Absence of Barren Plateau in Quantum Convolutional Neural Networks

arXiv:2011.02966

Arthur Pesah
University College London (UCL)
Los Alamos National Laboratory (LANL)

Marco Cerezo
LANL

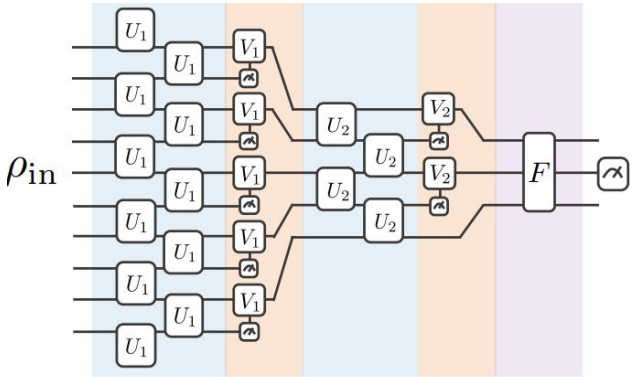
Samson Wang
Imperial/LANL

Tyler Volkoff
LANL

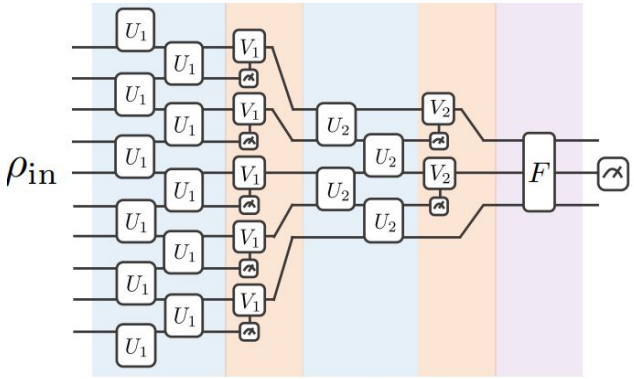
Andrew Sornborger
LANL

Patrick Coles
LANL

Is there a barren plateau problem in the quantum CNN?

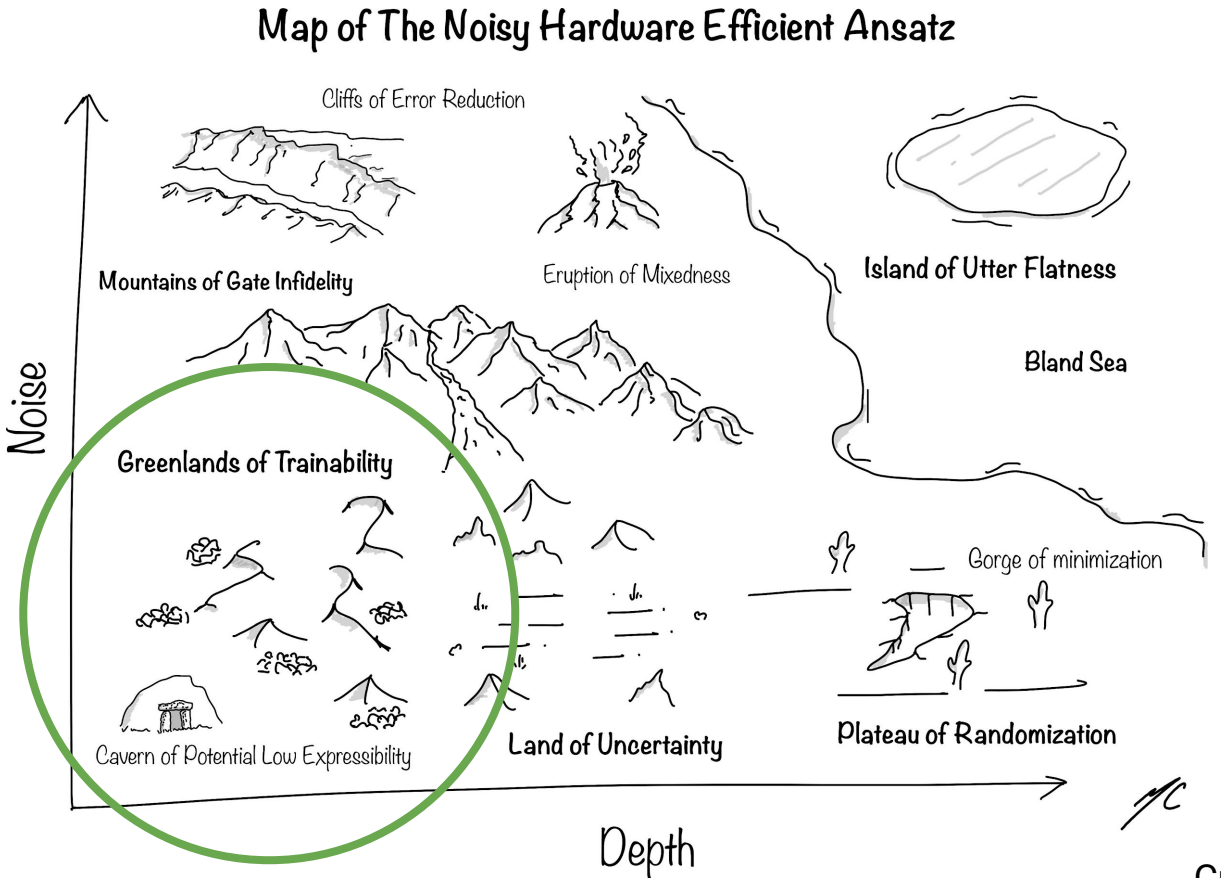


Is there a barren plateau problem in the quantum CNN?



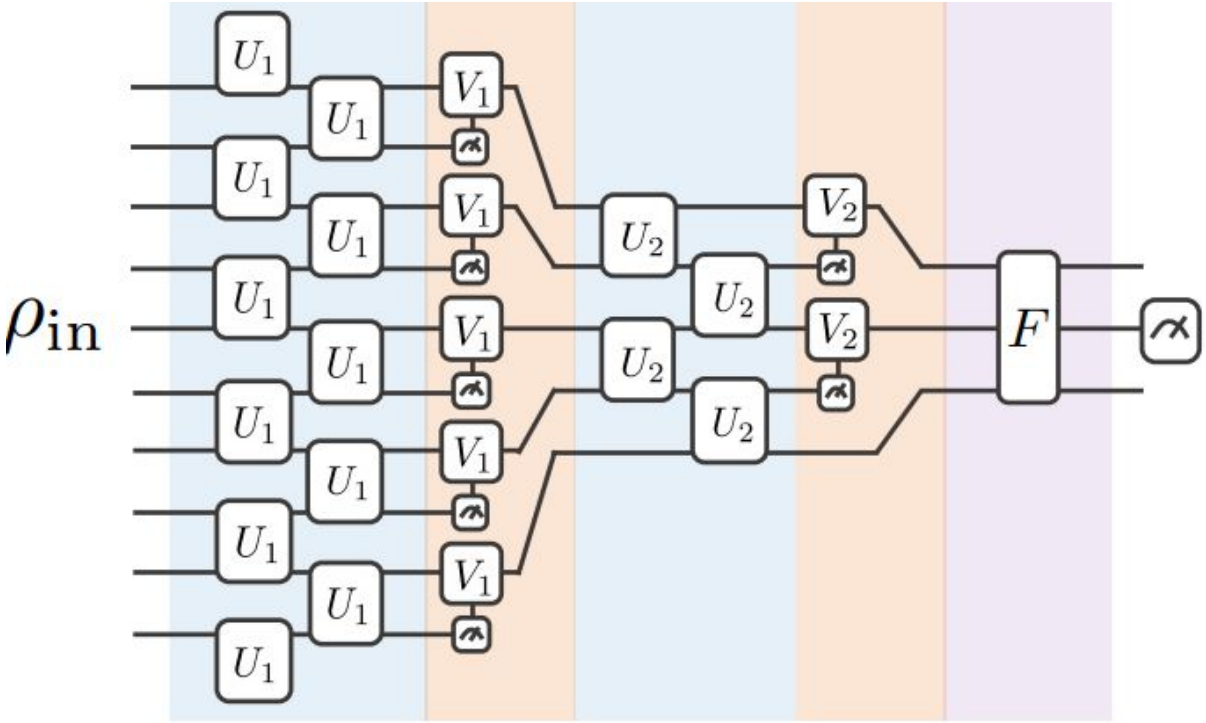
Our work

Is there a barren plateau problem in the quantum CNN?



Credit: Marco Cerezo

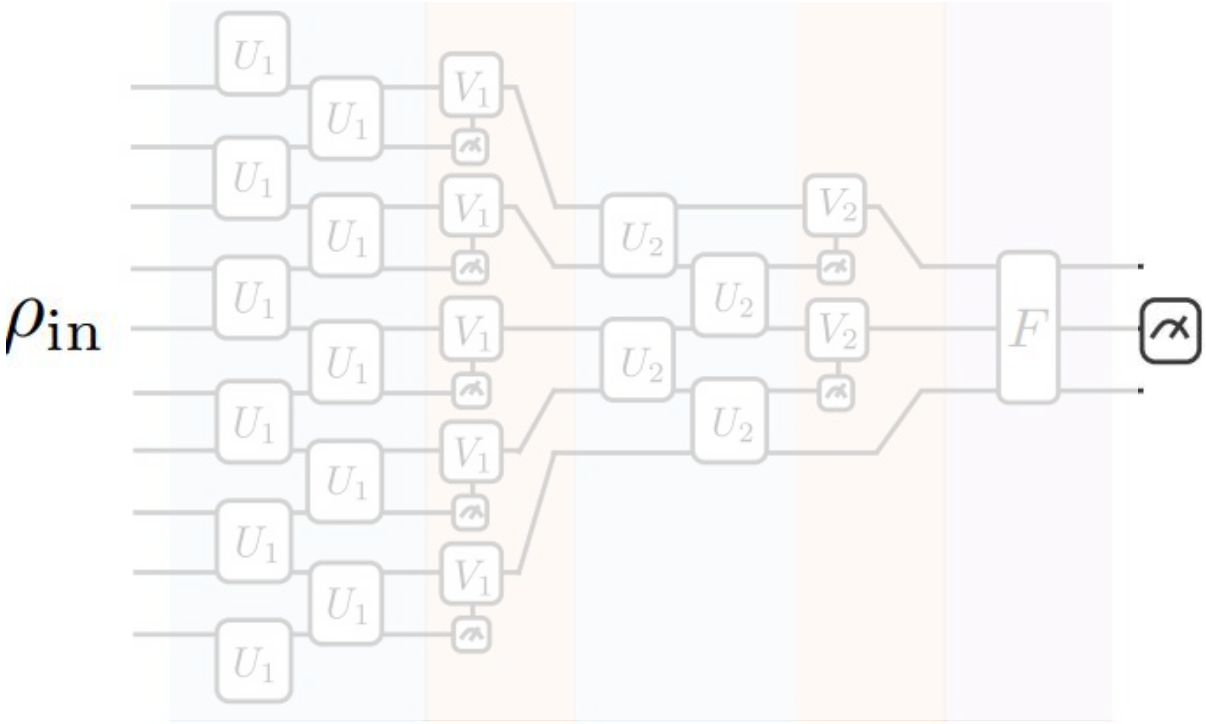
Quantum Convolutional Neural Network (QCNN)



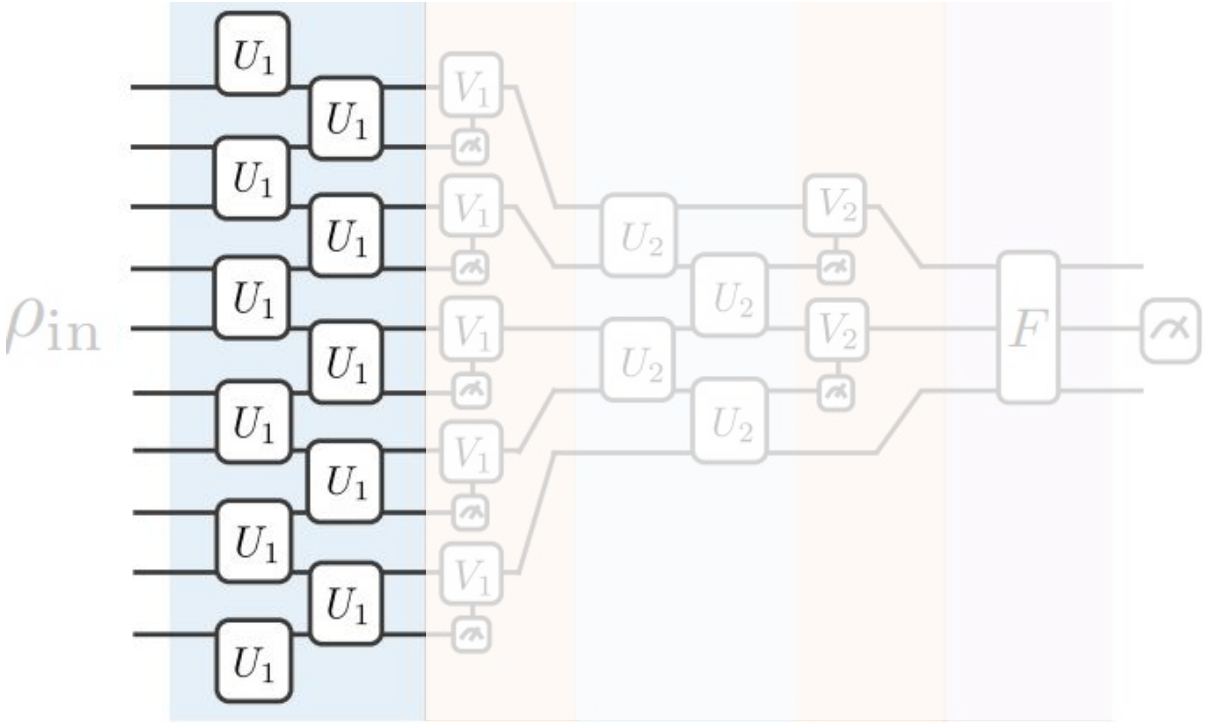
Cong et al, *Quantum Convolutional Neural Network*, 2018, arXiv:1810.03787

Grant et al., *Hierarchical quantum classifiers*, 2018, arXiv:1804.03680

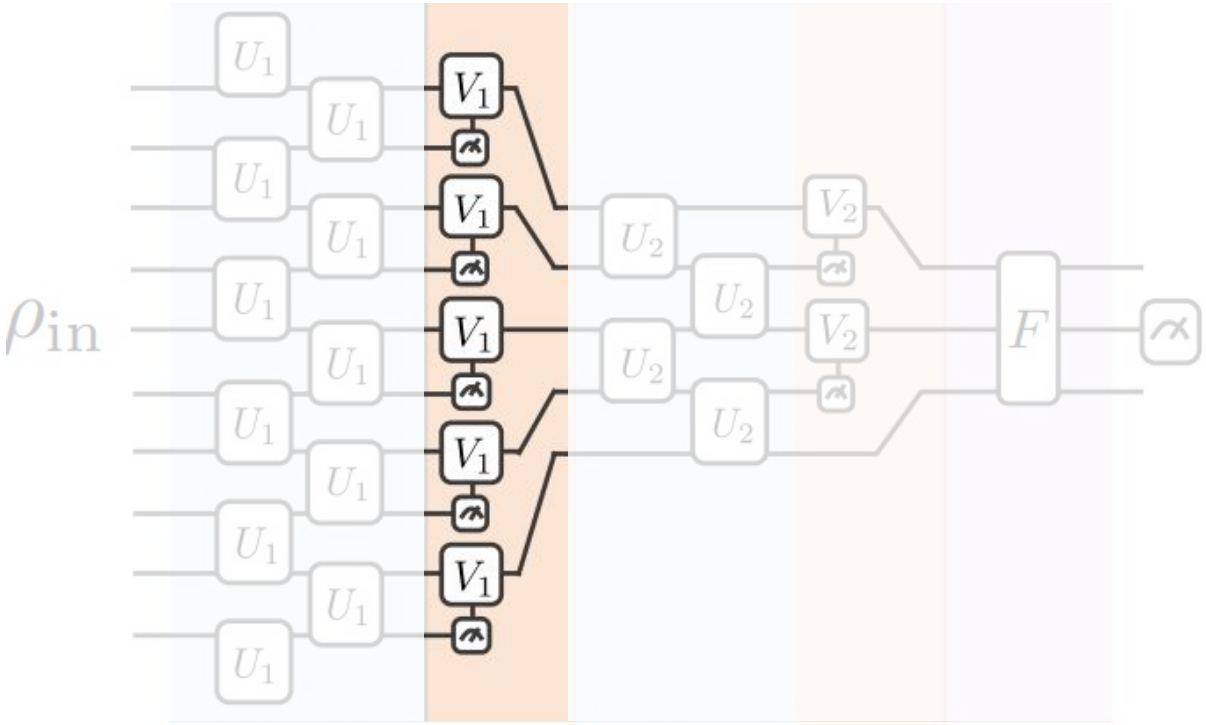
Quantum Convolutional Neural Network (QCNN)



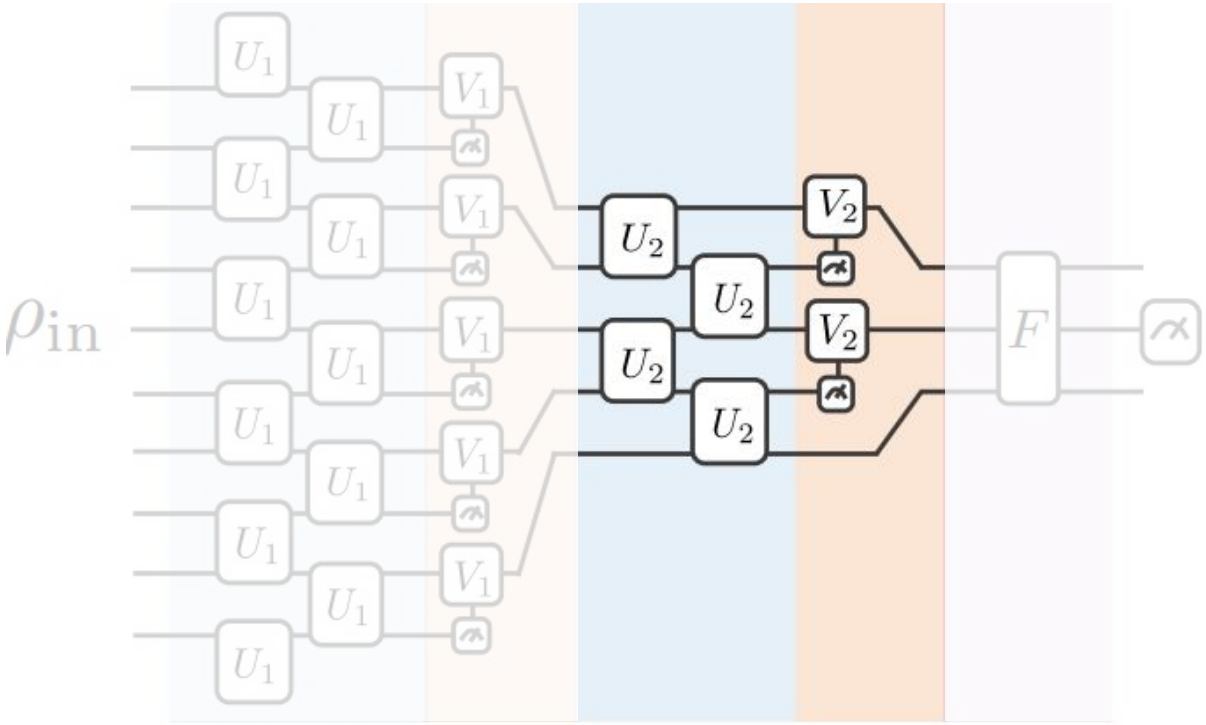
Quantum Convolutional Neural Network (QCNN)



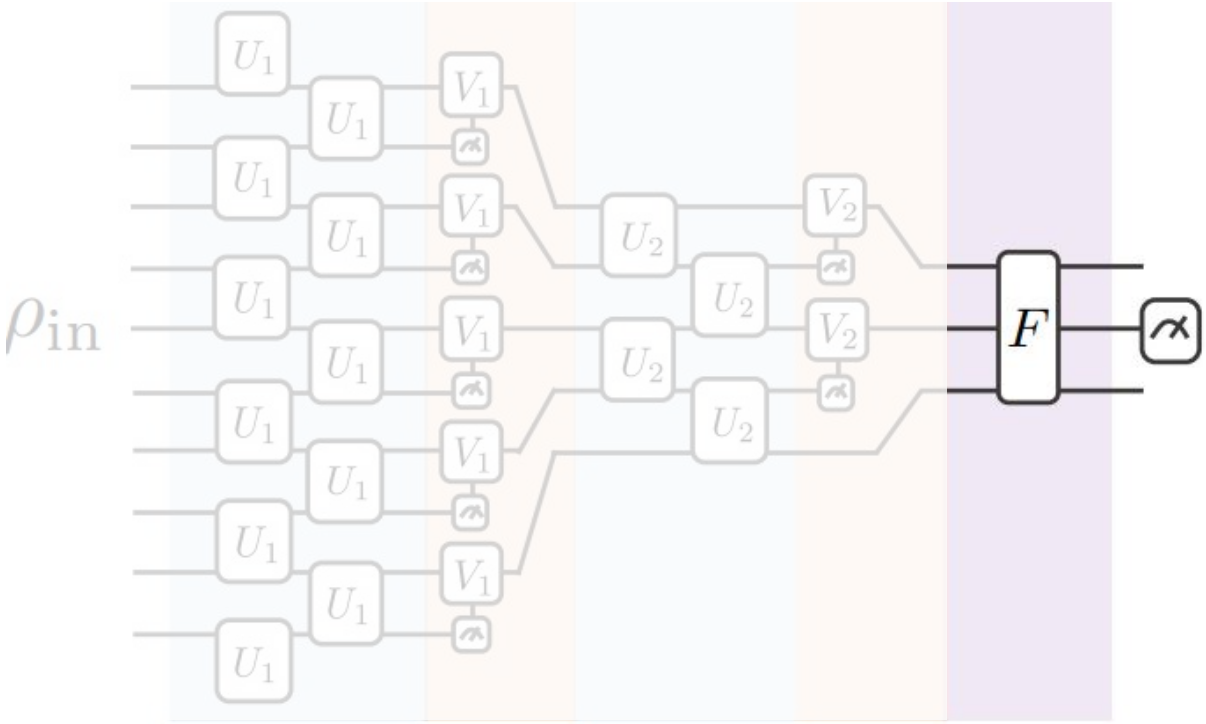
Quantum Convolutional Neural Network (QCNN)



Quantum Convolutional Neural Network (QCNN)



Quantum Convolutional Neural Network (QCNN)

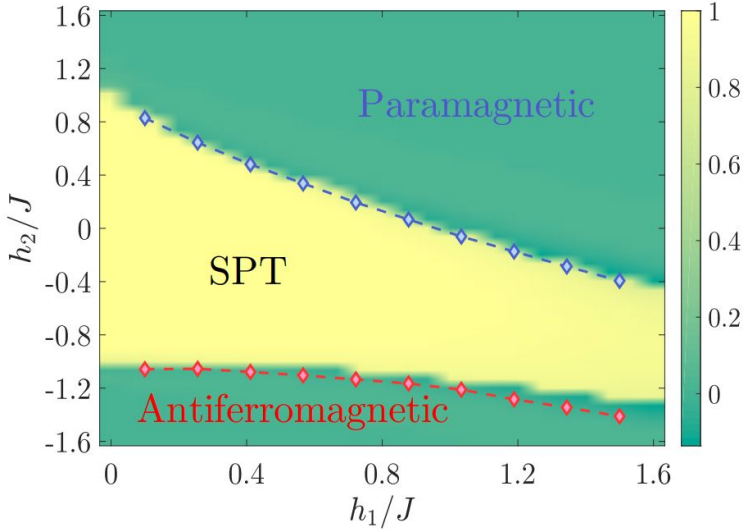


Applications

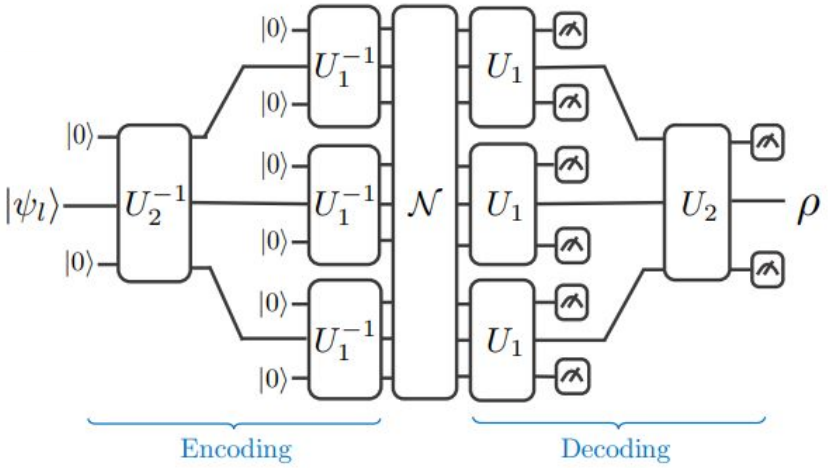
Phase detection

1D Haldane Chain

$$H = -J \sum_{i=1}^{N-2} Z_i X_{i+1} Z_{i+2} - h_1 \sum_{i=1}^{N} X_i - h_2 \sum_{i=1}^{N-1} X_i X_{i+1}$$



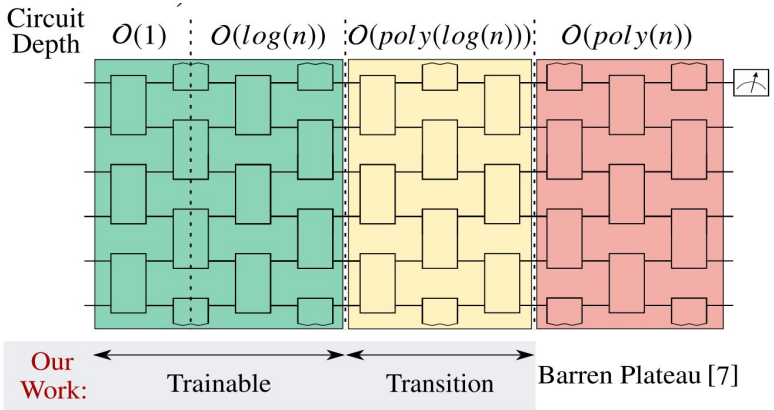
Quantum error correction



Barren plateau for the QCNN: first intuition

Previous work: the hardware-efficient ansatz does **not** have barren plateau if:

1. We have a local-cost function
2. We have a logarithmic number of layers



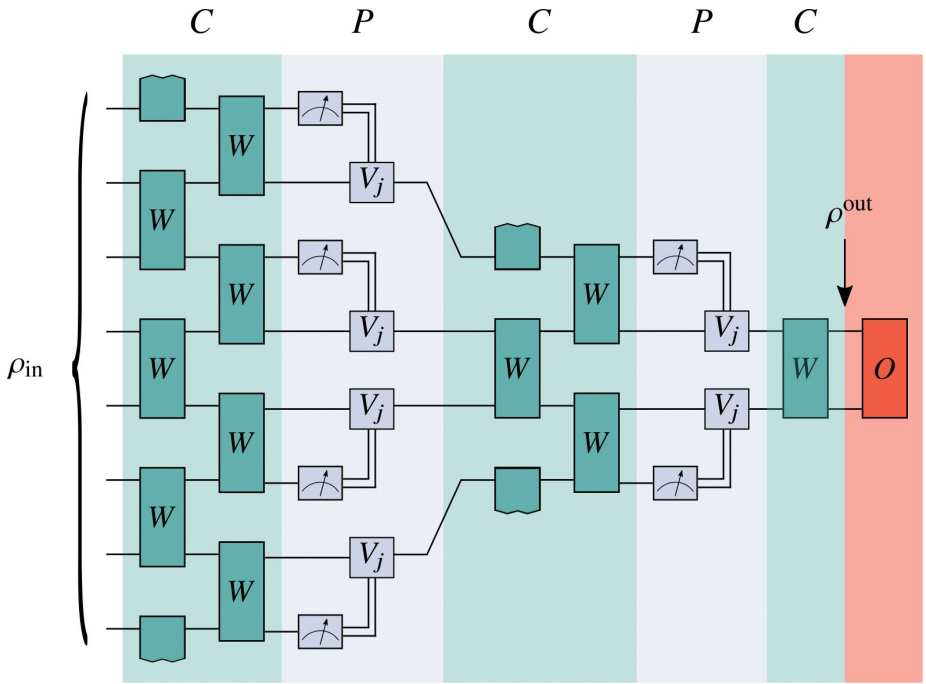
Cerezo et al, *Cost-Function-Dependent Barren Plateaus in Shallow Quantum Neural Networks*, arXiv:2001.00550

Intuition: QCNN won't have any barren plateau because of the logarithmic number of layers

Problem: the QCNN is **not** the hardware-efficient ansatz => we have to do some work

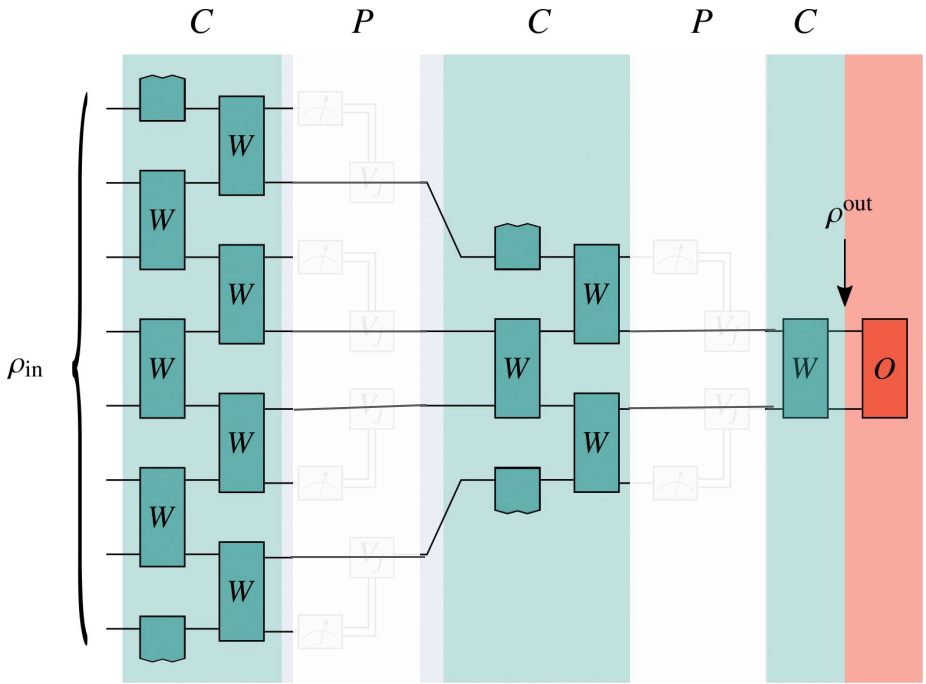
How to study this architecture?

First simplification: modify the circuit to avoid conditional measurement between two blocks



How to study this architecture?

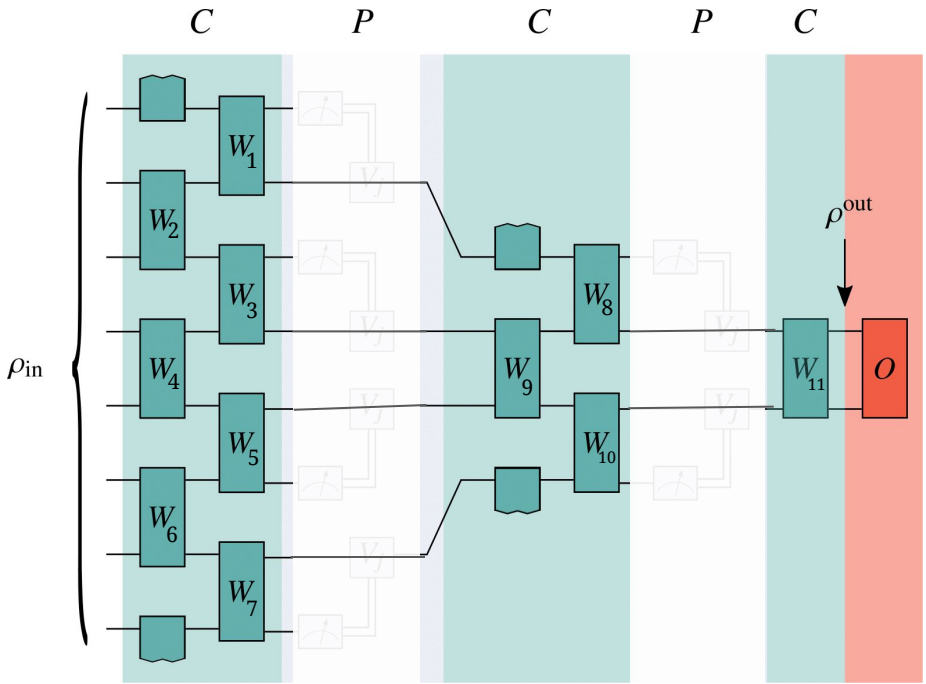
Second simplification: remove the conditional measurements (doesn't change the statistics)



Reason: property of the Haar measure: $\mu_{\text{Haar}}(WV) = \mu_{\text{Haar}}(W)$

How to study this architecture?

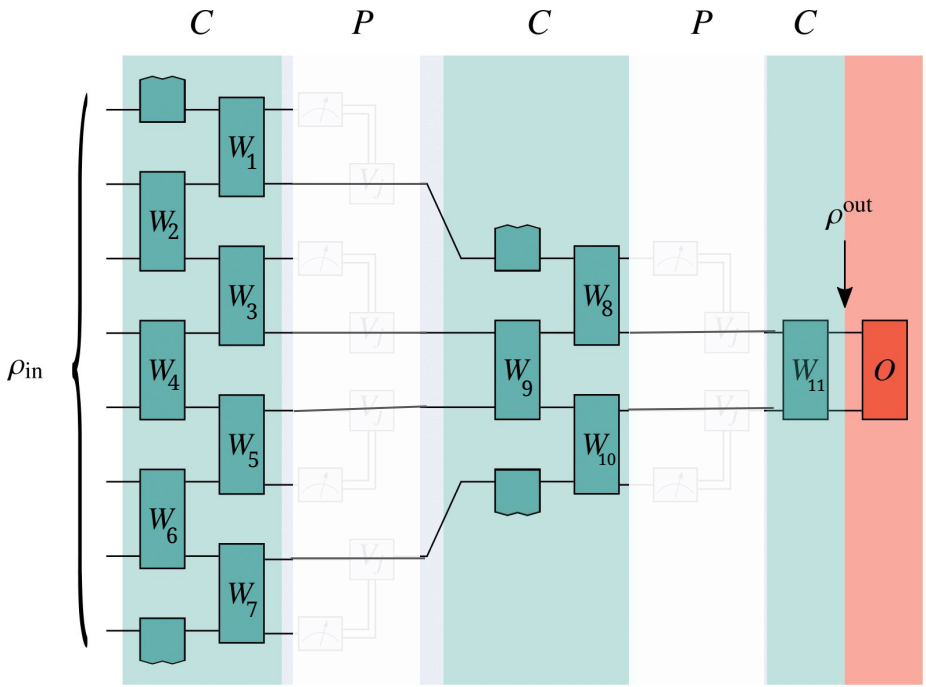
Third simplification: we decorrelate the unitaries



Reason: this should give a lower bound on the correlated version (mix of analytical and empirical evidence)

How to study this architecture?

Fourth simplification: linear cost-function with respect to the input state



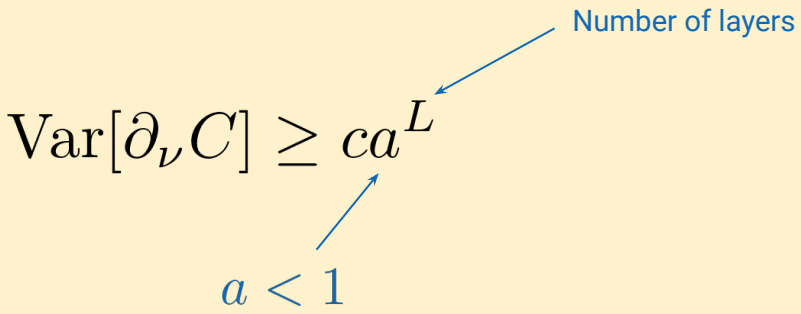
$$C(\theta) = \mathrm{Tr}[OU_\theta \rho_{\text{in}} U_\theta^\dagger]$$

Example: classification (0 vs 1)

$$C(\theta) = \frac{1}{M_0} \sum_{\rho_i \in \mathcal{S}_0} \mathrm{Tr}[U(\theta)\rho_i U^\dagger(\theta) O] - \frac{1}{M_1} \sum_{\rho_i \in \mathcal{S}_1} \mathrm{Tr}[U(\theta)\rho_i U^\dagger(\theta) O]$$

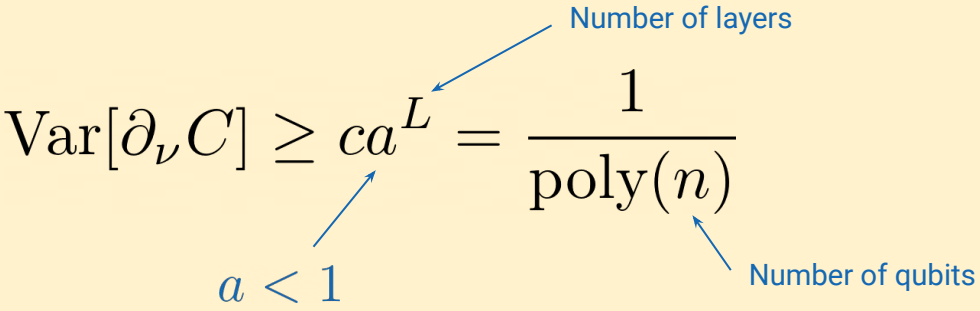
Result

Result: with the hypotheses described above, the variance of the QCNN has at worst an inverse polynomial scaling, i.e. it is trainable

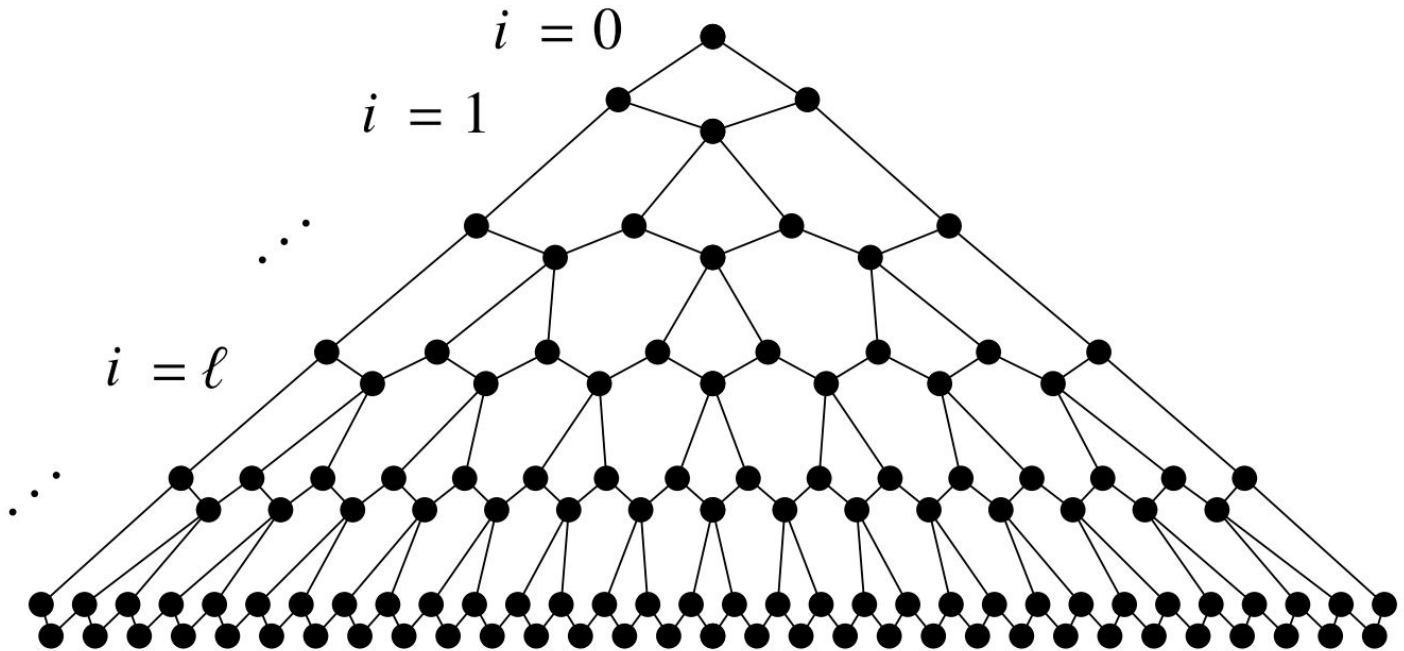


Result

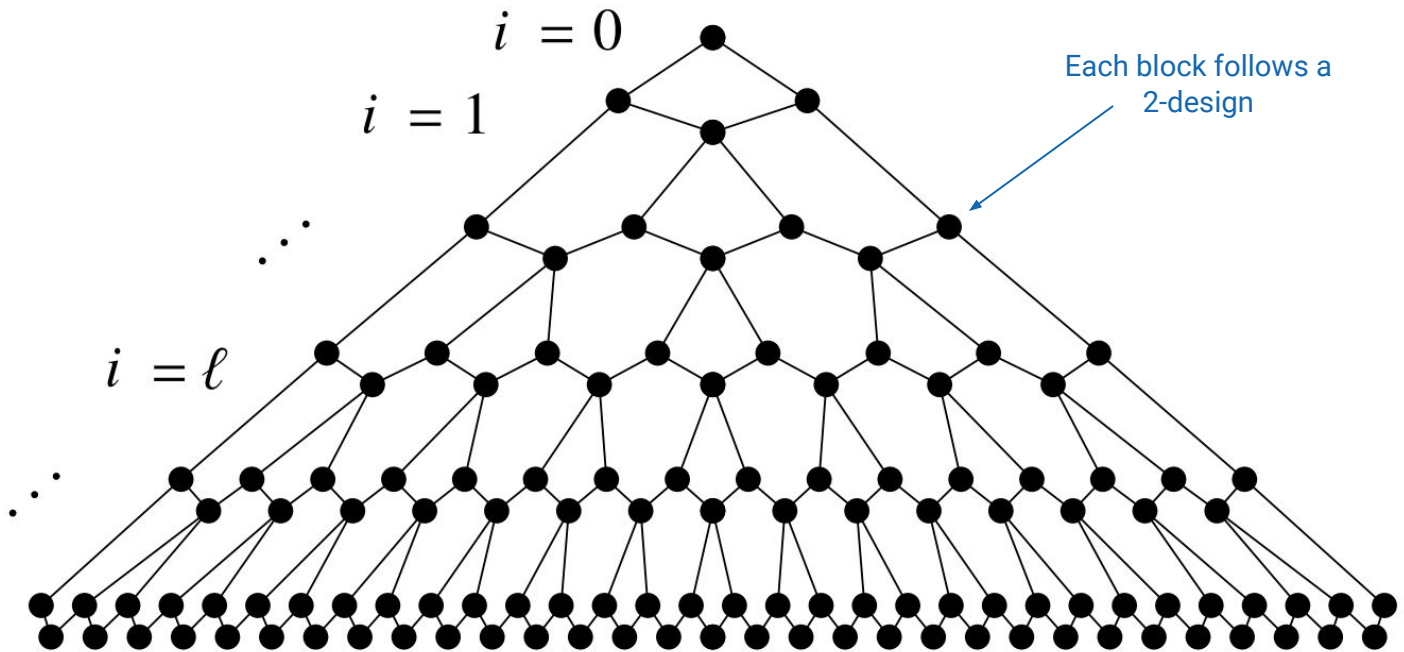
Result: with the hypotheses described above, the variance of the QCNN has at worst an inverse polynomial scaling, i.e. it is trainable



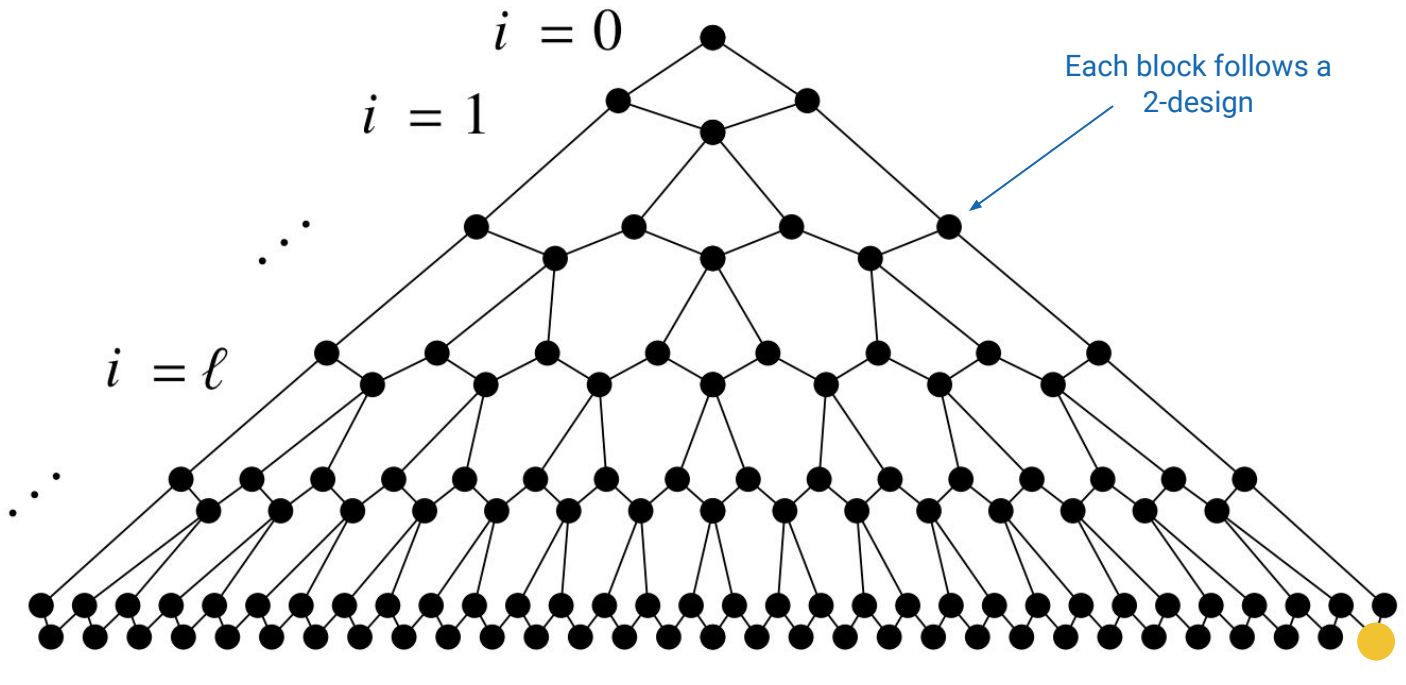
Proof: Graph Recursion Integration Method (GRIM)



Proof: Graph Recursion Integration Method (GRIM)

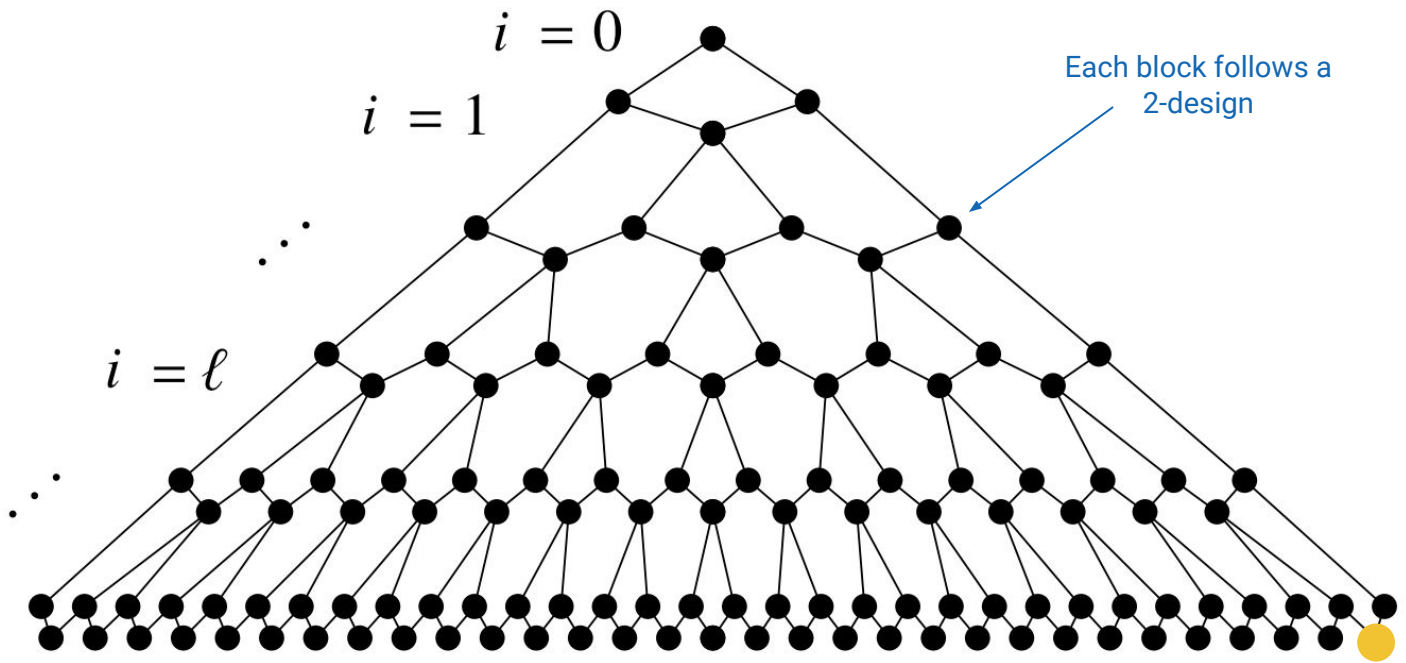


Proof: Graph Recursion Integration Method (GRIM)



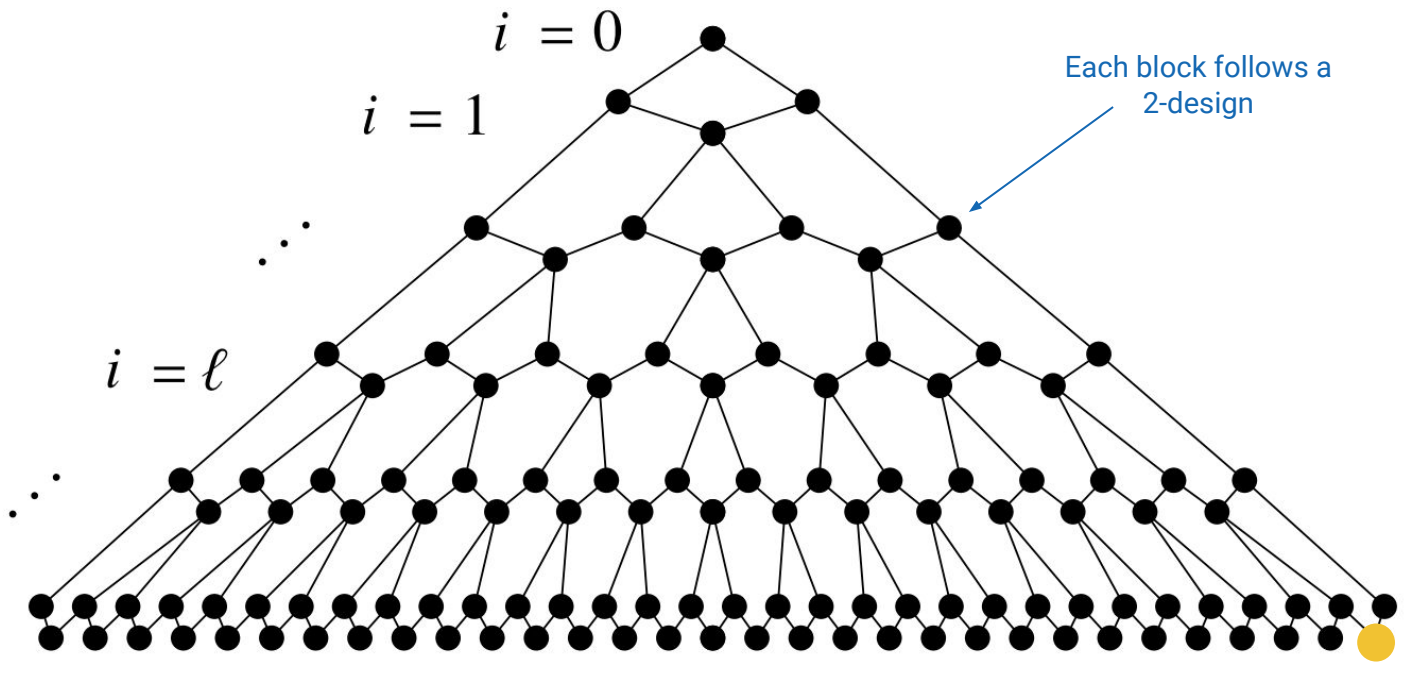
Proof: Graph Recursion Integration Method (GRIM)

Goal: obtain the variance of the gradient $\int_U |\partial_\nu C|^2$



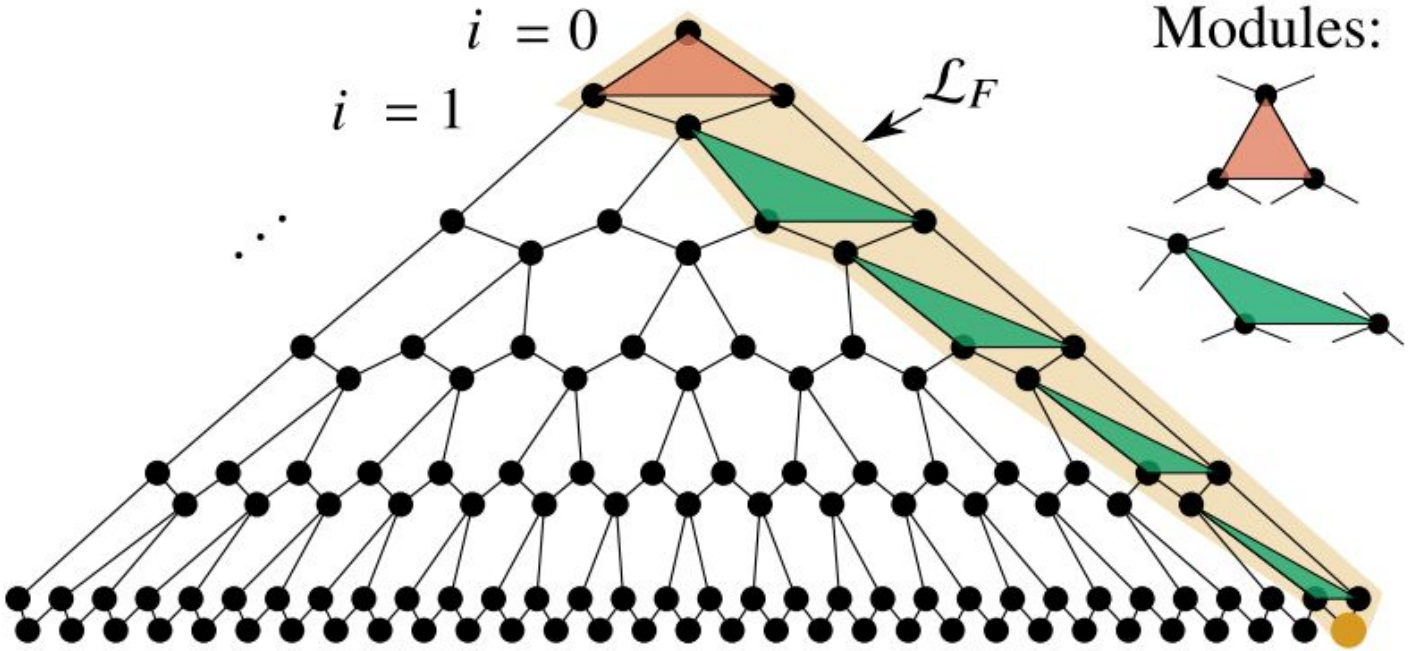
Proof: Graph Recursion Integration Method (GRIM)

Problem: way too many terms to integrate



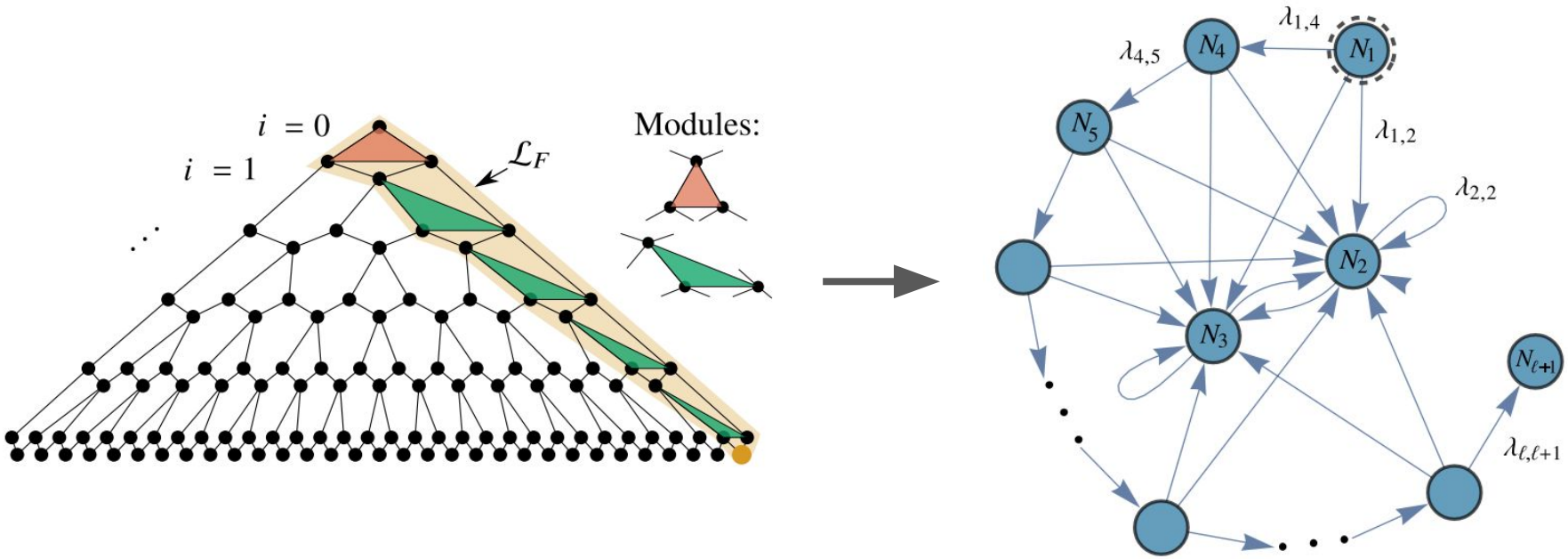
Proof: Graph Recursion Integration Method (GRIM)

Idea: causal cone + module decomposition



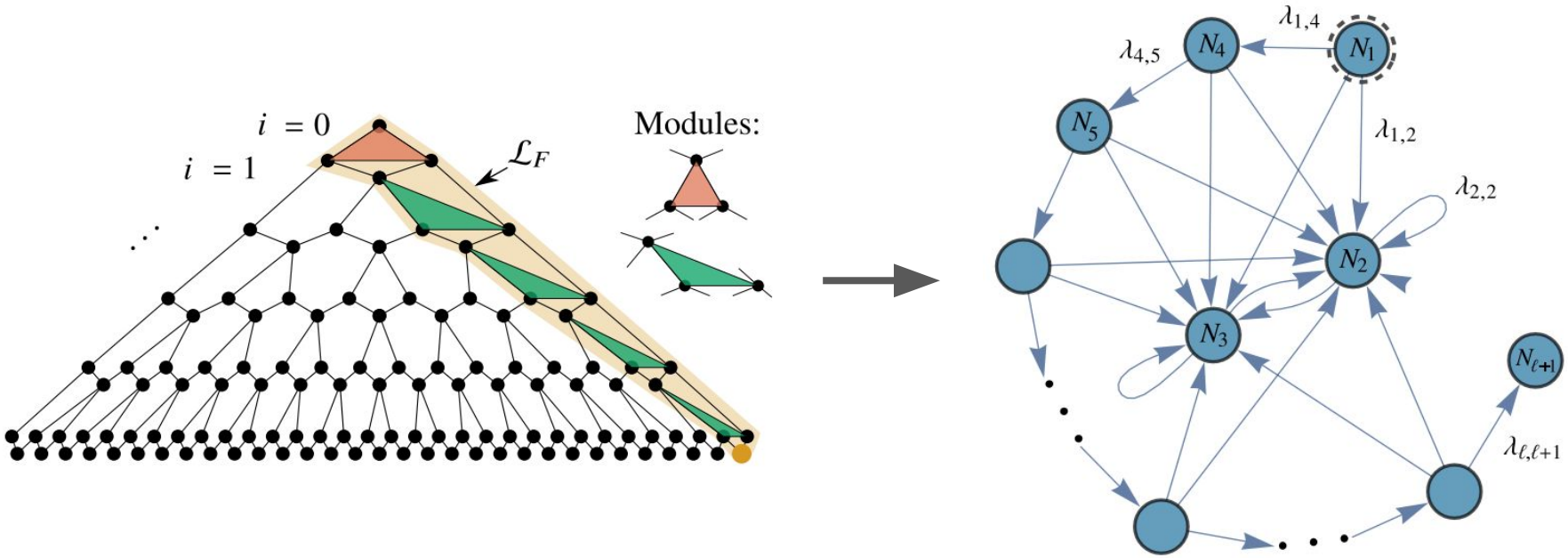
Proof: Graph Recursion Integration Method (GRIM)

Method: keep track of all the integrals in a graph



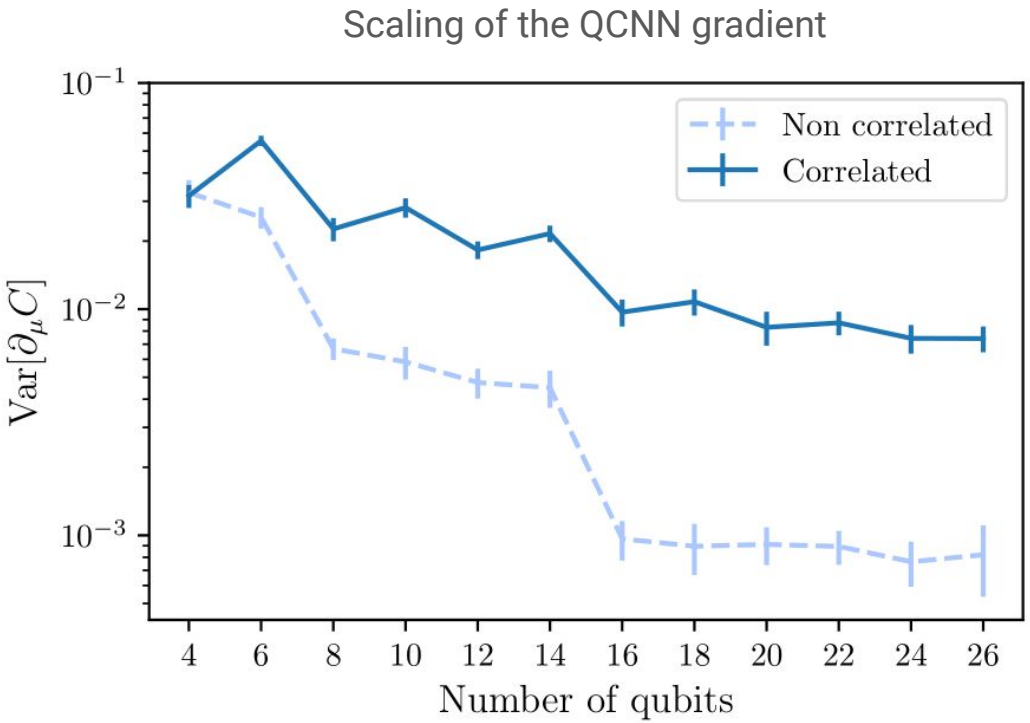
Proof: Graph Recursion Integration Method (GRIM)

Result: variance = sum over all paths of length log(n) in the graph



Numerical experiments

Simulator: Cirq + TensorFlow Quantum



Future work

- How to study correlated architectures systematically?
- How to study non-linear cost-functions? (e.g. mean-squared error)
- Can we deduce the trainability of MERA from our result?
- Can we use GRIM to analyze other architectures
- How useful are QCNNs in practice?

Thanks for your attention