Ph 220: Quantum Learning Theory
Lecture Note 5: Learning Quantum Neural Networks and Short-Range-Entangled States

Hsin-Yuan Huang (Robert)

Caltech

1 Introduction: Learning Quantum Neural Networks

This lecture transitions from the fundamentals of quantum learning to the practical challenges of learning and training quantum neural networks. We focus on Quantum Neural Networks (QNNs), which are parameterized quantum circuits that serve as a quantum analogue of classical neural networks. Consider the following learning problem. Given a training set $T = \{(|\psi_i\rangle, |\phi_i\rangle)\}_{i=1}^N$ of input-output quantum state pairs, how can we learn a parameterized unitary $U(\vec{\theta})$ such that

$$|\phi_i\rangle \approx U(\vec{\theta})|\psi_i\rangle$$

for all $i \in [N]$? This is a foundational task in quantum machine learning with applications ranging from quantum compilation to learning physical dynamics.

The training set T may be provided in two primary formats:

1. **Quantum data:** Multiple copies of the quantum states $|\psi_i\rangle$ and $|\phi_i\rangle$ are provided directly (e.g., stored in quantum memory).
2. **Classical data:** Classical descriptions of the input states $|\psi_i\rangle$ (e.g., simple product states like $|0\ldots0\rangle$) along with classical measurement data from the target states $|\phi_i\rangle$ (e.g., classical shadows or other classical representations).

For either case, a standard approach defines a loss function and employs an optimization algorithm such as gradient descent to determine the optimal parameters $\vec{\theta}$.

2 The Challenge: Barren Plateaus

The central challenge in training QNNs is the **barren plateau** phenomenon: for most parameters, the loss landscape is flat, providing no gradient signal to guide optimization. To overcome this obstacle, we must first systematically understand the mechanisms that give rise to barren plateaus.

2.1 Mechanism 1: Global Loss Functions

A first attempt for defining the loss function to train the QNN is to use a **global loss function**, which measures the overall fidelity of the output state:

$$L_{\text{global}}(\vec{\theta}) = 1 - \frac{1}{N}\sum_{i=1}^{N} |\langle\phi_i|U(\vec{\theta})|\psi_i\rangle|^2.$$

This approach fails even for extremely simple QNNs. Consider a depth-1 circuit $U(\vec{\theta}) = \bigotimes_{j=1}^{n} e^{-i\theta_j \sigma_x}$ over n qubits. Let the training set be given by $|\psi_i\rangle \in \{|0\rangle, |1\rangle\}^{\otimes n}$ and $|\phi_i\rangle = U(\vec{\theta}^*)|\psi_i\rangle$ for some target parameters $\vec{\theta}^*$. The loss function takes the form

$$L_{\text{global}}(\vec{\theta}) = 1 - \prod_{j=1}^{n} \cos^2(\theta_j - \theta_j^*).$$

For almost all $\vec{\theta}$, this loss is exponentially close to 1. Since we cannot measure fidelity to exponentially small error in polynomial time, we have no way to observe meaningful gradients in the landscape. This causes any local optimization method to fail, creating what is known as a **cost-function-induced barren plateau**.

Conclusion 1: Global loss functions are not a viable path for training QNNs due to exponentially many bad local minima and exponentially flat landscapes.

2.2 Mechanism 2: Polylog-Depth Circuits

The failure of global loss functions suggests we should try a **local loss function**, which only depends on a few qubits. A typical local loss is of the form

$$L_{\text{local}}(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \text{tr}\left(V_i \left(\sum_{j=1}^{n} |1\rangle\langle 1|_j \otimes I_{-j} \right) V_i^\dagger \cdot U(\vec{\theta})|\psi_i\rangle\langle\psi_i|U^\dagger(\vec{\theta}) \right),$$

where we assume the output state $|\phi_i\rangle = V_i|0^n\rangle$ is given via a classical description of the quantum circuit V_i, and $|1\rangle\langle 1|_j \otimes I_{-j}$ is the single-qubit projector on the j-th qubit.

Consider again the simple depth-1 circuit $U(\vec{\theta}) = \bigotimes_{j=1}^{n} e^{-i\theta_j \sigma_x}$ with the training set $|\psi_i\rangle \in \{|0\rangle, |1\rangle\}^{\otimes n}$ and $|\phi_i\rangle = U(\vec{\theta}^*)|\psi_i\rangle$. We have

$$L_{\text{local}}(\vec{\theta}) = \sum_{j=1}^{n} \sin^2(\theta_j - \theta_j^*).$$

This optimization landscape has only the global minimum $\vec{\theta}^*$ as a stable critical point. Hence, stochastic gradient descent will converge to the global optimum. We can see that the local loss fixes the problem of the global loss for this simple, depth-1 QNN.

However, what about learning higher-depth QNNs? Consider a 1D QNN with circuit depth $d = \text{poly}(\log n)$. Recent work has proven that a random quantum circuit of depth $d = \text{poly}(\log n)$ in any geometry (including 1D) is computationally indistinguishable from Haar-random unitaries. To understand the optimization landscape, we analyze its concentration by calculating the expectation and variance using moment calculations for Haar-random unitaries.

Expectation Value. The expectation of the loss, $\mathbb{E}_{\vec{\theta}}[L_{\text{local}}(\vec{\theta})]$, is given by:

$$\mathbb{E}_{\vec{\theta}}[L_{\text{local}}(\vec{\theta})] = \frac{1}{N} \sum_{i=1}^{N} \text{tr}\left(V_i \left(\sum_{j=1}^{n} |1\rangle\langle 1|_j \otimes I_{-j} \right) V_i^\dagger \cdot \mathbb{E}[U(\vec{\theta})|\psi_i\rangle\langle\psi_i|U^\dagger(\vec{\theta})] \right).$$

Using the 1-design property $\mathbb{E}[U(\vec{\theta})|\psi\rangle\langle\psi|U^\dagger(\vec{\theta})] = I/2^n$, the expectation becomes:

$$\mathbb{E}_{\vec{\theta}}[L_{\text{local}}(\vec{\theta})] = \frac{n}{2}.$$

Here, we exploit the fact that $U(\vec{\theta})$ is effectively Haar-random, allowing us to use the t-design properties of Haar-random unitaries to obtain these statistical properties.

Variance. To understand *how* flat the landscape is, we must calculate the variance:

$$\mathrm{Var}[L_{\mathrm{local}}(\vec{\theta})] = \mathbb{E}[L_{\mathrm{local}}(\vec{\theta})^2] - (\mathbb{E}[L_{\mathrm{local}}(\vec{\theta})])^2.$$

Applying the 2-design formula, this calculation yields:

$$\mathrm{Var}[L_{\mathrm{local}}(\vec{\theta})] = \frac{n^2}{4} \cdot \frac{1 + 2^{n+1}}{2^{2n} - 1}.$$

The variance decays exponentially in system size n. Since a 1D random quantum circuit of depth $\mathrm{poly}(\log n)$ is computationally indistinguishable from Haar-random unitaries (distinguishing requires superpolynomial time), the variance is smaller than $1/\mathrm{poly}(n)$. Hence, the variance is superpolynomially small and becomes exponentially small at linear depth.

Conclusion 2: A superpolynomially small variance means the landscape is superpolynomially flat almost everywhere. Therefore, *even a local loss function fails* if the circuit is slightly deeper ($d = \mathrm{poly}(\log n)$). Because the QNN is computationally indistinguishable from Haar-random unitaries, there is fundamentally nothing we can learn about the system using polynomial-time methods. This produces a **depth-induced barren plateau** that is independent of the choice of loss function.

2.3 Mechanism 3: Standard Parameterization with Constant Depth

From Conclusions 1 and 2, we have established that our solution *must* use a **local loss function** and a **constant-depth circuit** (i.e., shallow QNNs). Is this combination sufficient?

The answer is **No**. Even with the correct combination of local loss and constant depth, the standard parameterization produces an intractable landscape:

- The standard landscape for shallow QNNs remains extremely bad, with exponentially many local minima that traps gradient descent methods.
- Empirical evidence confirms this: the standard parameterization shows a probability of success that drops to zero as the number of qubits n increases.

See the accompanying slides to see how this mechanism arises visually.

Conclusion 3: The combination of a local loss and a constant-depth circuit still fails if we use a standard parameterization (where we parameterize each gates in our quantum neural network). The landscape itself is intractably bad, despite having no barren plateau in the strict sense (extremely flat landscape), the landscape is still reminiscent of barren plateau (highly suboptimal landscape for almost the entire parameter space but now with many ups and downs).

3 The Solution: Parameterization by Local Inversion

We have systematically identified the mechanisms that can cause QNNs to fail. The only remaining path is to change the **parameterization** itself.

Key Idea: Instead of a standard parameterization, we use **parameterization by local inversion**.

Definition 1 (Local Inversion)**.** *Given an n-qubit unitary U, a* ***local inversion*** *V_j* ***of*** *U* ***for qubit*** *j is a unitary operation such that the composite operation $U \cdot V_j$ acts as the identity on qubit j. The action on the remaining qubits may be arbitrary.*

This is the final, crucial piece of the puzzle. We use a **local loss function** to train a **constant-depth** QNN parameterized by its **local inversions** $V_j(\vec{\theta}_j)$.

3.1 Why Local Inversion Works

The power of local inversion lies in two fundamental properties:

1. **Unique determination:** The complete set of local inversions $\{V_1, \ldots, V_n\}$ uniquely determines the global unitary U.
2. **Efficient reconstruction:** The global unitary can be reconstructed from these local pieces through a systematic "sewing" procedure that iteratively combines the inversions.

To see how one can efficiently reconstruct the global unitary U from the local inversions, please refer to the accompanying slides. Here is the concise formula for sewing the local inversion to form the n-qubit unitary U using $2n$ qubits (first n qubits are initialized in the maximally mixed state and the second half corresponds to any state we would like to apply to),

$$\mathcal{E}_{\{V_i\}_{i=1}^n}(\rho) = \mathrm{Tr}_{>n}\left(\prod_{i=1}^{n}(\mathcal{V}_i^\dagger \otimes \mathcal{I})\mathcal{SWAP}_{i,n+i}(\mathcal{V}_i \otimes \mathcal{I})\left(\frac{I_n}{2^n} \otimes \rho\right)\right) = U\rho U^\dagger, \tag{1}$$

where $\mathcal{V}_i(\sigma) = V_i \sigma V_i^\dagger$, $\mathcal{I}(\sigma) = \sigma$, and $\mathcal{SWAP}_{i,n+i}$ is the quantum channel implementing a swap between qubit i and qubit $n+i$. This implements the same procedure as the one in the slides. Also, note that the ordering from $i = 1$ to n can be arbitrary. One can apply the local inversion in any order one wants. The ideal ordering is one that can be highly parallelized by applying non-overlapping local inversions simultaneously.

By parameterizing the local inversions $V_j(\vec{\theta}_j)$ rather than the global unitary $U(\vec{\theta})$, we change the optimization problem. Each local inversion can be optimized using a local loss function that measures how well $U \cdot V_j$ acts as the identity on qubit j. This local loss has favorable structure.

3.2 Improved Loss Landscape

The specific combination of local loss, constant depth, and local inversion parameterization is what enables efficient learning:

- For each local inversion training/learning, there will only be a constant number of parameters in the QNNs involved (those in the light cone of qubit j).
- Because the optimization problem is effectively over constant number of parameters, the landscape can be shown to yield only a constant number of local minima. Hence, by performing gradient-based optimization with restarts, one can find the global minimum efficiently.

This leads to the central theorem:

Theorem 1 (Provably Efficient Learning)**.** *Any n-qubit shallow QNN can be learned to error ϵ in $poly(n, 1/\epsilon)$ time using parameterization by the local inversion.*

Empirical evidence shows that local inversion methods maintain high probability of success as n grows, in stark contrast to standard methods; see the accompanying slides.

4 Parallel Problem: Learning Short-Range-Entangled States

The local inversion framework extends beyond learning unitaries. A parallel and powerful application is the learning of **Short-Range-Entangled (SRE) states** defined as states that can be generated by a constant-depth quantum circuit in a finite-dimensional lattice of qubits.

This leads to the following question: Given access to copies of an unknown SRE state $|\psi\rangle$, how can we efficiently learn a description for generating the state?

The approach is analogous to learning shallow QNNs, but adapted for states:

1. **Learn Local RDMs:** Perform local measurements to learn the reduced density matrices (RDMs) $\rho_{B\cup\partial B} = \mathrm{tr}_{\overline{B\cup\partial B}}(|\psi\rangle\langle\psi|)$ on overlapping blocks B and their boundaries ∂B.

2. **Find Local Inversion:** For each block B, find a local inversion (or disentangling) unitary V_B acting on $B \cup \partial B$ that transforms the local RDM into the all-zero product state $|0_B\rangle$ on B (but can be an arbitrary state on the boundary ∂B).

3. **Sew the Inversion:** These local disentangling operations V_B can be "sewn" together to construct a global unitary that prepares the SRE state. To see this, consider a 1D system (this can be easily generalized to higher dimension), and the overlapping blocks are $B_1, \ldots, B_m$ (ordered from left to right). Let these blocks be big enough such that the the boundary of B_i does not touch B_{i-2}, B_{i+2} (but could touch B_{i-1}, B_{i+1}). Let

$$\mathcal{E}_i(\cdot) = V_{B_i}^\dagger \left(|0_{B_i}\rangle\langle 0_{B_i}| \otimes \mathrm{tr}_{B_i}(V_{B_i}(\cdot)V_{B_i}^\dagger) \right) V_{B_i} \tag{2}$$

From the definition of local inversion, we have the following identity:

$$\mathcal{E}_i(|\psi\rangle\langle\psi|) = V_{B_i}^\dagger \left(|0_{B_i}\rangle\langle 0_{B_i}| \otimes \mathrm{tr}_{B_i}(V_{B_i}|\psi\rangle\langle\psi|V_{B_i}^\dagger) \right) V_{B_i} = |\psi\rangle\langle\psi|. \tag{3}$$

Using this identity, we have

$$(\mathcal{E}_2 \circ \mathcal{E}_4 \circ \ldots) \circ (\mathcal{E}_1 \circ \mathcal{E}_3 \circ \ldots)\,(|\psi\rangle\langle\psi|) = |\psi\rangle\langle\psi|. \tag{4}$$

Furthermore, as one makes the blocks to be a big enough constant and overlapping enough, then the tr_{B_i} operations will cause the initial state $|\psi\rangle\langle\psi|$ being traced out. Hence, we can place whatever state we want to obtain

$$(\mathcal{E}_2 \circ \mathcal{E}_4 \circ \ldots) \circ (\mathcal{E}_1 \circ \mathcal{E}_3 \circ \ldots) \left(\frac{I}{2^n} \right) = |\psi\rangle\langle\psi|. \tag{5}$$

This enables us to generate the unknown state $|\psi\rangle\langle\psi|$ from the local inversions.

This provides a constructive and efficient algorithm for learning SRE states by focusing only on local information, thereby avoiding the "exponential curse of dimensionality" associated with the full Hilbert space. The key insight is that SRE states have limited entanglement range, which allows their description to be decomposed into local components that can be learned independently.

5 Application 1: Generative Quantum Advantage

The efficient learning algorithm developed through local inversions serves as a key enabler for demonstrating **generative quantum advantage**. A **generative model** learns an unknown probability distribution $p(y|x)$ from a training dataset $\{(x_i, y_i)\}_{i=1}^N$, subsequently generating new outputs y for previously unseen inputs x.

A **generative QNN** implements this task through quantum circuits:

1. Encode the classical input x into a quantum state $|\psi_x\rangle$ and specify a measurement basis M_x.
2. Apply a trainable quantum circuit $U(\vec{\theta})$ to evolve the state.
3. Measure $U(\vec{\theta})|\psi_x\rangle$ in the basis M_x to sample an output y from the distribution $p(y|x;\vec{\theta})$.

To establish generative quantum advantage, we must demonstrate three essential properties:

1. **Efficient quantum learning:** Quantum computers can efficiently learn the distribution from polynomial-sized training sets.
2. **Efficient quantum generation:** Quantum computers can efficiently generate new outputs.
3. **Classical hardness:** Classical computers cannot efficiently generate these outputs.

Let's see how we can establish these three key ingredients.

Classical hardness: As we have seen from previous lectures, constant-depth QNNs can generate classical distributions that are classically hard to sample from under the widely accepted conjecture that the polynomial hierarchy does not collapse.

Efficient generation: By construction, shallow (constant-depth) QNNs are low-depth circuits that execute efficiently on quantum computers.

Efficient learning: This was the missing component. As we showed in Section 2, standard training methods fail even for shallow circuits. However, the local inversion method (Section 3) provides a provably efficient learning algorithm, completing the picture.

Therefore, generative quantum advantage is achievable. The local inversion algorithm furnishes the (previously absent) efficient training method for QNNs that are classically hard to simulate and quantumly easy to execute. This has been validated experimentally, demonstrating that learned QNNs can outperform classical models such as XGBoost and Transformers while achieving sampling fidelities that surpass classical computational limits (see the accompanying slides).

6 Application 2: Compressing Quantum Dynamics

The framework of learning and sewing local inversions extends naturally to compressing physical dynamics. Consider the following task: given a target unitary U corresponding to a very deep circuit (e.g., arising from long-time evolution of a physical system), find a low-depth circuit that accurately approximates its action. The method proceeds through three steps:

1. **Decompose:** Decompose a shallow QNN into spatially-overlapping components $U_1, U_2, \ldots, U_k$.
2. **Learn:** Apply the local inversion method to learn each component individually. Each piece is learned using local loss functions that avoid barren plateaus.
3. **Sew:** Sew the learned local pieces together to construct the trained shallow QNN. The sewing procedure combines the local inversions to reconstruct the global dynamics.

This reconstruction can be implemented in substantially shallower depth than the original circuit by judiciously employing ancilla qubits and SWAP operations. As long as a shallow quantum circuit that approximates the target unitary U, this procedure will provably succeed. This approach has been validated experimentally, demonstrating that compressed models can accurately reproduce the dynamics of complex quantum systems while requiring significantly reduced circuit depth. For instance, dynamics that originally required depth proportional to evolution time (but have a slow-growing light cone) can be compressed to low circuit depth, making them much easier to implement.