landscape e.g., arXiv:1903.02537

Candidate training strategy of VQCs in special cases

(a) QAOA for certain classes of instances,

(b) extremely over-parameterized circuits, e.g., arXiv:2001.11897, arXiv:1905.12134

Landscape in Training Quantum Variational Methods

VQCs could be very hard in training:

(a) random initialization => zero gradients for slightly larger VQCs [McClean et al., Nat Com, 9(1):4812, 2018]

(b) bad numerical landscape e.g., arXiv:1903.02537

Candidate training strategy of VQCs in special cases

(a) QAOA for certain classes of instances,

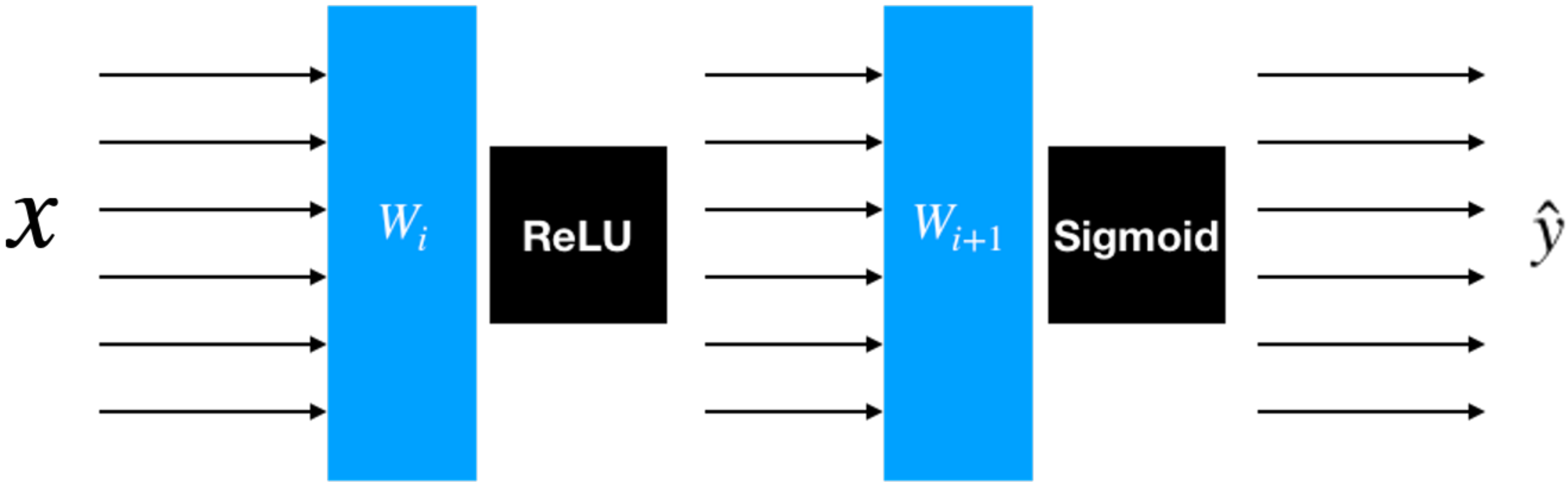
(b) extremely over-parameterized circuits, e.g., arXiv:2001.11897, arXiv:1905.12134

[You & Wu, ICML 2021] in the context of supervised learning

Theorem 1 (Construction) For *almost all* p-parameter d-dimensional under-parameterized QNN designs $(\mathbf{U}, \mathbf{M})$ with $p = O(\log d)$, there *exists* a hard dataset $\mathcal{S}$ such that the loss function $L(\boldsymbol{\theta}; \mathcal{S})$ has $2^p - 1$ local minima within each period.

Theorem 2 (Upper bound) The number of strict local minima for p-parameter QNNs are bounded by $(4p)^p$ for non-degenerated cases.

Classical Neural-Networks vs VQCs



Input: x , Output: $\hat{y}$, parameters: W, for data point (x,y), Loss = l (y, $\hat{y}$(x, W))

The loss function over m training data points

$$L(W) = \sum_{i}^{m} l(y_i, \hat{y}_i(W, x_i))$$

Square Loss

$$l(y, \hat{y}) = (y - \hat{y})^2$$

[Kawaguchi, NIPS 16] No bad local optima for (deep) **linear** networks.

[Auer et al, NIPS 95] Exponentially many local optima for 1 neuron.

Classical Neural-Networks vs VQCs

$$\rho_x \longrightarrow \boxed{e^{i\theta_1 X}} \longrightarrow \boxed{e^{i\theta_2 Z}} \longrightarrow \hat{y}$$

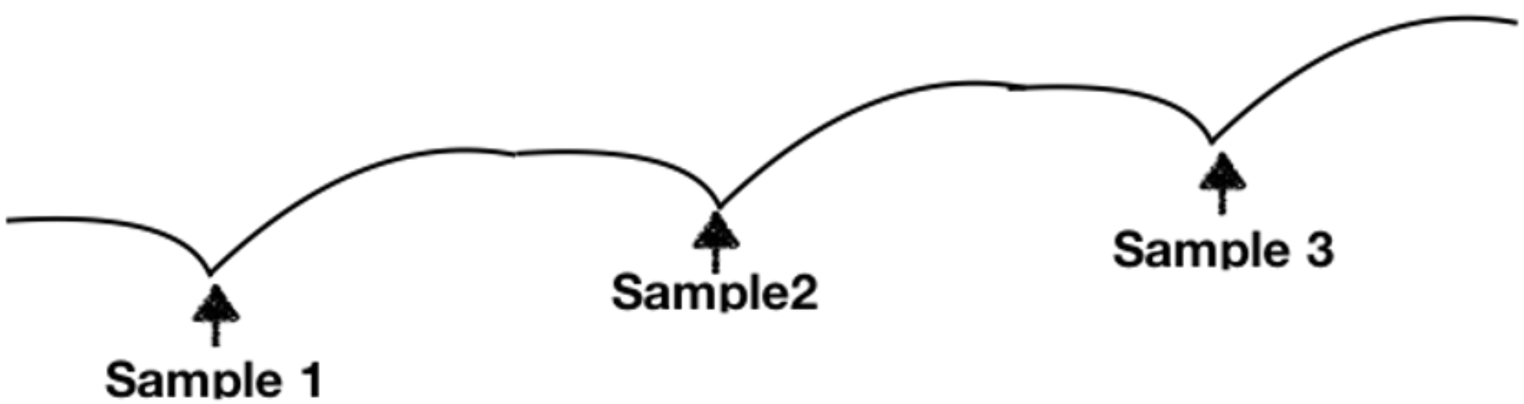
No ReLU/Sigmoid. Very linear with nice input encoding and output measurement except the Pauli-rotation gates. *Does it have a nice landscape?*

Classical Neural-Networks vs VQCs

No ReLU/Sigmoid. Very linear with nice input encoding and output measurement except the Pauli-rotation gates. *Does it have a nice landscape?*

Unfortunately, no! Quantum has another way to create local optima!

Non-linearity => Classical Local Optima



Interference => Q. Local Optima

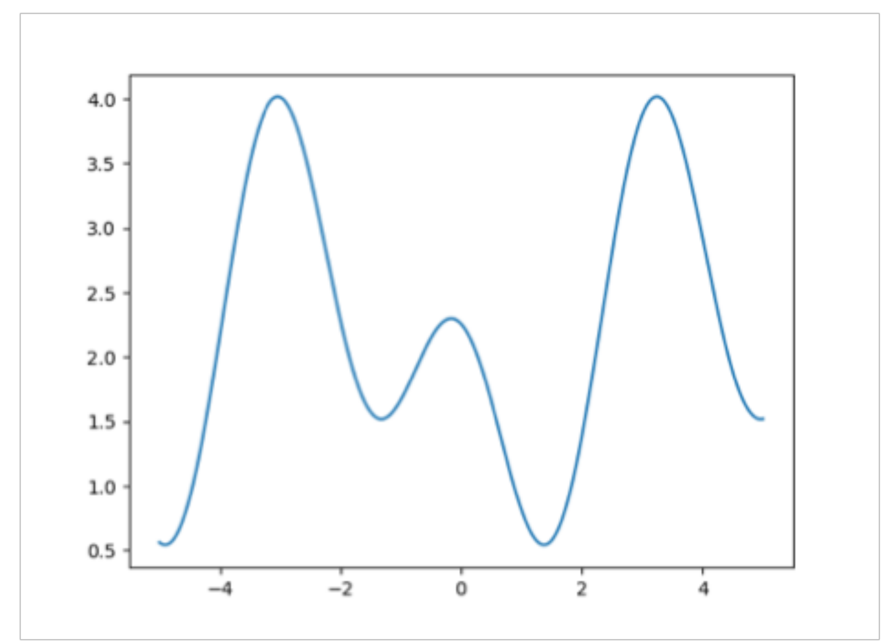
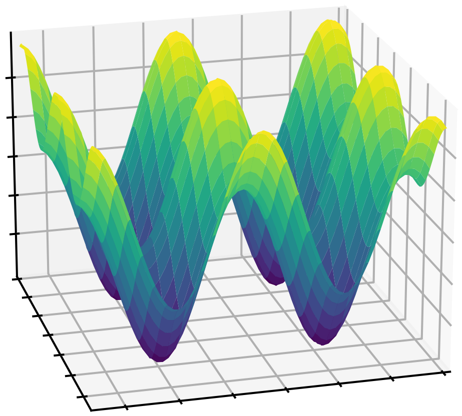


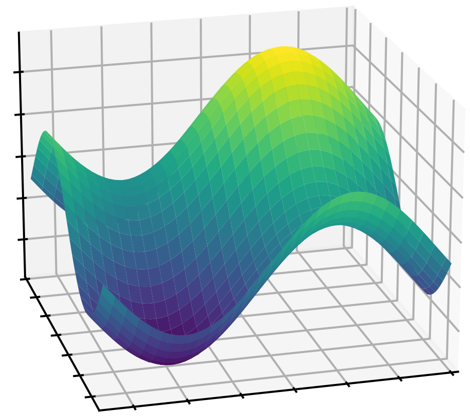
Figure 1: 1 qubit with spurious local minima

Construction of Hard Datasets

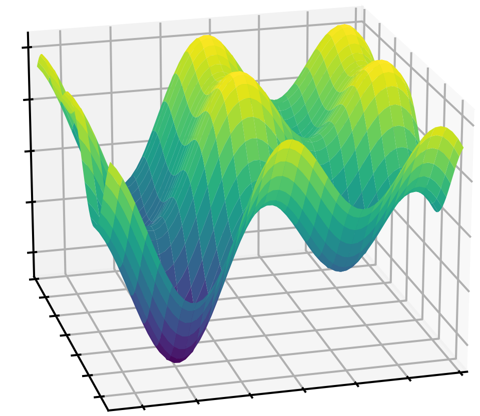
We use the classical idea of symmetry breaking to construct hard datasets



$L(\boldsymbol{\theta}; \mathcal{S}_0)$



$L(\boldsymbol{\theta}; \mathcal{S}_1)$



$L(\boldsymbol{\theta}; \mathcal{S}_0 \cup \mathcal{S}_1)$

- For classical neural networks: permutation of hidden neurons

- For quantum neural networks: the existence ($\mathcal{S}_0$) and breaking ($\mathcal{S}_1$) of the $\frac{\pi}{2}$-translational invariance in parameterization

- Expanding the observable in the Heinsenberg's picture

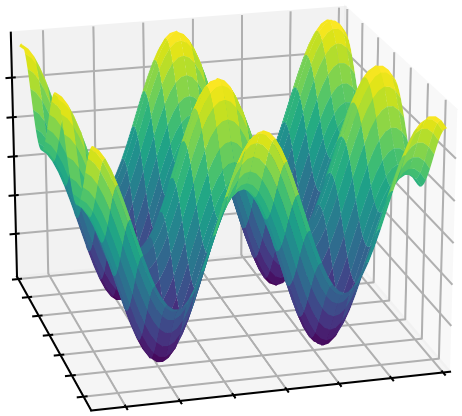
$$\mathbf{M}(\boldsymbol{\theta}) := \mathbf{U}^\dagger(\boldsymbol{\theta})\mathbf{M}\mathbf{U}(\boldsymbol{\theta}) = \sum_{\boldsymbol{\xi} \in \{0,1,2\}^p} \Phi_{\boldsymbol{\xi}}(\mathbf{M}) \prod_{l:\xi_l=1} \cos 2\theta_l \prod_{l':\xi_{l'}=2} \sin 2\theta_{l'}$$

  with $\Phi_{\boldsymbol{\xi}}(\mathbf{M})$ being Hermitians, the form of which depending on the QNN design.

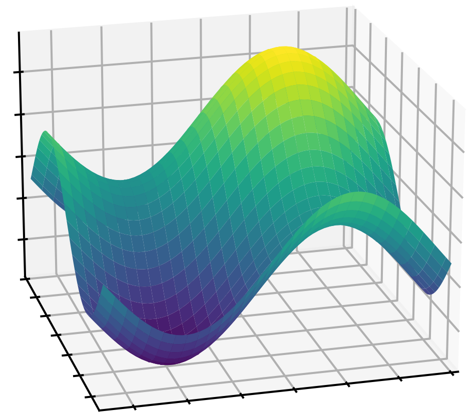
- Datasets $\mathcal{S}_0$ and $\mathcal{S}_1$ can be constructed by solving a linear system given that $\{\Phi_{\boldsymbol{\xi}}(\mathbf{M})\}_{\boldsymbol{\xi} \in \{0,1,2\}^p, \boldsymbol{\xi} \neq \mathbf{0}}$ forms a linearly independent set (*L.D. condition*)

Construction of Hard Datasets

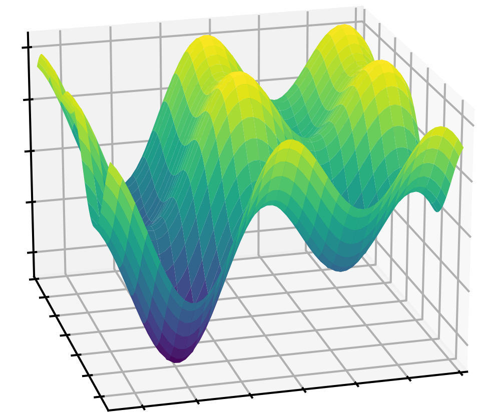
We use the classical idea of symmetry breaking to construct hard datasets



$L(\boldsymbol{\theta}; \mathcal{S}_0)$



$L(\boldsymbol{\theta}; \mathcal{S}_1)$



$L(\boldsymbol{\theta}; \mathcal{S}_0 \cup \mathcal{S}_1)$

- For classical neural networks: permutation of hidden neurons
- For quantum neural networks: the existence ($\mathcal{S}_0$) and breaking ($\mathcal{S}_1$) of the $\frac{\pi}{2}$-translational invariance in parameterization
- Expanding the observable in the Heinsenberg's picture

$$\mathbf{M}(\boldsymbol{\theta}) := \mathbf{U}^\dagger(\boldsymbol{\theta})\mathbf{M}\mathbf{U}(\boldsymbol{\theta}) = \sum_{\boldsymbol{\xi}\in\{0,1,2\}^p} \Phi_{\boldsymbol{\xi}}(\mathbf{M}) \prod_{l:\xi_l=1} \cos 2\theta_l \prod_{l':\xi_{l'}=2} \sin 2\theta_{l'}$$

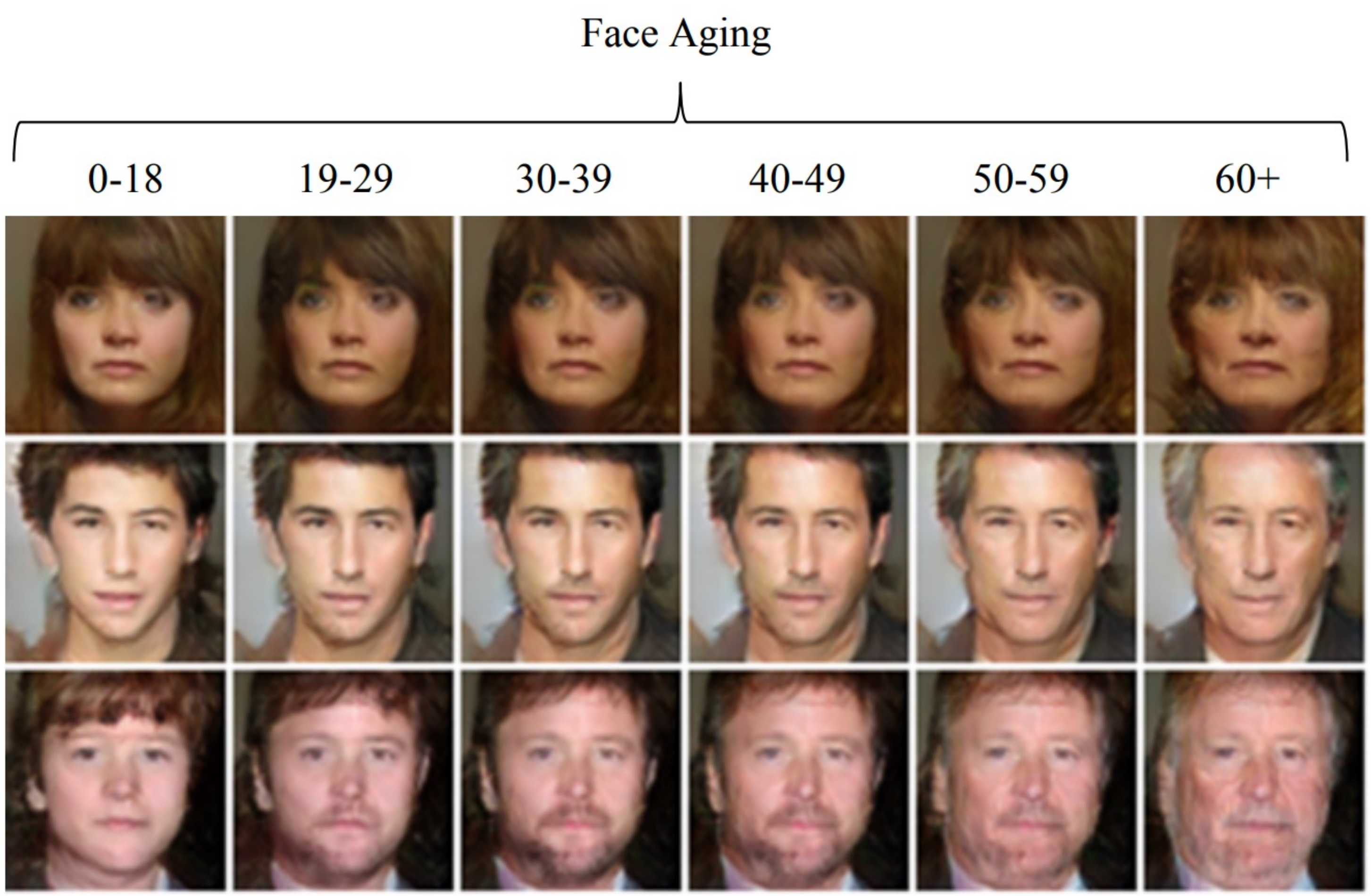
  with $\Phi_{\boldsymbol{\xi}}(\mathbf{M})$ being Hermitians, the form of which depending on the QNN design.

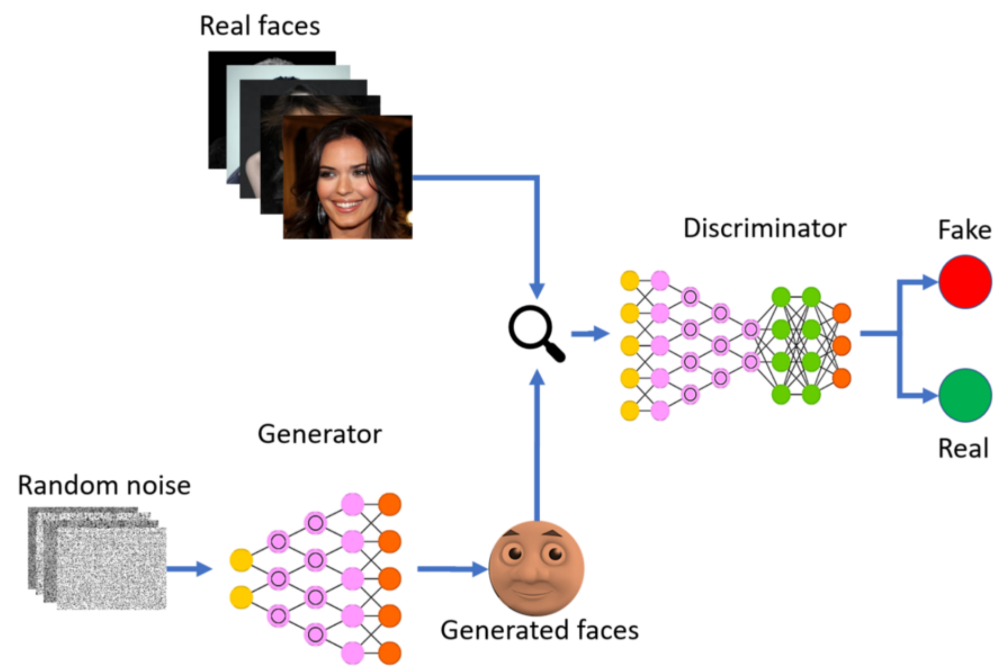
- Datasets $\mathcal{S}_0$ and $\mathcal{S}_1$ can be constructed by solving a linear system given that $\{\Phi_{\boldsymbol{\xi}}(\mathbf{M})\}_{\boldsymbol{\xi}\in\{0,1,2\}^p,\boldsymbol{\xi}\neq\mathbf{0}}$ forms a linearly independent set (*L.D. condition*)

L.D. condition proven to hold for almost all under-parameterized QNNs.

Generative Models

Powerful tool from machine learning: generative adversarial networks (GANs)



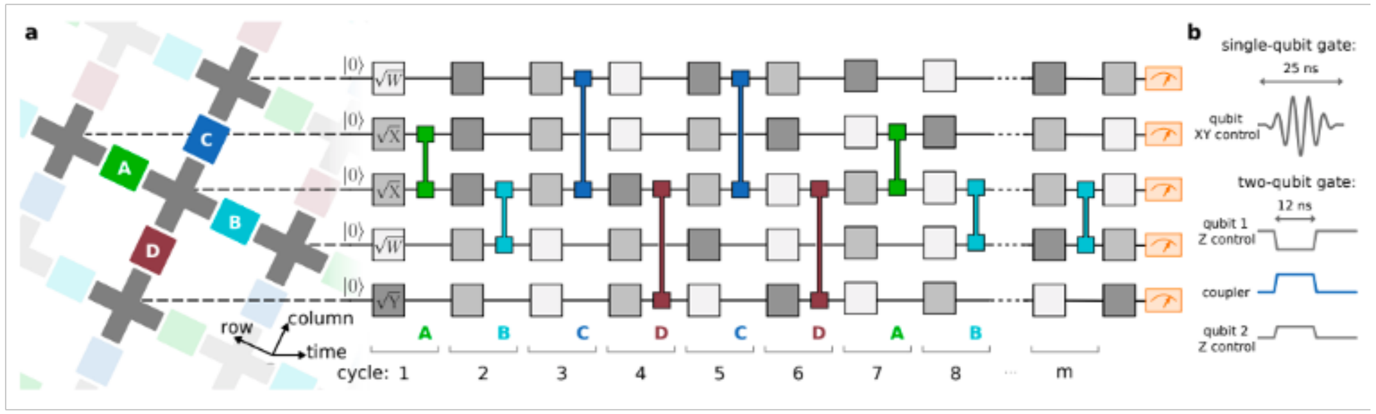


Or popularly known in deep-fake examples.

Quantum Generative Adversarial Networks (GANs)

classical distributions

quantum circuits are good at sampling!

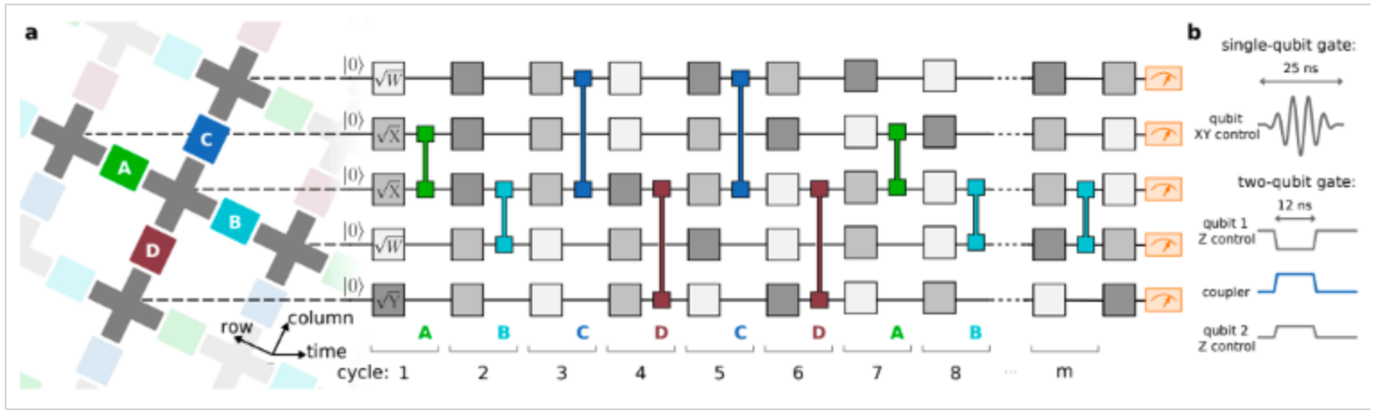


most quantum supremacy proposals (Google's random circuits, Boson sampling, etc) are sampling tasks

Quantum Generative Adversarial Networks (GANs)

classical distributions

quantum circuits are good at sampling!



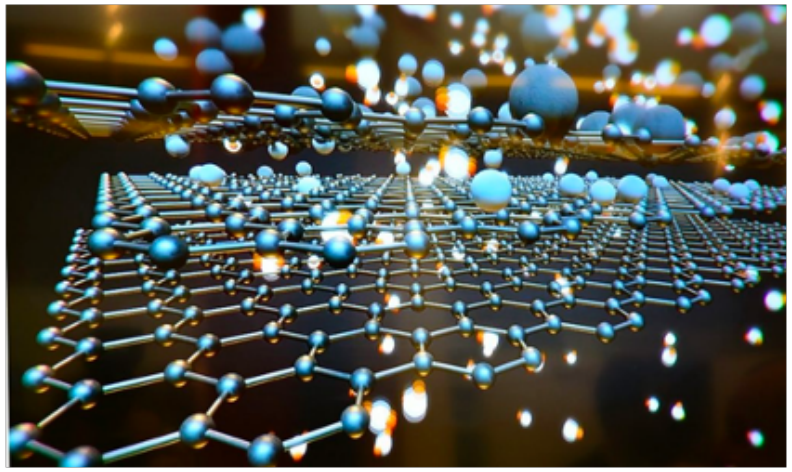
most quantum supremacy proposals (Google's random circuits, Boson sampling, etc) are sampling tasks

quantum data

only quantum circuits can generate q. data!

probing unknown quantum materials w/ GANs!

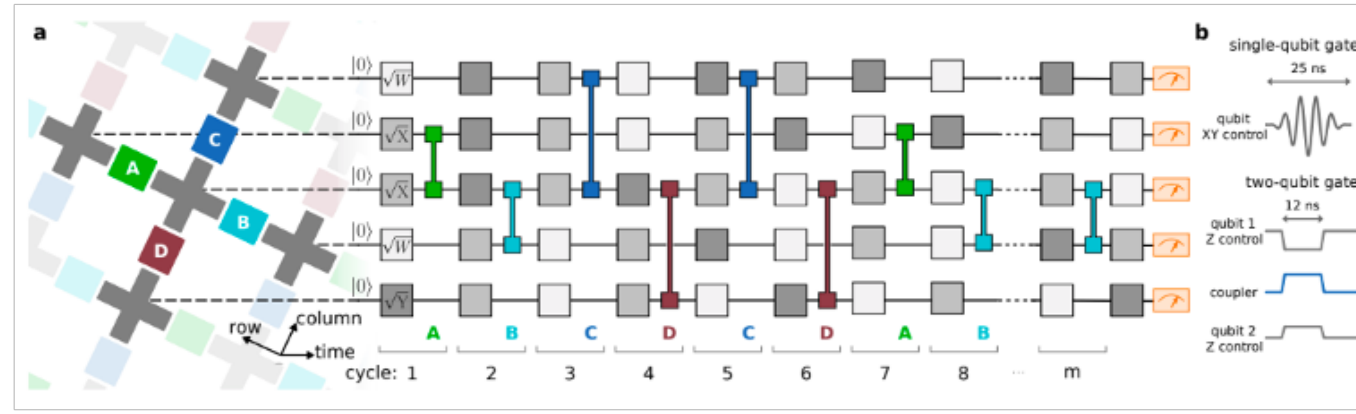
surprising quantum applications !



Quantum Generative Adversarial Networks (GANs)

classical distributions

quantum circuits are good at sampling!



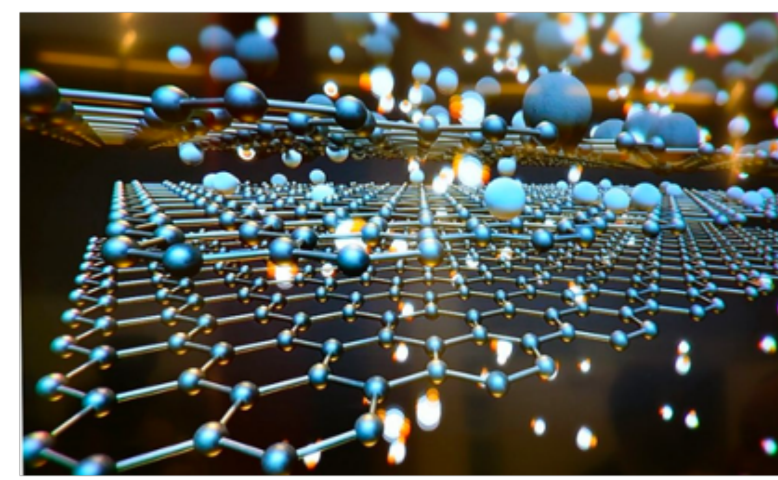
most quantum supremacy proposals (Google's random circuits, Boson sampling, etc) are sampling tasks

quantum data

only quantum circuits can generate q. data!

probing unknown quantum materials w/ GANs!

surprising quantum applications !



- Implementation: simple prototypes of quantum GANs are likely implementable on near-term noisy-intermediate-size-quantum (NISQ) machines.

Robust Training of Quantum Generative Models

Training of classical GANs is delicate and unstable!

due to the property of the loss function

Training quantum data could be even worse!

existing quantum GANs scale up poorly (limited #qbits, #para,
very slow convergence) in [BGWS19, DK18, Hu et al. 19]

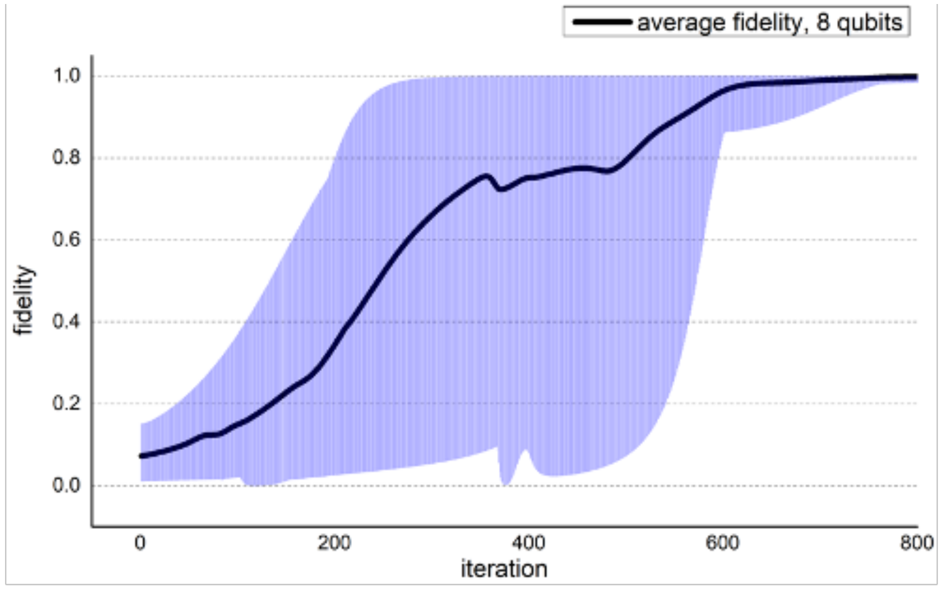
Robust Training of Quantum Generative Models

Training of classical GANs is delicate and unstable!

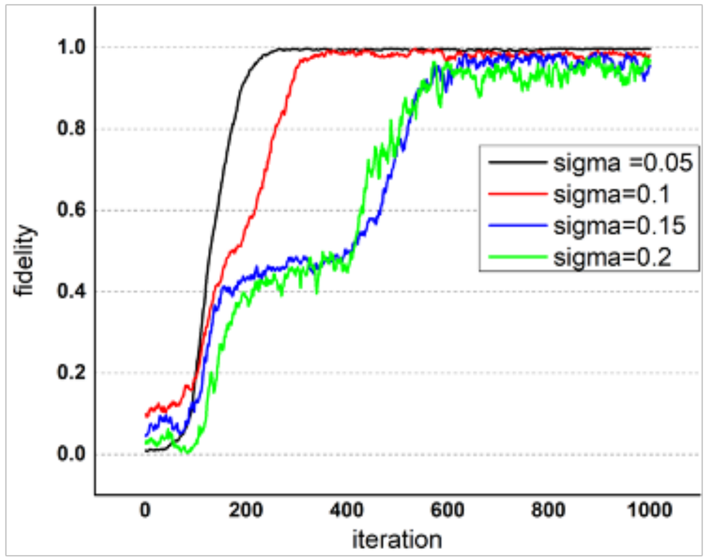
due to the property of the loss function

Training quantum data could be even worse!

existing quantum GANs scale up poorly (limited #qbits, #para, very slow convergence) in [BGWS19, DK18, Hu et al. 19]



8 qubits
200 para



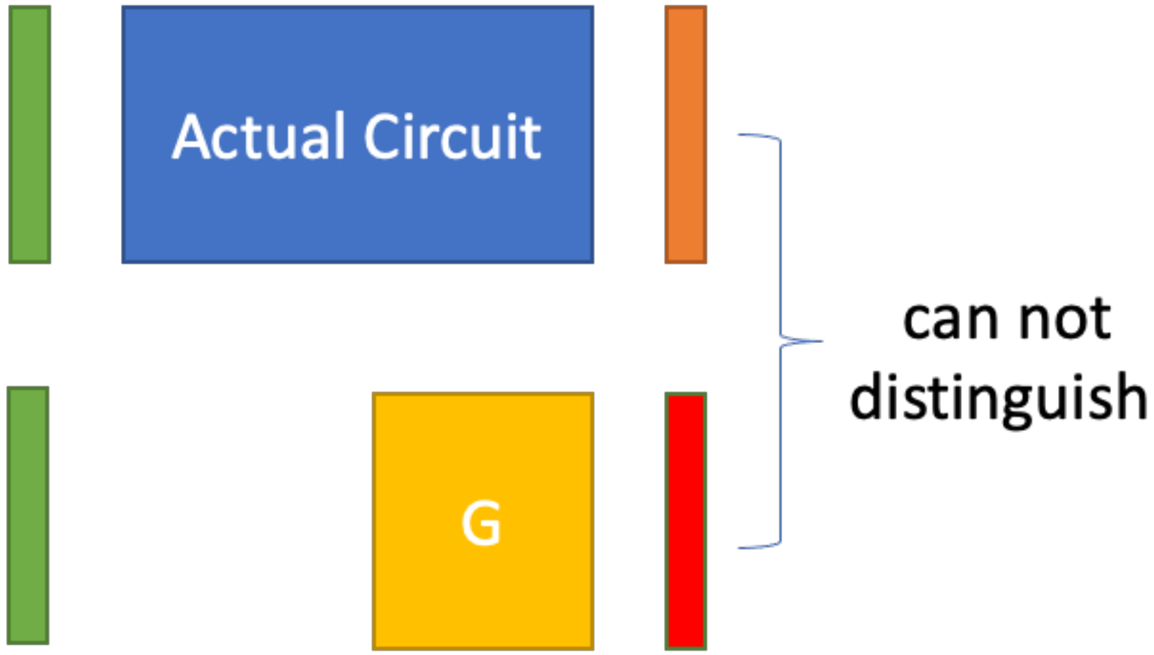
noisy
4 qubits

Contribution: **[CHLFW19, NeurIPS 2019]**

(1) more **robust** and **scalable** training even with **noisy** qubits

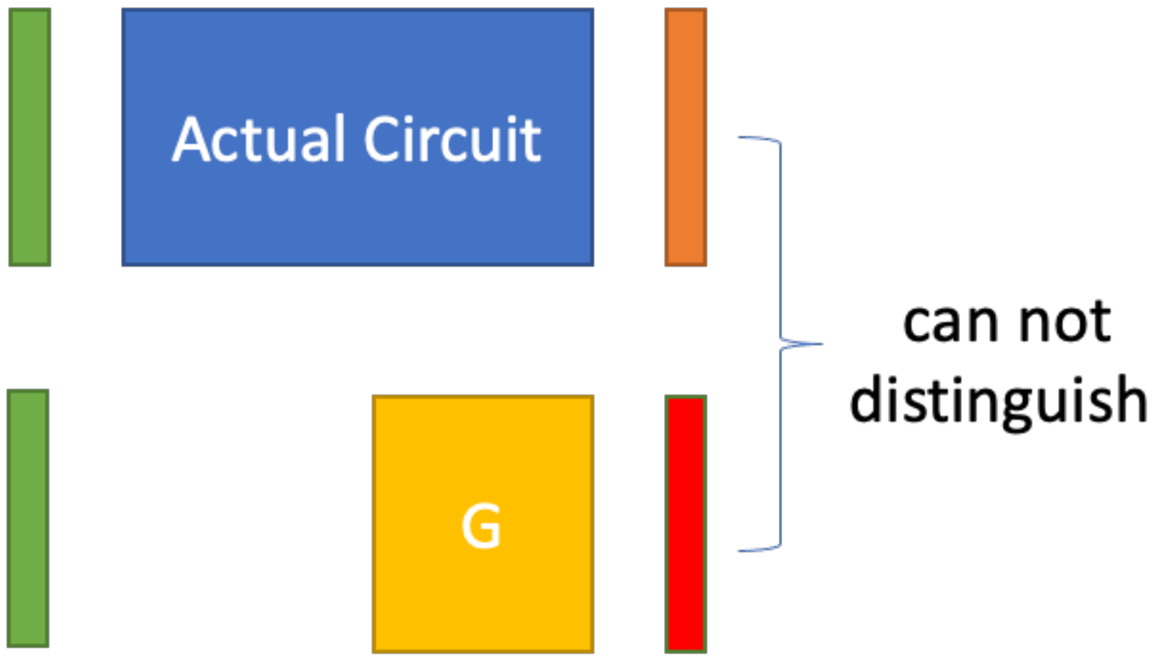
(2) a 52-gate circuit approximating a 10k-gate circuit (product-formula)

Compressing Quantum Circuits

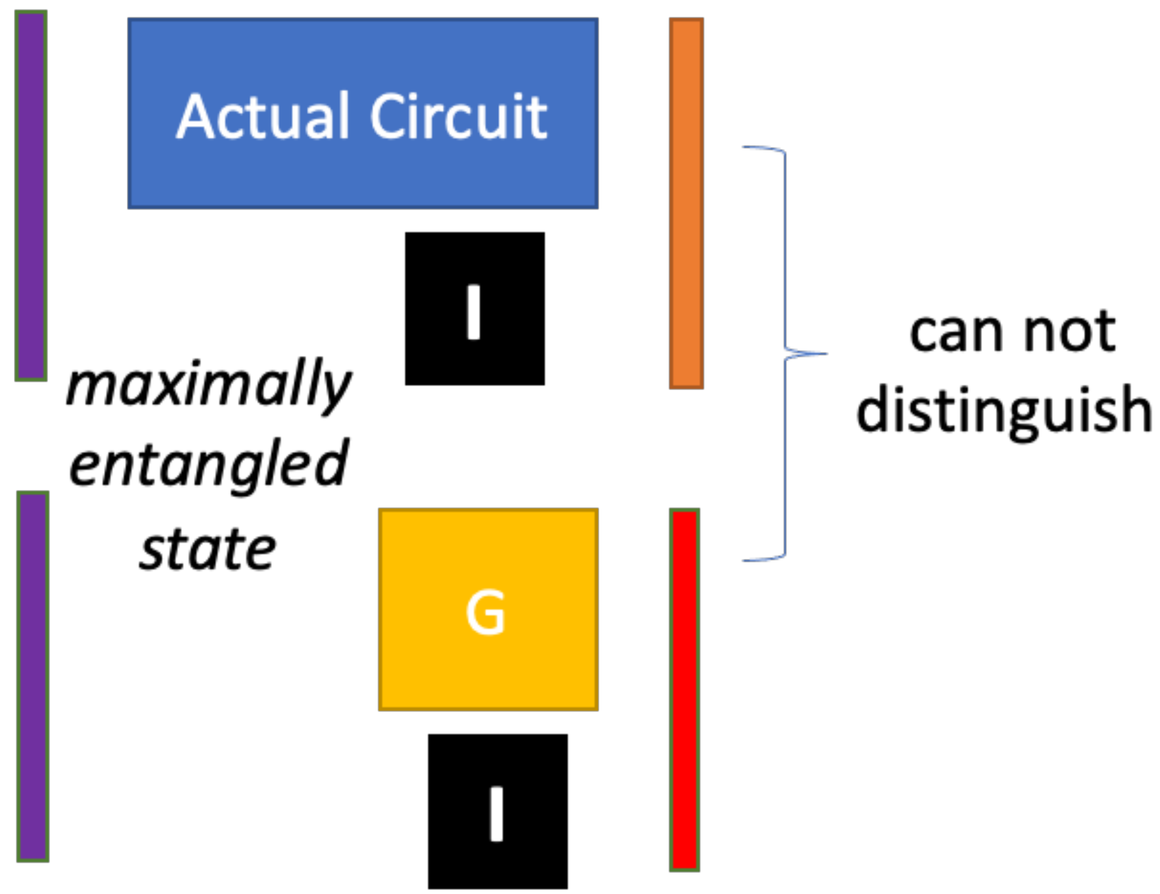


approximate the output for one input

Compressing Quantum Circuits

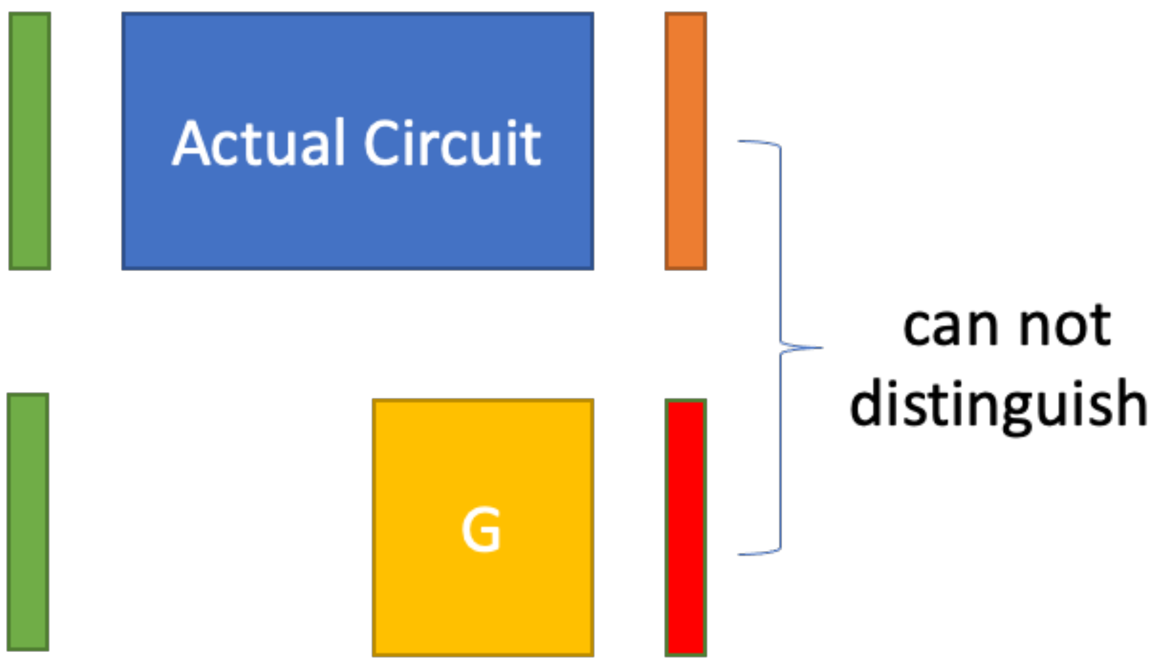


approximate the output for one input

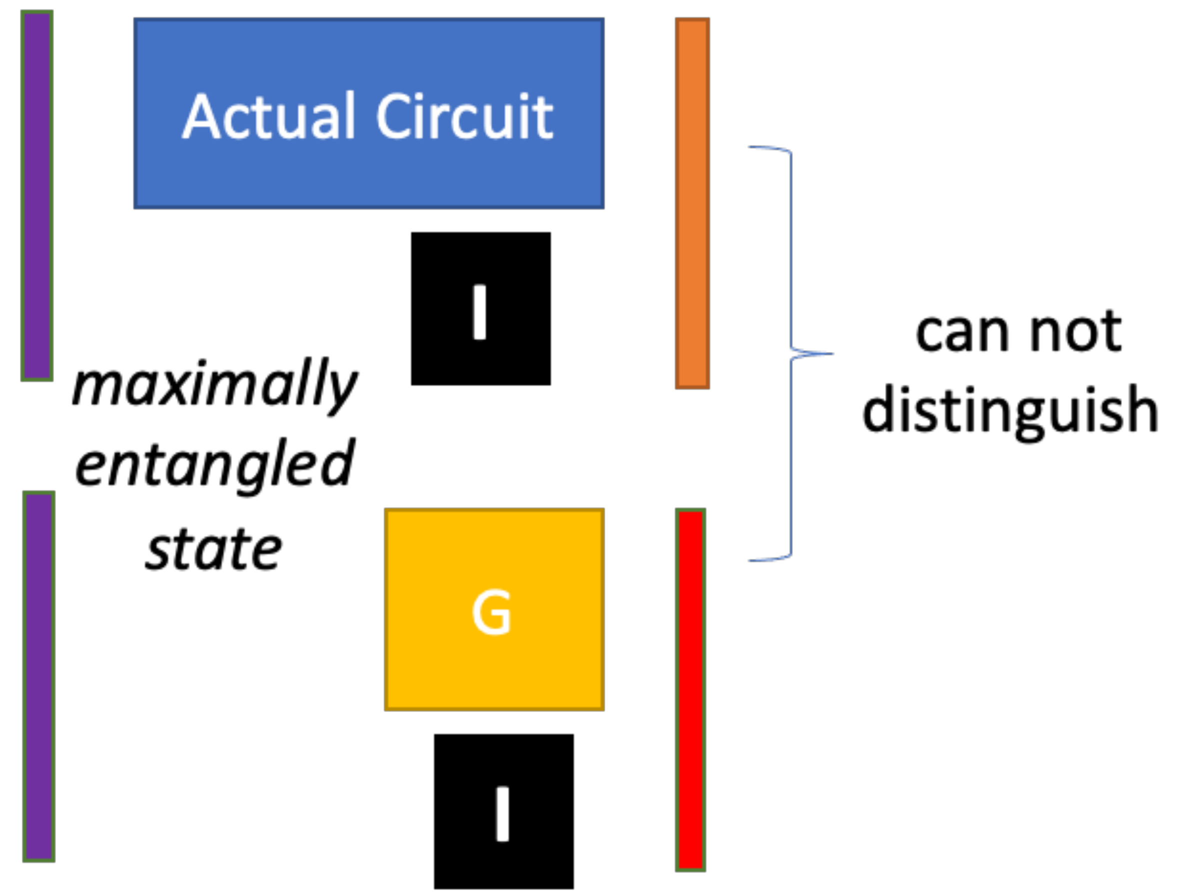


approximate the whole circuit using ***Choi-Jamiołkowski*** *isomorphism*

Compressing Quantum Circuits



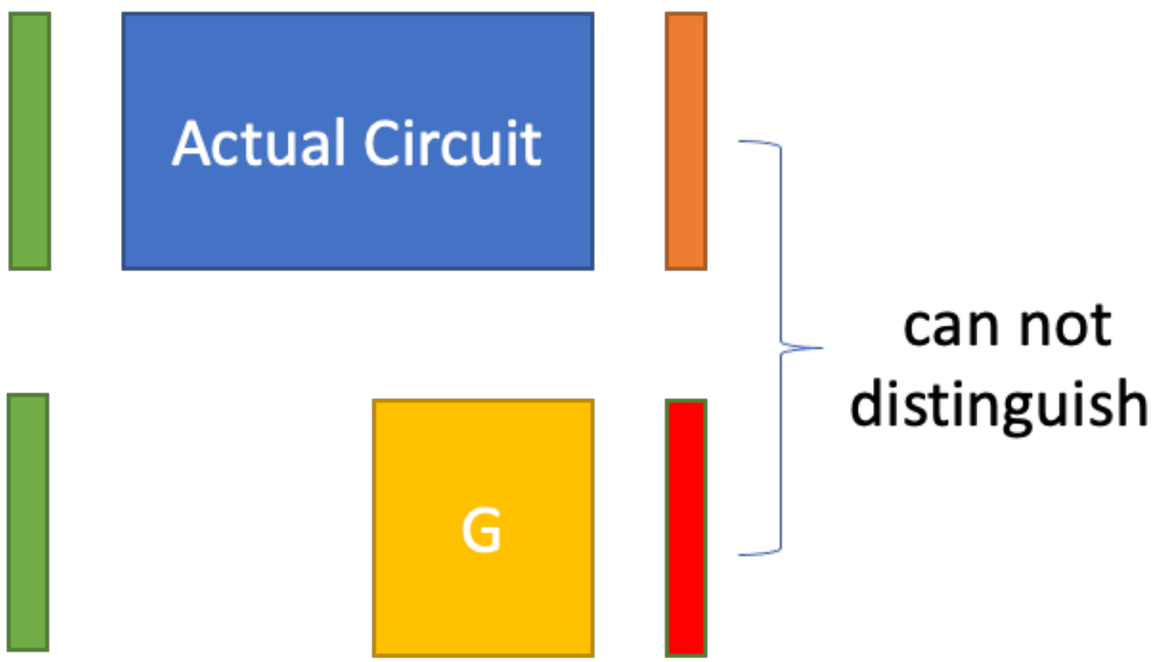
approximate the output for one input



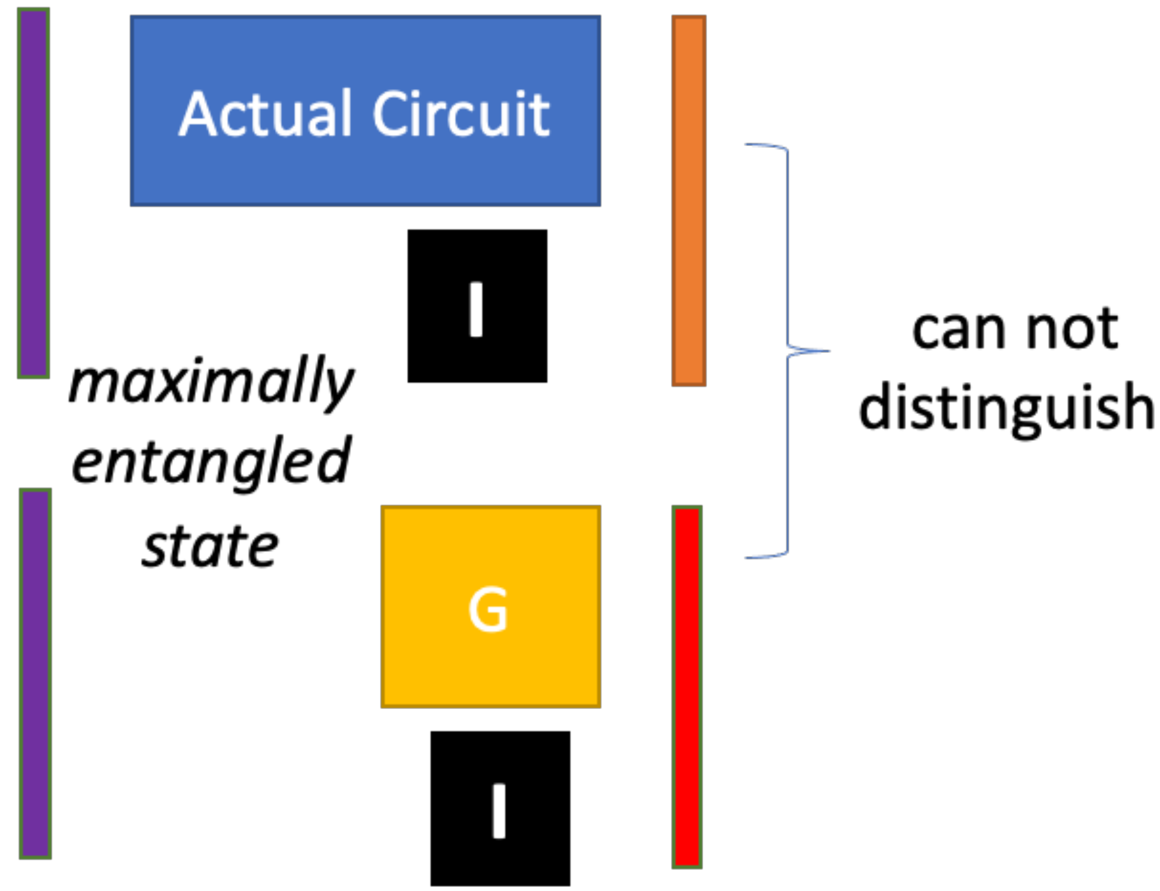
worst-case -> average-case guarantee

approximate the whole circuit using ***Choi-Jamiołkowski*** *isomorphism*

Compressing Quantum Circuits



approximate the output for one input



worst-case -> average-case guarantee

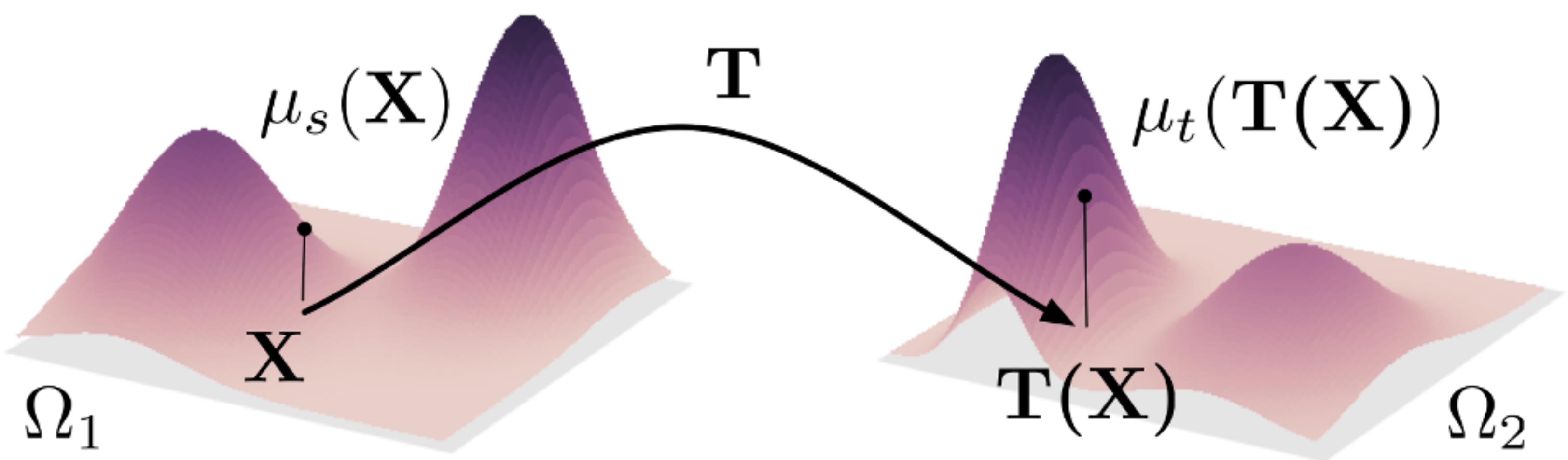
approximate the whole circuit using ***Choi-Jamiołkowski*** *isomorphism*

Examples:

(1) a **52**-gate circuit approximating a **10k**-gate circuit (product-formula) output fidelity *0.9999* over average input, worst-case error *0.15* .

(2) a scale-down experimental proposal for compressing 50 to 10 gates for a lot of physics-motivated quantum circuits.

Quantum Wasserstein Distance w/ regularization

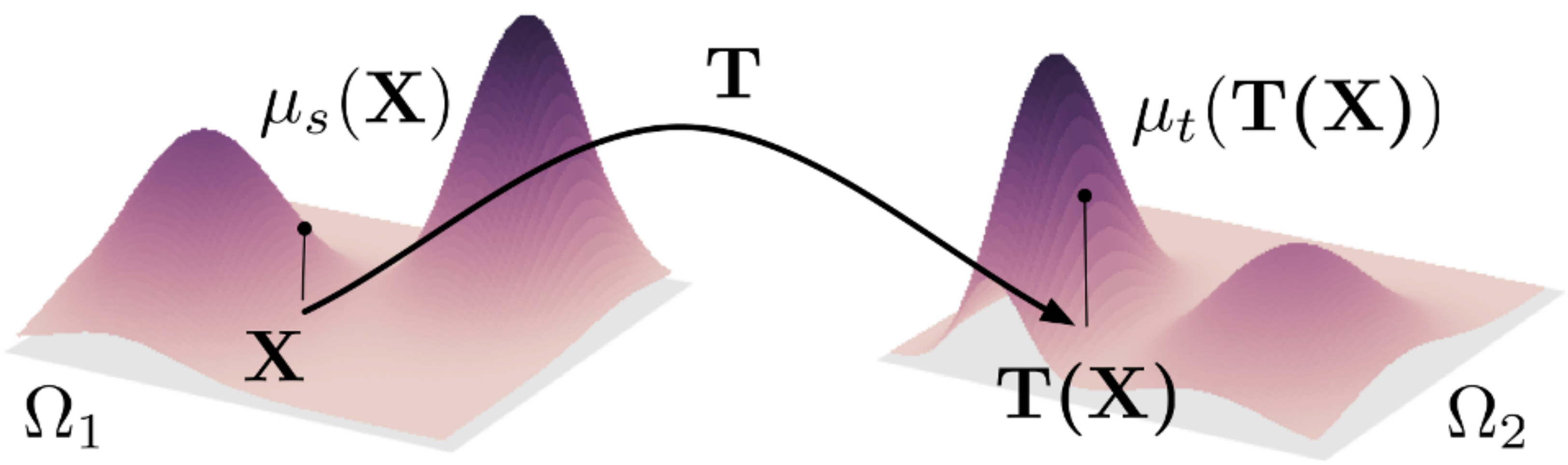


Optimal transport over probability spaces Ω_1, Ω_2

* can induce a class of measures that are **smooth** and **numerically stable**.

- lead to the design of Wasserstein GAN

Quantum Wasserstein Distance w/ regularization



Optimal transport over probability spaces Ω_1, Ω_2

* can induce a class of measures that are **smooth** and **numerically stable**.

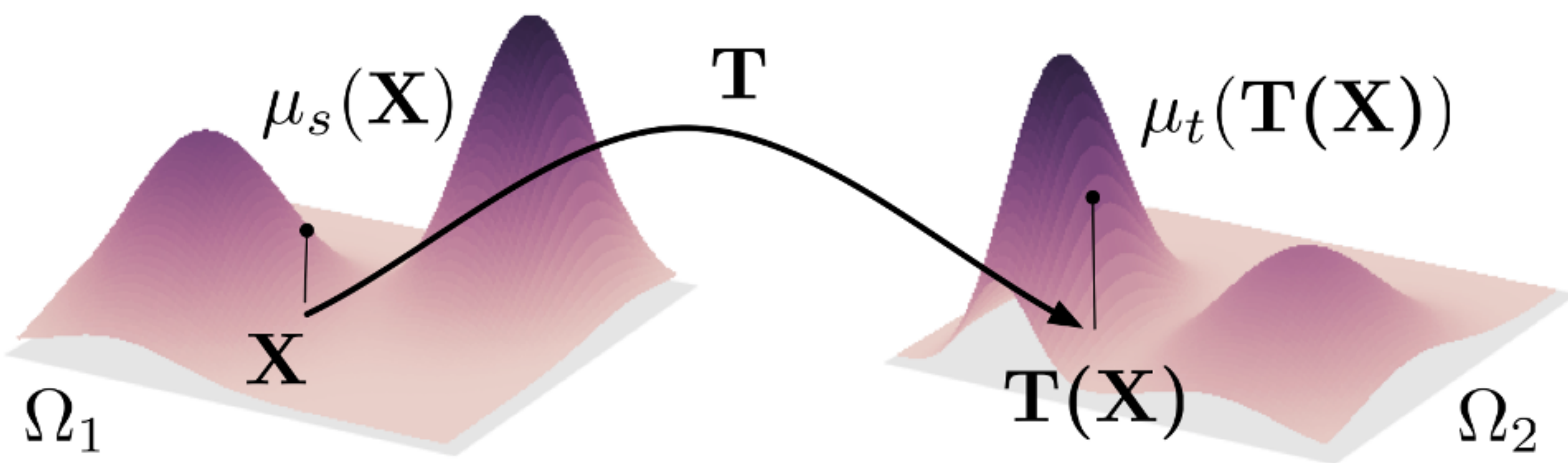
- lead to the design of Wasserstein GAN

Technical Contribution:

(1) one proposal to study *optimal transport* in *operator/non-commutative* space, including a quantum Wasserstein distance and its property.

(2) architecture of quantum WGAN for robust and scalable training, i.e., all training steps could in principle run efficiently on quantum machines.

Quantum Wasserstein Distance w/ regularization



Optimal transport over probability spaces Ω_1, Ω_2

* can induce a class of measures that are **smooth** and **numerically stable**.

- lead to the design of Wasserstein GAN

Technical Contribution:

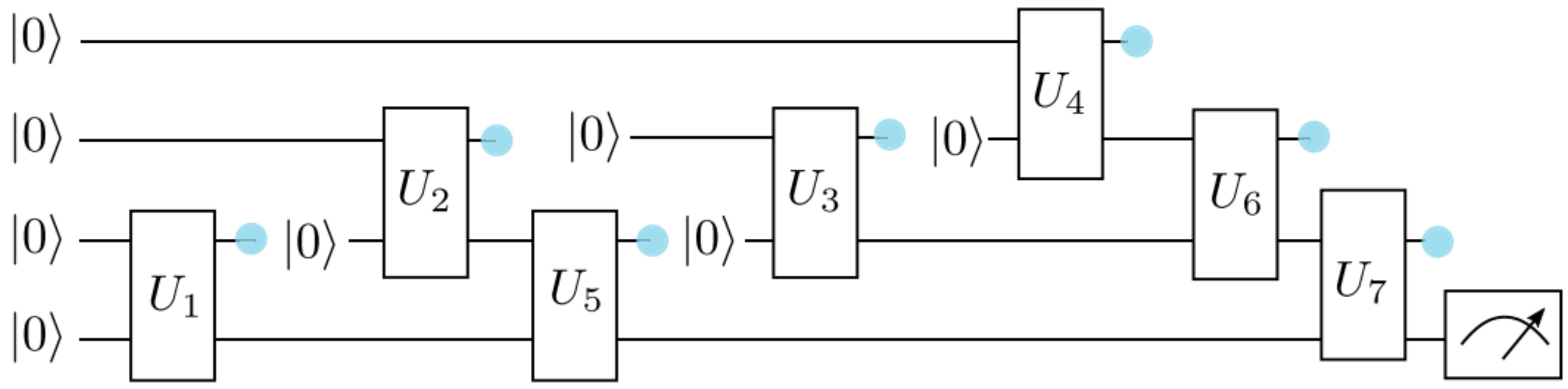
(1) one proposal to study *optimal transport* in *operator/non-commutative* space, including a quantum Wasserstein distance and its property.

(2) architecture of quantum WGAN for robust and scalable training, i.e., all training steps could in principle run efficiently on quantum machines.

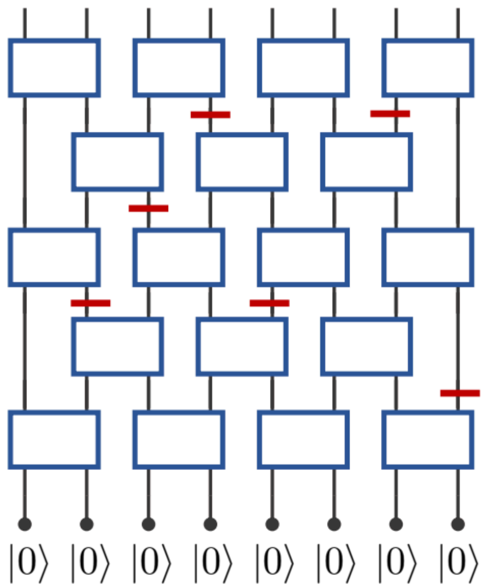
** Proposals using quantum optimal transport to study non-equilibrium physics*

Differentiable Quantum Programming

New Variational Constructs:



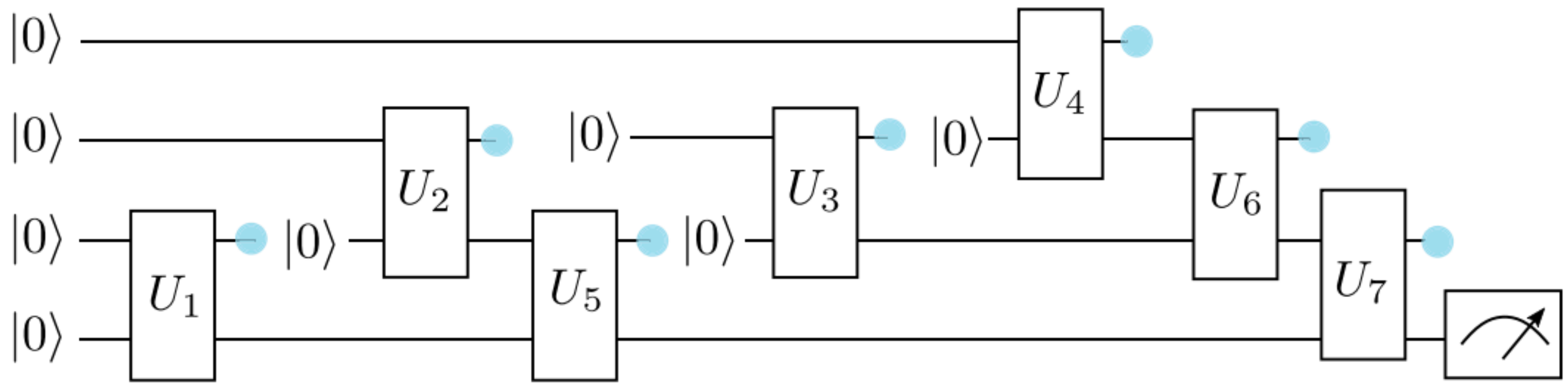
Resets + **Measurements** to Save Resources in NISQ machines



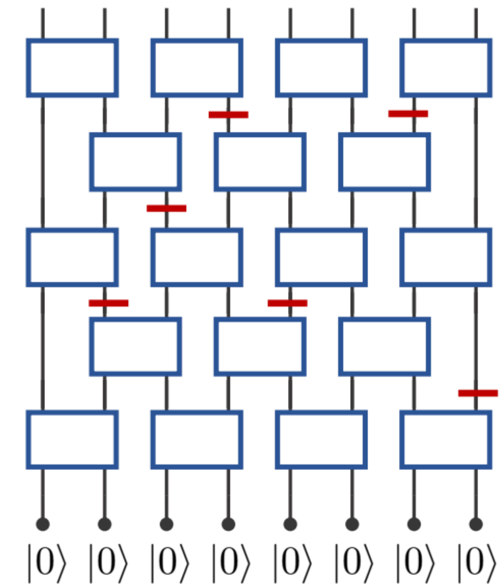
Measurement induced Phase Transition

Differentiable Quantum Programming

New Variational Constructs:



Resets + **Measurements** to Save Resources in NISQ machines



Measurement induced Phase Transition

Classically,

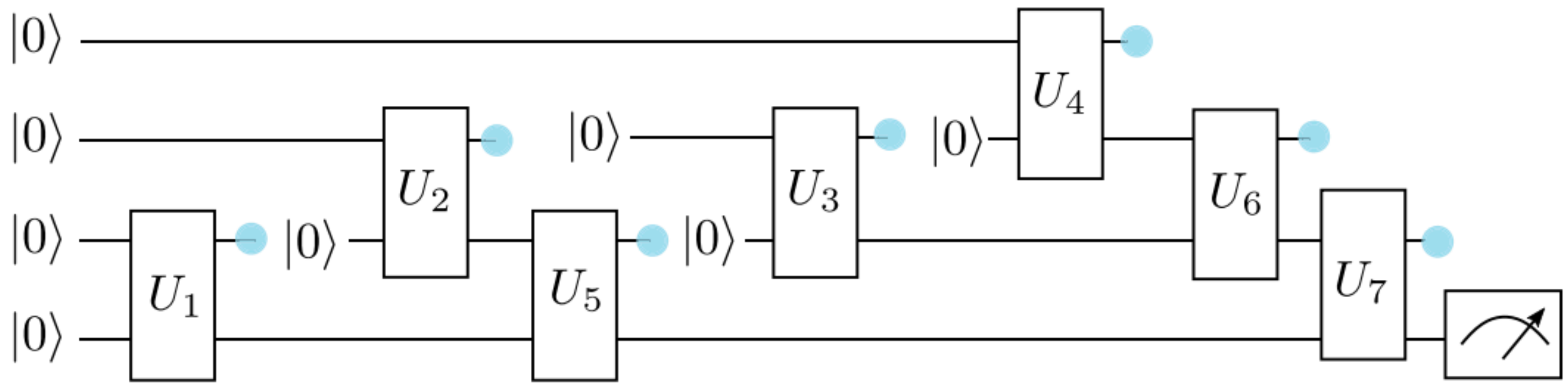
Neural Networks + Program Features (Control/Loop) -> Differentiable Programming

"Deep Learning est mort. Vive Differentiable Programming!"

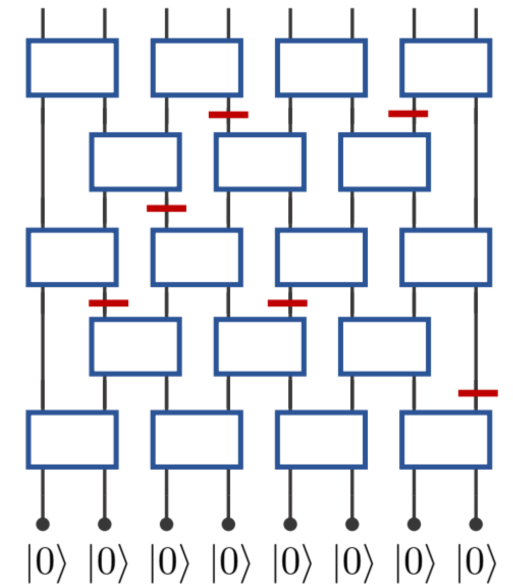
----- Yann LeCun

Differentiable Quantum Programming

New Variational Constructs:



Resets + **Measurements** to Save Resources in NISQ machines



Measurement induced Phase Transition

Classically,

Neural Networks + Program Features (Control/Loop) -> Differentiable Programming

"Deep Learning est mort. Vive Differentiable Programming!"

----- Yann LeCun

Quantumly, build the foundation of *differentiable quantum programming* **[PLDI'20]**

allow **efficient** training of q. variational models w/ program features

demonstrate the first quantum *neural-symbolic* application

Quantum Neuro-Symbolic Application

$$
\begin{aligned}
Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4]; \\
& R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4]; \\
& R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4],
\end{aligned}
$$

Quantum Neuro-Symbolic Application

$$
\begin{aligned}
Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4]; \\
& R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4]; \\
& R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4],
\end{aligned}
$$

(no control) $P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi).$

Quantum Neuro-Symbolic Application

$$
\begin{aligned}
Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4]; \\
& R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4]; \\
& R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4],
\end{aligned}
$$

(no control) $P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi).$

(w/ control) $P_2(\Theta, \Phi, \Psi) \equiv Q(\Theta); \textbf{case } M[q_1] = 0 \rightarrow \quad Q(\Phi)$

(measurements in the middle) $\qquad 1 \rightarrow \quad Q(\Psi).$

Quantum Neuro-Symbolic Application

$$
\begin{aligned}
Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4]; \\
& R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4]; \\
& R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4],
\end{aligned}
$$

Note that $P_1(\Theta, \Phi)$ and $P_2(\Theta, \Phi, \Psi)$ run the same # of gates.

(no control) $P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi).$

(w/ control) $P_2(\Theta, \Phi, \Psi) \equiv Q(\Theta); \mathbf{case}\ M[q_1] = 0 \rightarrow \quad Q(\Phi)$

(measurements in the middle) $\quad 1 \rightarrow \quad Q(\Psi).$

Quantum Neuro-Symbolic Application

$$\begin{aligned} Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4]; \\ & R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4]; \\ & R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4], \end{aligned}$$

Note that $P_1(\Theta, \Phi)$ and $P_2(\Theta, \Phi, \Psi)$ run the same # of gates.

(no control) $P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi).$

(w/ control) $P_2(\Theta, \Phi, \Psi) \equiv Q(\Theta); \mathbf{case}\ M[q_1] = 0 \rightarrow \quad Q(\Phi)$

(measurements in the middle) $\qquad 1 \rightarrow \quad Q(\Psi).$

Simple classification task w/ ground

$$f(z) = \neg(z_1 \oplus z_4), z = z_1 z_2 z_3 z_4 \in \{0,1\}^4$$

via the following **square loss**

$$\text{loss} = \sum_{z \in \{0,1\}^4} 0.5 * (l_{\boldsymbol{\theta}}(z) - f(z))^2$$

Quantum Neuro-Symbolic Application

$$
\begin{aligned}
Q(\Gamma) \equiv \quad & R_X(\gamma_1)[q_1]; R_X(\gamma_2)[q_2]; R_X(\gamma_3)[q_3]; R_X(\gamma_4)[q_4];\\
& R_Y(\gamma_5)[q_1]; R_Y(\gamma_6)[q_2]; R_Y(\gamma_7)[q_3]; R_Y(\gamma_8)[q_4];\\
& R_Z(\gamma_9)[q_1]; R_Z(\gamma_{10})[q_2]; R_Z(\gamma_{11})[q_3]; R_Z(\gamma_{12})[q_4],
\end{aligned}
$$

Note that $P_1(\Theta, \Phi)$ and $P_2(\Theta, \Phi, \Psi)$ run the same # of gates.

(no control) $P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi).$

(w/ control) $P_2(\Theta, \Phi, \Psi) \equiv Q(\Theta); \textbf{case } M[q_1] = 0 \rightarrow Q(\Phi)$

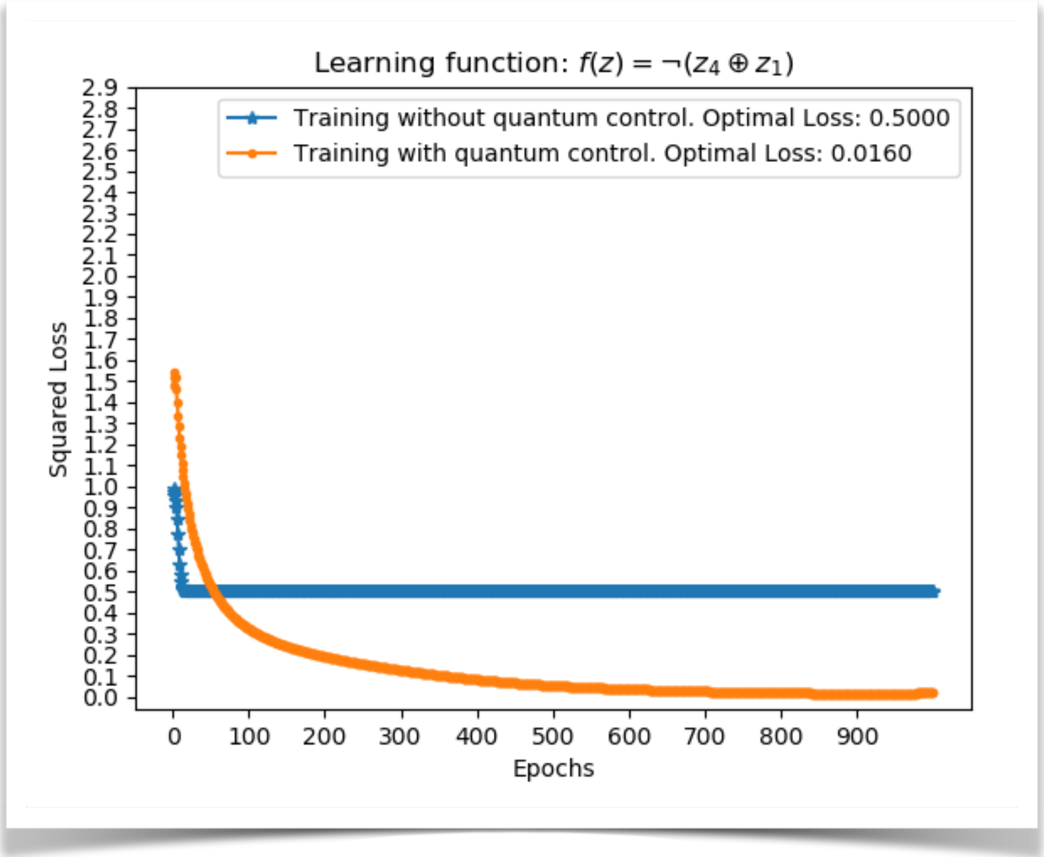
(measurements in the middle) $\quad 1 \rightarrow Q(\Psi).$

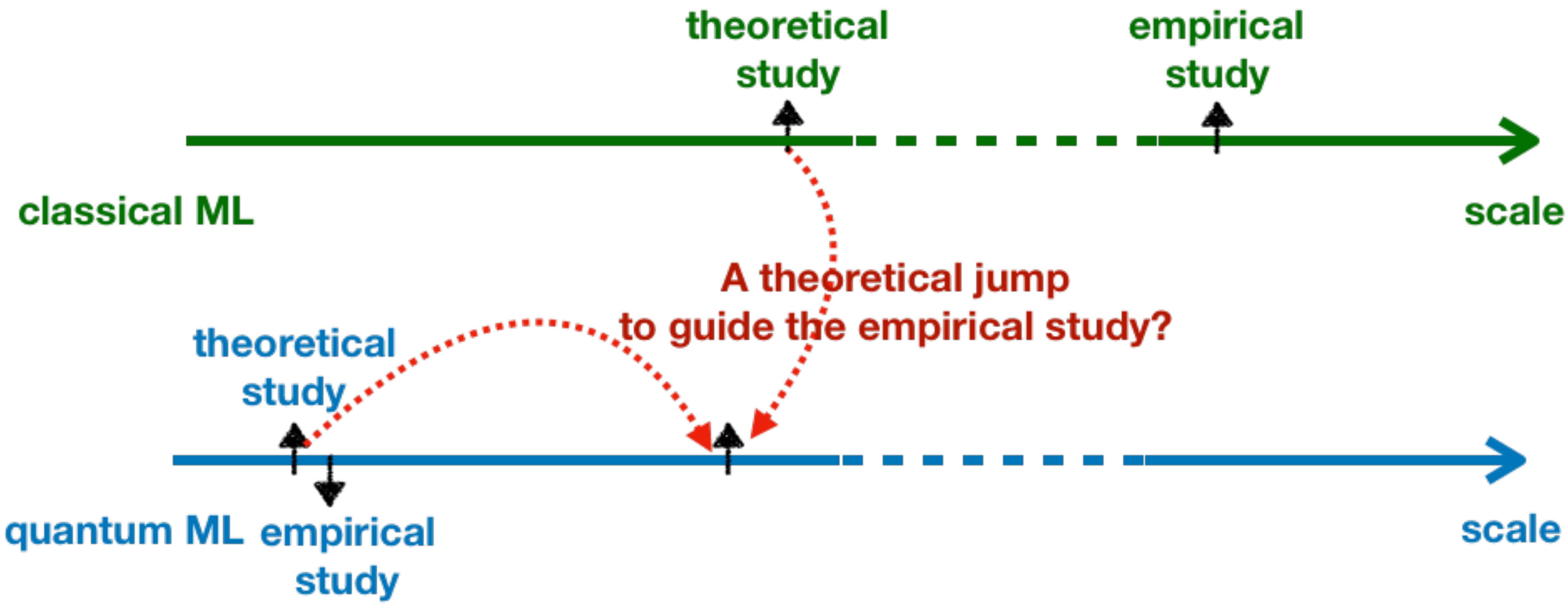
Simple classification task w/ ground

$$f(z) = \neg(z_1 \oplus z_4), z = z_1 z_2 z_3 z_4 \in \{0,1\}^4$$

via the following **square loss**

$$\text{loss} = \sum_{z \in \{0,1\}^4} 0.5 * (l_{\boldsymbol{\theta}}(z) - f(z))^2$$





Quantum Wasserstein GAN:
(NeurIPS 2019) github:/yiminghwang/qWGAN

Differentiable Quantum Prog-Lang:
(PLDI 2020) github:/LibertasSpZ/adcompile