# Supplemental Information for “Superior resilience to poisoning and amenability to unlearning in quantum machine learning”

Yu-Qin Chen[1, *] and Shi-Xin Zhang[2, †]

[1] *Graduate School of China Academy of Engineering Physics, Beijing 100193, China*
[2] *Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China*

(Dated: August 12, 2025)

## I. Analytical Derivation of Landscape Curvature from Minimal Models

To provide a rigorous theoretical foundation for the empirically observed differences in landscape stability, we analyze minimal yet faithful versions of our classical and quantum models. These simplified models, which still incorporate the core sigmoid activation and binary cross-entropy loss, are analytically tractable and reveal the fundamental mechanism behind the QNN’s superior resilience.

### A. The Minimal Models

We consider a single data point $(x, y)$, where the feature $x$ is a scalar and the label $y \in \{0, 1\}$.

*a. Classical Model: Single Neuron with Sigmoid* This model serves as a one-parameter MLP. The pre-activation output (logit) is given by $z = \theta x$. The predicted probability is then $p = \sigma(z) = (1 + e^{-\theta x})^{-1}$, where $\sigma(\cdot)$ is the sigmoid function. The loss is the binary cross-entropy:

$$\mathcal{L}_{MLP}(\theta) = -[y \log(p) + (1 - y)\log(1 - p)]. \tag{S1}$$

*b. Quantum Model: Single-Qubit Classifier with Sigmoid* This is one of the simplest possible QNNs. The feature $x$ is encoded into a quantum state via a rotation, and a single trainable parameter $\theta$ defines the classification boundary. The pre-activation logit is derived from a quantum measurement, $z = \cos(\pi x + \theta)$. This functional form is a direct consequence of a standard and fundamental single-qubit classifier architecture involving three steps: angle encoding, a parameterized rotation, and a final measurement.

1. **State Encoding:** We encode a normalized scalar feature $x$ into the state of a qubit using angle encoding. Starting from the ground state $|0\rangle$, we apply a rotation gate, for instance $R_Y(\pi x)$. The input state becomes:

$$|\psi_{\text{in}}\rangle = R_Y(\pi x)|0\rangle = \begin{pmatrix} \cos(\pi x/2) \\ \sin(\pi x/2) \end{pmatrix}. \tag{S2}$$

2. **Parameterized Processing:** Next, a learnable, parameterized gate is applied. We use a rotation $R_Y(\theta)$ with a single trainable parameter $\theta$. The output state is $|\psi_{\text{out}}\rangle = R_Y(\theta)|\psi_{\text{in}}\rangle$. This simplifies to:

$$|\psi_{\text{out}}\rangle = R_Y(\pi x + \theta)|0\rangle. \tag{S3}$$

3. **Measurement:** To obtain a classical output logit $z$, we compute the expectation value of a standard observable, the Pauli-Z operator $\sigma_z$. The expectation value is calculated as:

$$z = \langle\psi_{\text{out}}|\sigma_z|\psi_{\text{out}}\rangle = \cos^2\left(\frac{\pi x + \theta}{2}\right) - \sin^2\left(\frac{\pi x + \theta}{2}\right) = \cos(\pi x + \theta). \tag{S4}$$

The probability is subsequently calculated as $p = \sigma(z) = \sigma(\cos(\pi x + \theta))$. The loss function is identical to the classical case:

$$\mathcal{L}_{QNN}(\theta) = -[y \log(p) + (1 - y)\log(1 - p)]. \tag{S5}$$

* yqchen@gscaep.ac.cn
† shixinzhang@iphy.ac.cn

## B. Analytical Derivation of the Hessian

The Hessian, $\mathbf{H} = \frac{d^2\mathcal{L}}{d\theta^2}$, represents the local curvature of the loss landscape. We use the chain rule for its derivation. A key property of the cross-entropy loss is that its gradient with respect to the parameter $\theta$ simplifies to:

$$\frac{d\mathcal{L}}{d\theta} = \frac{d\mathcal{L}}{dp}\frac{dp}{dz}\frac{dz}{d\theta} = \left(\frac{p-y}{p(1-p)}\right) \cdot [p(1-p)] \cdot \frac{dz}{d\theta} = (p-y)\frac{dz}{d\theta}. \tag{S6}$$

### 1. Hessian of the Single-Neuron MLP

For the MLP, we have $z = \theta x$, which implies $\frac{dz}{d\theta} = x$. The gradient is therefore $\frac{d\mathcal{L}_{MLP}}{d\theta} = (\sigma(\theta x) - y)x$. Differentiating this expression again with respect to $\theta$ yields the Hessian:

$$\begin{aligned} \mathbf{H}_{MLP} &= \frac{d}{d\theta}\left[(\sigma(\theta x) - y)x\right] & \text{(S7)}\\ &= \left[\frac{d}{d\theta}\sigma(\theta x)\right] \cdot x & \text{(S8)}\\ &= [\sigma'(\theta x) \cdot x] \cdot x = x^2\sigma'(\theta x) & \text{(S9)}\\ &= x^2\sigma(\theta x)(1 - \sigma(\theta x)) = x^2 p(1-p). & \text{(S10)} \end{aligned}$$

Since both $x^2$ and the term $p(1-p)$ are always non-negative, the curvature contribution from any single data point to the MLP's loss landscape is strictly non-negative.

At first glance, the expression $\mathbf{H}_{MLP} = x^2 p(1-p)$ appears to be independent of the label $y$. However, this dependence is implicit and strong. The Hessian is evaluated at the optimal parameter $\theta^*$ found during training, and $\theta^*$ is chosen specifically to minimize the loss for a given label $y$. For a clean dataset, the model finds a $\theta^*$ that pushes the prediction $p = \sigma(\theta^* x)$ towards saturation (0 or 1), a region where the term $p(1-p)$ is very small, resulting in a relatively flat minimum. When the training set contains conflicting labels for the same feature $x$, the model is forced to find a compromise parameter $\theta^*$ that results in an uncertain prediction ($p \approx 0.5$). It is precisely in this region of high uncertainty that the term $p(1-p)$ is maximized, leading to a sharp increase in the Hessian's trace. Thus, label noise forces the MLP into a region of high curvature.

### 2. Hessian of the Single-Qubit QNN

For the QNN, we have $z = \cos(\pi x + \theta)$, which implies $\frac{dz}{d\theta} = -\sin(\pi x + \theta)$. The gradient is $\frac{d\mathcal{L}_{QNN}}{d\theta} = (\sigma(\cos(\pi x + \theta)) - y)(-\sin(\pi x + \theta))$. To compute the Hessian, we differentiate this product using the product rule:

$$\begin{aligned} \mathbf{H}_{QNN} &= \frac{d}{d\theta}\left[(p-y)(-\sin(\pi x + \theta))\right] & \text{(S11)}\\ &= \left[\frac{d(p-y)}{d\theta}\right](-\sin(\pi x + \theta)) + (p-y)\left[\frac{d(-\sin(\pi x + \theta))}{d\theta}\right] & \text{(S12)}\\ &= \left[\frac{dp}{d\theta}\right](-\sin(\pi x + \theta)) - (p-y)\cos(\pi x + \theta). & \text{(S13)} \end{aligned}$$

We substitute the chain rule expansion for $\frac{dp}{d\theta} = \frac{dp}{dz}\frac{dz}{d\theta} = p(1-p)(-\sin(\pi x + \theta))$:

$$\begin{aligned} \mathbf{H}_{QNN} &= [p(1-p)(-\sin(\pi x + \theta))](-\sin(\pi x + \theta)) - (p-y)\cos(\pi x + \theta) & \text{(S14)}\\ &= \underbrace{p(1-p)\sin^2(\pi x + \theta)}_{\text{Term A: Information-Geometric Term}} - \underbrace{(p-y)\cos(\pi x + \theta)}_{\text{Term B: Error-Phase Interaction Term}}. & \text{(S15)} \end{aligned}$$

The QNN's Hessian is composed of two distinct parts. Term A, similar to the MLP's Hessian, is always non-negative. However, Term B's sign depends on both the prediction error $(p-y)$ and the quantum phase $\cos(\pi x + \theta)$. For a mislabeled point that the model correctly identifies as an outlier (e.g., $p \approx 1$ but $y = 0$), Term B can become strongly negative.

### C. Conclusion from the Analytical Model

This derivation reveals a fundamental asymmetry. The MLP's landscape is constructed from purely positive curvatures, forcing it to deform to fit noise. The QNN's landscape, due to the oscillatory nature of quantum measurements, contains a noise-adaptive component (Term B) that can introduce negative curvature. This allows the global Hessian to balance or cancel the influence of outliers, leading to a more stable landscape geometry. This analytical insight provides a rigorous theoretical mechanism that underpins the superior resilience and unlearning plasticity observed in our numerical experiments.

## II. Robustness of Learning Patterns Across Model Sizes

A potential concern regarding our main findings is whether the observed differences in resilience are merely an artifact of the specific model sizes chosen for comparison, rather than an intrinsic difference between the classical and quantum learning paradigms. To address this, we conducted a systematic study where we varied the capacity (or expressiveness) of both the MLP and QNN models and re-evaluated their performance under the label flipping corruption protocol on quantum XXZ dataset.

For the classical MLP, we varied its capacity by changing the number of neurons in its two hidden layers, ranging from a larger model with [64, 16] neurons to a much smaller one with [8, 2] neurons. For the quantum QNN, we varied its capacity by adjusting the depth of the parameterized quantum circuit, testing depths of 5, 6, and 7. The results of these experiments are presented in Fig. S1.

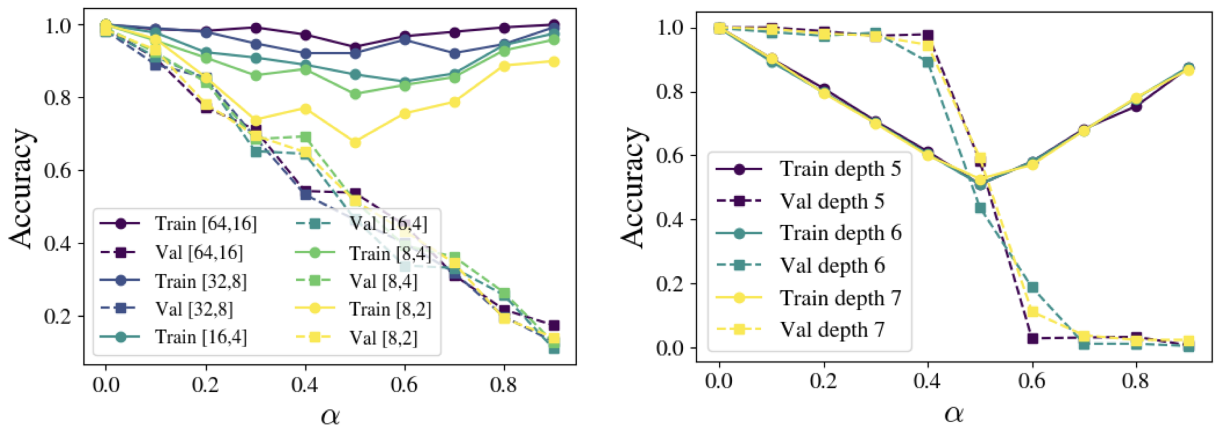

FIG. S1. **Resilience patterns are consistent across different model capacities.** Training (solid lines) and validation (dashed lines) accuracy as a function of the label flip noise ratio $\alpha$ for models of varying sizes. Left: Performance of MLPs with different hidden layer sizes (indicated in the legend). All models, regardless of capacity, exhibit the characteristic continuous compromise, where validation accuracy steadily degrades with increasing data noise. Right: Performance of QNNs with different circuit depths. All models, regardless of depth, display the distinct robust plateau followed by a critical transition, demonstrating that this behavior is an intrinsic feature of the quantum learning model.

The results unequivocally demonstrate that the learning patterns are an intrinsic property of the model class, not its size. As shown in Fig. S1(left), every MLP, from the largest to the smallest, exhibits the same qualitative behavior: a continuous compromise where validation accuracy begins to degrade immediately with the introduction of noise. While larger MLPs can better memorize the noisy training labels (as seen by their higher training accuracy), their generalization performance is similar to smaller MLPs.

Conversely, Fig. S1(right) shows that all QNNs, regardless of their circuit depth, display the characteristic pattern of resilience we identified in the main text. Each QNN maintains a robust performance plateau for a significant range of noise ratios before undergoing a sharp critical transition. This demonstrates that the QNN's ability to disregard noise and preserve a generalizable solution is not an accident of a specific parameter count but is fundamental to its learning nature.

Therefore, the stark qualitative difference between the behavior shown in Fig. S1 confirms that our main conclusion is robust. The contrast between brittle memorization and resilient generalization is an intrinsic feature distinguishing the classical and quantum learning paradigms investigated, not a mere consequence of model expressiveness or parameter count.

## III. Unlearning Efficacy on the Forget Set

To complement the validation accuracy results presented in the main text, we here provide a direct measure of unlearning efficacy on the forget set, $\mathcal{D}_{\text{forget}}$. For this analysis, we define a metric we call forgetting accuracy, which is the accuracy of the model when evaluated on the **polluted version** of the forget set, i.e., the set with the intentionally flipped labels that the original model was trained on.

A successful unlearning process should cause the model to forget the erroneous associations from the poisoned data. Therefore, a lower Forgetting Accuracy indicates a more effective unlearning procedure. A value of 1.0 means the model still perfectly remembers the incorrect, poisoned labels, while a value of 0.0 means the model has completely forgotten them and now classifies the samples based on their true features, which contradict the poisoned labels.

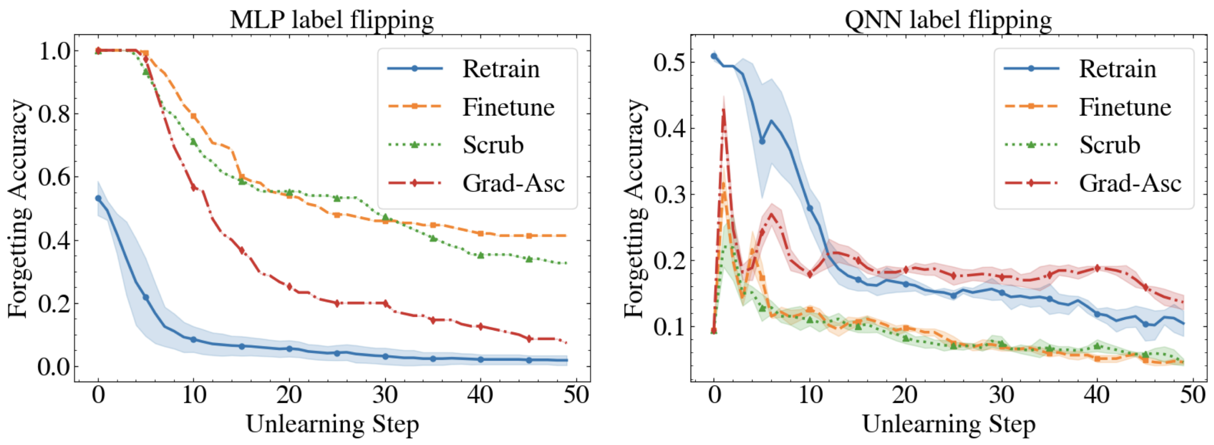

FIG. S2. **Forgetting Accuracy on the MNIST dataset.** The plot shows the evolution of forgetting accuracy on the polluted forget set versus the unlearning step. A rapid decrease to zero indicates effective unlearning.

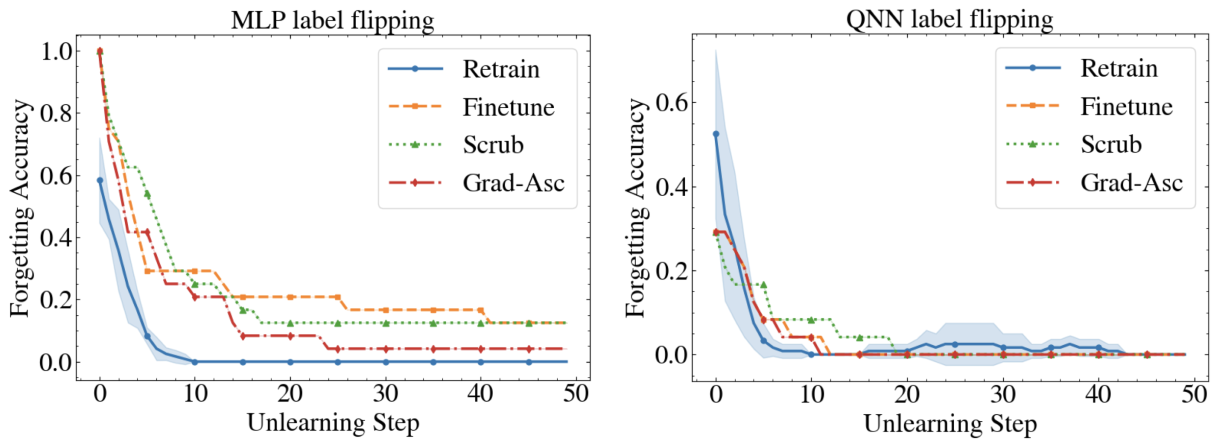

FIG. S3. **Forgetting Accuracy on the XXZ dataset.** The trends are consistent with the MNIST results. The MLP exhibits slow forgetting, indicative of stubborn memories. The QNN demonstrates rapid forgetting across all methods, indicative of high model plasticity.

The results, presented in Fig. S2 for the MNIST dataset and Fig. S3 for the XXZ dataset, provide direct evidence for the conclusions drawn in the main text.

For the classical MLP, we observe that while the Retrain baseline quickly achieves a forgetting accuracy near zero, the approximate methods lag significantly behind. Their forgetting accuracy decreases slowly and remains substantially above zero even after 50 unlearning steps. This large gap between the approximate methods and the baseline is a clear signature of the stubborn memories formed by the MLP; the incorrect knowledge is deeply entrenched and difficult to erase efficiently.

In contrast, the QNN demonstrates a much higher degree of model plasticity. All unlearning methods, including the approximate ones, show a rapid and dramatic decrease in forgetting accuracy. This indicates that the QNN's memory of the corrupted data is far less rigid and can be efficiently and effectively erased. These findings directly support our central thesis: the QNN's inherent resilience and the stable geometry of its loss landscape make it not only resistant to initial corruption but also highly amenable to subsequent correction.

## IV. Hyperparameters in architectures and setups

We evaluate and compare an MLP and a QNN on machine learning and unlearning tasks. The model architectures and hyperparameters for both the initial training and subsequent unlearning phases are detailed below.

### A. Model Architectures

*a. Classical Model (MLP):* The MLP is a fully connected neural network utilizing ReLU activation functions for hidden layers and a final Sigmoid activation for binary classification.

- **Hidden Units Number:** The architecture of the MLP varied across experiments.
  - For the `XXZ` dataset, the MLP consists of two hidden layers with 64 and 16 units, respectively.
  - For the `MNIST` dataset, a smaller MLP with 16 and 4 hidden units was used.

*b. Quantum Model (QNN):* The QNN is built on top of variational quantum circuits.

- **Circuit Depth:** The depth of the quantum circuit was also varied.
  - For the `XXZ` experiments, a circuit depth of 4 and 12 qubits was used.
  - For the `MNIST` experiments, a circuit depth of 6 and 10 qubits was used.

### B. Initial Training Process

The models were first trained on a dataset, which includes poisoned or mislabeled samples. All experiments were averaged over 5 independent runs for statistical robustness. The Adam optimizer was used for training all models. The specific hyperparameters for the two main experimental sets are detailed in Table I.

TABLE I. Hyperparameters for the initial training process.

| `XXZ` **Dataset Experiments** | | |
|---|---|---|
| **Parameter** | **MLP** | **QNN** |
| Epochs | 400 | 200 |
| Batch Size | 32 | 32 |
| Learning Rate | 0.01 | 0.03 |
| `MNIST` **Dataset Experiments** | | |
| **Parameter** | **MLP** | **QNN** |
| Epochs | 100 | 100 |
| Batch Size | 128 | 256 |
| Learning Rate | 0.01 | 0.005 |

### C. Unlearning Process and Hyperparameters

After initial training, various unlearning methods (`retrain`, `finetune`, `scrub`, `gradient ascent`) were applied.

- **Definition of an Unlearning Step:** An "unlearning step" is defined as a single, full epoch. The batch size for each unlearning step is set to the entire size of the retain set. This means the model processes all retain data in one pass for each step.
- **Unlearning Steps:** The unlearning process was run for 50 steps (`Epochs=50`).
- **Unlearning Optimizer/Hyperparameters:** The unlearning process also utilizes the Adam optimizer. The learning rate was set to 0.01 for the `XXZ` experiments and 0.001 for the `MNIST` experiments.
- **Unlearning Loss Weights:** The unlearning loss function is a weighted sum of multiple components, including a KL-divergence term ($\lambda_{kl}$), a cross-entropy term ($\lambda_{ce}$), and a forgetting term ($\lambda_{fo}$ or $\beta$). After preliminary experiments, we selected a set of weights that demonstrated strong performance and the specific weight configurations for each method are detailed below:
  - Retrain: This method involves retraining the model from scratch on the retain set only. The loss function is the standard cross-entropy on the retain set, effectively setting the unlearning-specific weights as: $\lambda_{ce} = 1.0$.
  - Finetune: This method finetunes the pre-trained model on the retain set. Similar to Retrain, the loss function is the standard cross-entropy, with the weights set to: $\lambda_{ce} = 1.0$.
  - Scrub: This method is a specific unlearning algorithm tested with the following weights: $\lambda_{kl} = 0.0$, $\lambda_{ce} = 1.0$, $\lambda_{fo} = 0.2$.
  - Grad-Asc: This gradient-based method was tested with the same weight configuration as the Scrub method: $\lambda_{ce} = 1.0$, $\beta = 0.2$.

## D. Hessian Matrix Evaluation

We use `MNIST` classification task to evaluate Hessian matrix for trained models. For methods requiring the computation of the Hessian matrix, a subset of the training data is used to make the calculation computationally tractable. The size of this subset is explicitly set to 100 samples. These samples are typically chosen randomly from the training set for each evaluation.

- **Training Data:** For each of the two classes of digits, we randomly select 250 samples.
- **Preprocessing:** The original $28 \times 28$ images are flattened into vectors. To accommodate the input requirements of the quantum model, these vectors are zero-padded to a dimension of 1024 and then L2-normalized.
- **Noise Injection:** To simulate a data poisoning scenario, label noise is introduced into the training set. The experiments are conducted across multiple noise levels, with noise ratios $\alpha$ of 20%, 30%, and 40%.