Chapter 10
Verifying Adversarial Robustness in Quantum Machine Learning: From Theory to Physical Validation Via a Software Tool

Ji Guan and Mingsheng Ying

Abstract As with classical neural networks, quantum machine learning (QML) models are vulnerable to small input perturbations that can significantly alter output predictions. Certifying the robustness of QML models, particularly on NISQ hardware, is therefore a fundamental step toward trustworthy quantum AI. This chapter reviews our recently developed comprehensive formal framework for *verifying adversarial robustness* in QML. The core of this framework is a fidelity-based robustness lower bound computable directly from the measurement outcome distribution, which enables both formal verification and empirical estimation on real quantum devices. Additionally, the optimal bound can be computed via semidefinite programming (SDP) with full knowledge of the quantum machine learning models. We incorporate these results into: (1) an efficient formal verification framework; (2) VERIQR, the first dedicated QML robustness verification tool; and (3) the first experimental benchmark of quantum adversarial robustness on a 20-qubit superconducting processor. Together, these systematic advances enable scalable, physically grounded robustness evaluation of QML models.

10.1 Introduction

Quantum machine learning (QML) has emerged as a promising paradigm at the intersection of quantum computing and artificial intelligence, offering the potential to accelerate data-driven tasks such as classification, pattern recognition, and quan-

J. Guan (✉)
Key Laboratory of System Software (Chinese Academy of Sciences), Institute of Software, Chinese Academy of Sciences, Beijing, China
e-mail: guanj@ios.ac.cn

M. Ying
Centre for Quantum Software and Information, University of Technology Sydney, Ultimo, NSW, Australia
e-mail: Mingsheng.Ying@uts.edu.au

M. Usman (ed.), *Quantum Robustness in Artificial Intelligence*, Quantum Science and Technology, https://doi.org/10.1007/978-3-032-11153-1_10

tum system identification [1, 2]. Enabled by rapid progress in quantum hardware platforms—particularly superconducting circuits—the implementation and empirical study of QML algorithms is advancing swiftly [3–10]. Industrial momentum is also building; for instance, Google has introduced *TensorFlow Quantum*, integrating quantum circuit training into the mainstream classical ML framework TensorFlow [11].

However, like their classical counterparts, QML models are susceptible to *adversarial perturbations*—subtle modifications to the input quantum state that can cause incorrect predictions or degrade performance [12, 13]. These perturbed inputs, known as *adversarial examples*, present a serious threat to the reliability and security of quantum AI, particularly in the noisy intermediate-scale quantum (NISQ) era where hardware noise is pervasive. Ensuring that QML models are robust against such perturbations is therefore a pressing concern [14].

While adversarial robustness has been widely studied in classical deep learning [15, 16], its quantum counterpart poses distinct challenges [17]. Quantum states are probabilistic, subject to decoherence and readout noise, and constrained by physical laws that restrict allowable perturbations to those preserving trace and positivity. This necessitates new verification frameworks grounded in quantum-specific geometry, such as fidelity-based distances.

In this chapter, we review our recently developed comprehensive framework for **verifying adversarial robustness in quantum machine learning**, grounded in rigorous theoretical analysis and experimental validation supported by software tool. Our approach spans three interlinked dimensions:

- **Theoretical Foundations**. We formalize robustness using fidelity-based metrics and introduce a sound, instance-specific *robustness lower bound* that can be computed directly from the measurement outcome distribution [18]. This bound enables certification even in black-box scenarios without model access, which is used in the physical validation. Additionally, we formulate the exact robustness radius as a semidefinite program (SDP), enabling provably optimal verification when the model is known. Based on these results, we design a set of verification algorithms that certify robustness under various information assumptions.
- **Algorithmic Realization**. We implement these ideas in VERIQR [19], the first dedicated tool for QML robustness verification. VERIQR supports both exact (sound and complete) and approximate robustness verification algorithms, and can identify quantum adversarial examples that may be used for adversarial training. The tool accepts models in OpenQASM 2.0 format and simulates realistic noise to reflect NISQ hardware characteristics, offering a unified benchmarking framework compatible with multiple quantum platforms.
- **Experimental Validation**. To assess practical viability, the first physical benchmark of QML robustness was conducted on a 20-qubit superconducting processor [20] based on our theoretical idea and using the VeriQR tool. It was demonstrated that the fidelity-based lower bound provides a tight and stable estimate of

robustness in the presence of real quantum noise and validates its correlation with robustness upper bounds obtained by adversarial attack methods and SDP-derived optimal bounds.

Together, the above components establish a scalable, hardware-compatible, and formally grounded approach to robustness in quantum machine learning. They show that rigorous certification is feasible on near-term devices and can be integrated seamlessly into the QML development lifecycle.

Organization. The remainder of this review is organized as follows. Section 10.2 introduces the necessary preliminaries, including quantum classifiers, fidelity-based distance measures, and the formal definition of the robustness verification problem. Section 10.3 develops a series of robustness bounds that form the theoretical foundation for solving the verification task. Section 10.4 presents verification algorithms derived from these bounds. Section 10.5 describes the design and functionality of the VERIQR tool, which implements these verification techniques. Section 10.6 details the experimental setup and results from benchmarking adversarial robustness on real superconducting quantum hardware using the robustness bounds. Finally, Sect. 10.7 concludes with a discussion of open problems and future research directions.

10.2 Preliminaries

This section presents the foundational concepts essential for verifying adversarial robustness in quantum machine learning. We begin by introducing the structure and semantics of quantum classifiers, followed by the definition of fidelity-based distance measures used to quantify adversarial perturbations. These perturbations can lead to adversarial examples, which are input states that induce misclassification. Building on these definitions, we then formalize the robustness verification problem, which forms the central focus of this chapter.

10.2.1 Quantum Classifiers

Let $\mathcal{H}$ be a 2^n-dimensional Hilbert space on an n-qubit quantum system. A *quantum state* $\rho \in \mathcal{D}(\mathcal{H})$ is a positive semidefinite operator ($\rho \succeq 0$) on $\mathcal{H}$ with trace one ($\mathrm{Tr}(\rho) = 1$). Here $\mathcal{D}(\mathcal{H})$ represents the set of quantum states on $\mathcal{H}$.

As illustrated in Fig. 10.1, a *quantum classifier* is a quantum algorithm that takes quantum input states and produces classical output labels, corresponding to predefined classes of interest. Formally, a quantum classifier over the Hilbert space $\mathcal{H}$ is defined as a pair:

$$\mathcal{A} = (\mathcal{E}, \{M_c\}_{c \in \mathcal{C}}),$$

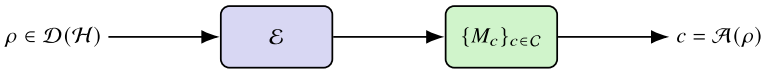

Fig. 10.1 Quantum classifier pipeline. The input quantum state ρ is processed by a quantum channel $\mathcal{E}$, followed by measurement via a POVM $\{M_c\}_{c\in C}$, to produce a classical class label $c = \mathcal{A}(\rho)$

where

- $\mathcal{E} : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{H})$ is a *quantum channel*, a completely positive and trace-preserving (CPTP) map, that models the quantum evolution (including noise) of the algorithm;
- $\{M_c\}c \in C$ is a *positive operator-valued measure* (POVM), where each $M_c \succeq 0$ and $\sum_c M_c = I$, used to perform quantum measurement and extract classical outputs;
- C is a finite set of possible output labels representing the classification categories.

Given an input quantum state $\rho \in \mathcal{D}(\mathcal{H})$, the classifier outputs a label determined by the most probable measurement outcome:

$$\mathcal{A}(\rho) := \arg\max_{c\in C} \mathrm{Tr}[M_c\mathcal{E}(\rho)],$$

where $\mathrm{Tr}[M_c\mathcal{E}(\rho)]$ is the probability of obtaining outcome c upon measuring the output state $\mathcal{E}(\rho)$ of $\mathcal{E}$ with the POVM $\{M_c\}_{c\in C}$.

10.2.2 Distance Between Quantum States

A central component of robustness analysis is the choice of a metric to quantify the similarity between quantum states. In the context of quantum computing, the standard metric used to characterize adversarial perturbations is based on *quantum fidelity* [17].

Definition 1 (*Fidelity* [17]) Let $\rho, \sigma \in \mathcal{D}(\mathcal{H})$ be two quantum states. The fidelity between ρ and σ is defined as

$$F(\rho, \sigma) := \left(\mathrm{Tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right)^2.$$

Fidelity measures the closeness of two quantum states. It ranges from 0 to 1, where $F(\rho, \sigma) = 1$ if and only if $\rho = \sigma$, and $F(\rho, \sigma) = 0$ when ρ and σ have orthogonal support.

Definition 2 (*Fidelity Distance*) The *fidelity distance* (also called *infidelity*) between two quantum states is defined as

$$D_F(\rho, \sigma) := 1 - F(\rho, \sigma).$$

Fidelity distance provides a quantitative measure of how distinguishable two quantum states are under optimal measurements [17]. In robustness verification, it serves as a natural metric for evaluating the magnitude of adversarial perturbations: smaller distances imply higher similarity, while larger distances indicate more significant deviations from the original state.

10.2.3 Adversarial Robustness

To assess the adversarial robustness of a quantum classifier at a given input state ρ, we require that all quantum states sufficiently close to ρ be classified identically to ρ. A violation of this condition indicates the presence of an *adversarial example*—a nearby state that is misclassified.

Definition 3 (*Adversarial Example*) Let $\mathcal{A}$ be a quantum classifier, $\rho \in \mathcal{D}(\mathcal{H})$ an input state, and $\varepsilon > 0$ a perturbation threshold. A quantum state σ is called an ε-*adversarial example* of ρ if

$$\mathcal{A}(\sigma) \neq \mathcal{A}(\rho) \quad \text{and} \quad D_F(\rho, \sigma) \leq \varepsilon.$$

If such a state σ exists, then ε is referred to as an *adversarial perturbation* of ρ.

Here, the condition $\mathcal{A}(\sigma) \neq \mathcal{A}(\rho)$ indicates that σ is classified differently from ρ, while the condition $D_F(\rho, \sigma) \leq \varepsilon$ ensures that σ is sufficiently close to ρ in fidelity distance. Thus, an ε-adversarial example is a small perturbation of ρ that causes the classifier to mispredict.

Note that this definition implicitly assumes that ρ is correctly classified. If $\mathcal{A}(\rho)$ is already incorrect, identifying σ such that $\mathcal{A}(\sigma) \neq \mathcal{A}(\rho)$ no longer signifies a robustness issue but a correctness one. Therefore, in the subsequent discussions, we restrict attention to correctly classified input states.

The absence of adversarial examples within a given neighborhood defines robustness.

Definition 4 (*Adversarial Robustness*) A quantum classifier $\mathcal{A}$ is said to be ε-*robust* at state ρ if there exists no ε-adversarial example of ρ.

Equivalently, $\mathcal{A}$ is ε-robust at ρ if

$$\forall \sigma \in \mathcal{D}(\mathcal{H}), \quad D_F(\rho, \sigma) \leq \varepsilon \;\Rightarrow\; \mathcal{A}(\sigma) = \mathcal{A}(\rho).$$

This notion allows us to define the exact robustness radius of a quantum state.

Definition 5 (*Robustness Radius*) Let $\mathcal{A}$ be a quantum classifier and ρ a correctly classified input state. The *robustness radius* of ρ, denoted $\varepsilon^*(\rho)$, is the maximum value ε such that $\mathcal{A}$ is ε-robust at ρ:

$$\varepsilon^*(\rho) := \sup_{\substack{\sigma \in \mathcal{D}(\mathcal{H}) \\ \mathcal{A}(\sigma) = \mathcal{A}(\rho)}} D_F(\rho, \sigma).$$

Intuitively, $\varepsilon^*(\rho)$ quantifies the largest allowable fidelity degradation that preserves the classifier's decision. For any $\varepsilon > \varepsilon^*(\rho)$, there exists an ε-adversarial example σ that is misclassified relative to ρ.

The robustness radius gives rise to the central verification task addressed in this chapter:

Proposition 1 (Robustness Verification Problem) *Given a quantum classifier $\mathcal{A}$, an input state $\rho \in \mathcal{D}(\mathcal{H})$, and a threshold $\varepsilon > 0$, determine whether*

$$\varepsilon \leq \varepsilon^*(\rho).$$

If so, $\mathcal{A}$ is ε-robust at ρ; otherwise, ε is an adversarial perturbation, and a violating state σ can be returned as an ε-adversarial example.

The objective is to determine whether adversarial examples exist within an ε-fidelity ball centered at ρ. The remainder of this chapter develops exact and approximate approaches to solving this verification problem.

Finally, robustness can be aggregated across a dataset to evaluate a classifier's overall robustness:

Definition 6 (*Robust Accuracy*) Let $\mathcal{A}$ be a quantum classifier. The *ε-robust accuracy* of $\mathcal{A}$ is the proportion of correctly classified input states in the dataset that are also ε-robust.

This definition extends the notion of robustness from individual quantum states to an entire dataset. Depending on the application, the dataset may consist of training samples, validation samples, or a combination of both, under the assumption that all included quantum states are correctly labeled.

10.3 Robustness Bounds

This section presents a series of robustness bounds that address the robustness verification problem formulated in Problem 1. These bounds characterize the robustness radius $\varepsilon^*(\rho)$ through exact, under-approximate, and over-approximate techniques. We derive these bounds using a variety of methods, including semidefinite programming, analysis of measurement outcome distributions, and adversarial attack construction.

10.3.1 Optimal Robustness Bound Via Semidefinite Programming

We first show that the robustness radius $\varepsilon^*(\rho)$ can be exactly computed using *semidefinite programming (SDP)*, assuming full knowledge of the quantum classifier.

Given the explicit description of the quantum channel $\mathcal{E}$ and the POVM $\{M_c\}_{c\in C}$, the robustness radius can be formulated as an SDP [18]:

Theorem 1 (Optimal Robustness Bound via SDP [18]) *Let $\mathcal{A} = (\mathcal{E}, \{M_c\}_{c\in C})$ be a quantum classifier. The exact robustness radius is given by*

$$\varepsilon^*(\rho) = \min_{\substack{c\in C \\ c\neq \mathcal{A}(\rho)}} \varepsilon_c^*(\rho),$$

where each $\varepsilon_c^*(\rho)$ is the solution to the following SDP:

$$\begin{aligned} \textit{minimize:} \quad & D_F(\rho, \sigma) \\ \textit{subject to:} \quad & \sigma \succeq 0, \\ & \mathrm{Tr}(\sigma) = 1, \\ & \mathrm{Tr}[(M_{\mathcal{A}(\rho)} - M_c)\mathcal{E}(\sigma)] \leq 0. \end{aligned}$$

If this SDP is infeasible for some c, then $\varepsilon_c^*(\rho) = \infty$, indicating that no adversarial example of ρ exists which is misclassified as class c.

This SDP formulation allows exact and efficient computation of the robustness radius using convex optimization. The tractability of this approach is rooted in a fundamental principle of quantum mechanics: the linearity of quantum operations. Both the quantum channel $\mathcal{E}$ and the measurement process are linear maps, which leads to a convex feasible set and objective in the optimization problem.

This stands in contrast to classical neural networks, where nonlinearities such as ReLU activations or pooling layers make the robustness verification problem non-convex. As a result, certifying adversarial robustness in classical settings is NP-complete [21], and typically requires approximations or linear encodings of nonlinear functions (e.g., NSVerify [22], MIPVerify [23], ILP [24], ImageStar [25]).

However, computing $\varepsilon^*(\rho)$ using Theorem 1 assumes full access to the internal structure of the classifier, including its noise-free quantum evolution. In practice, especially on near-term noisy intermediate-scale quantum (NISQ) devices, this assumption rarely holds due to unknown or device-specific noise effects.

To address this, we introduce lower and upper bounds on $\varepsilon^*(\rho)$ that can be estimated under partial or empirical knowledge of the model. While the SDP method offers a gold-standard verification backend, these bounds provide practical alternatives for applications where exact model information is unavailable. In the following subsections, we present such bounds and discuss how they partially solve the robustness verification problem (Problem 1).

10.3.2 Robustness Lower Bound Via Measurement Distribution

We now present a certified lower bound for the robustness radius $\varepsilon^*(\rho)$ based on the measurement outcome distribution of a quantum classifier.

Let $\mathcal{A} = (\mathcal{E}, \{M_c\}_{c\in C})$ be a quantum classifier. For a given input state $\rho \in \mathcal{D}(\mathcal{H})$, denote the probability of observing class c as

$$p_c^\rho := \mathrm{Tr}[M_c \mathcal{E}(\rho)].$$

Let $c^* := \mathcal{A}(\rho)$ be the predicted class label of ρ. Then, the measurement distribution $\{p_c^\rho\}_{c\in C}$ enables the following certified lower bound:

Theorem 2 (Robustness Lower Bound from Measurement Distribution [18]) *Let $\rho \in \mathcal{D}(\mathcal{H})$ and $c^* = \mathcal{A}(\rho)$. Then*

$$\varepsilon_{\mathrm{RLB}}(\rho) := \min_{c\neq c^*} \frac{1}{2}\left(\sqrt{p_{c^*}^\rho} - \sqrt{p_c^\rho}\right)^2$$

is a certified robustness lower bound: for all σ such that $D_F(\rho,\sigma) \leq \varepsilon_{\mathrm{RLB}}(\rho)$, it holds that $\mathcal{A}(\sigma) = \mathcal{A}(\rho)$.

This robustness lower bound has the following key features:

- **Efficient to Compute**. The bound depends only on the measurement probabilities $\{p_c^\rho\}_{c\in C}$, and can be evaluated directly from measurement outcomes of ρ without searching for adversarial perturbations. This efficiency allows for fast robustness certification and dataset-level evaluation of robust accuracy.
- **Model-agnostic**: Since it requires no access to the internal structure of $\mathcal{E}$, this bound is particularly suited for hardware-level evaluation. In real-device settings, one can estimate p_c^ρ by repeated execution of $\mathcal{E}$ on quantum hardware and compute $\varepsilon_{\mathrm{RLB}}(\rho)$ from the empirical outcome distribution.

10.3.3 Robustness Upper Bound Via Attack Generation

In contrast to certified lower bounds, robustness upper bounds estimate the minimum perturbation required to fool a quantum classifier. These bounds can be obtained through adversarial attack strategies and provide empirical insights into the model's worst-case vulnerability.

Definition 7 (*Empirical Robustness Upper Bound*) Let $\rho \in \mathcal{D}(\mathcal{H})$ be an input quantum state. An *adversarial attack method* constructs a perturbed state σ_{adv} such that

$$\mathcal{A}(\sigma_{\text{adv}}) \neq \mathcal{A}(\rho), \quad \text{and} \quad \varepsilon_{\text{RUB}}(\rho) := D_F(\rho, \sigma_{\text{adv}}),$$

where D_F is the fidelity distance. Then, $\varepsilon_{\text{RUB}}(\rho)$ serves as an *empirical robustness upper bound* for $\varepsilon^*(\rho)$.

Since any adversarial example must lie outside the certified robust region, the inequality

$$\varepsilon^*(\rho) < \varepsilon_{\text{RUB}}(\rho)$$

always holds. Thus, ε_{RUB} provides an over-approximation of the robustness radius. Although ε_{RUB} is not a certified bound, it provides practical evidence of vulnerability and complements robustness lower bounds. It is especially valuable in real-device settings where exact robustness certification via SDP is infeasible.

In classical adversarial learning, numerous algorithms have been developed to craft adversarial examples [26]. These methods are typically categorized as either white-box or black-box attacks, depending on the attacker's access to model parameters. A widely used white-box approach is the *Fast Gradient Sign Method (FGSM)* [27], which perturbs a legitimate input $\boldsymbol{x}$ as follows:

$$\boldsymbol{x}' = \boldsymbol{x} + \varepsilon \cdot \text{sgn}(\nabla_{\boldsymbol{x}} \mathcal{L}), \tag{10.1}$$

where ε denotes the perturbation magnitude, $\nabla_{\boldsymbol{x}} \mathcal{L}$ is the gradient of the loss function $\mathcal{L}$ with respect to the input and the function $\text{sgn}(\cdot)$ stands for the sign function to extract the direction of a gradient without its magnitude.

In the quantum setting, generating analogous adversarial examples requires gradient estimation using the parameter-shift rule [28]. This introduces significant computational overhead on quantum hardware: producing a single adversarial example typically demands $2N_s \cdot N_x$ circuit executions, where N_s is the number of shots per measurement and $N_x = \dim(\boldsymbol{x})$ is the input dimension. Consequently, applying FGSM to high-dimensional quantum data becomes impractical, particularly in batch settings. This overhead highlights a major limitation of naive gradient-based attacks in quantum machine learning, especially on current noisy intermediate-scale quantum (NISQ) devices [29, 30].

To mitigate the overhead of full-gradient-based attacks, a localized variant of the FGSM, termed *Mask FGSM*, has been proposed for efficient adversarial sample generation in quantum machine learning (QML) experiments [20]. This method restricts perturbations to a strategically selected sparse subset of input features, specified by a binary mask $\mathcal{M} = (m_1, m_2, \ldots, m_{\dim(\boldsymbol{x})})^T$. The mask identifies which components of the input are perturbed, reducing both computational cost and the influence of noisy gradient estimates in experimental settings.

Given a legitimate input $\boldsymbol{x}$, adversarial examples are generated by applying a perturbation vector $\boldsymbol{\delta}$ such that $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\delta}$, where each component δ_i is computed as

$$\delta_i = \begin{cases} \varepsilon \cdot \operatorname{sgn}\left(\frac{\partial \mathcal{L}}{\partial x_i}\right), & \text{if } m_i = 1, \\ 0, & \text{if } m_i = 0, \end{cases} \tag{10.2}$$

with ε denoting the perturbation strength and $\mathcal{L}$ the loss function. The mask $\mathcal{M}$ is constructed by ranking gradient magnitudes, allowing targeted perturbation of the most influential input features (see [20] for implementation details).

This sparse attack strategy achieves a favorable trade-off between efficiency and effectiveness, as confirmed in the quantum hardware experiments on EMNIST and LCEI classification tasks [20], which will be detailed in Sect. 10.6.

10.3.4 Visualizing the Bounds

Based on the certified lower bound and the empirical upper bound presented in the previous subsections, we can formalize their relationship through the following result:

Theorem 3 (Sandwich Robustness Bound) *Given a quantum input state ρ, a certified lower bound $\varepsilon_{\mathrm{RLB}}(\rho)$ (Theorem 2), and an adversarially generated state σ_{adv}, we have*

$$\varepsilon_{\mathrm{RLB}}(\rho) \leq \varepsilon^*(\rho) \leq \varepsilon_{\mathrm{RUB}}(\rho), \tag{10.3}$$

where $\varepsilon_{\mathrm{RUB}}(\rho) = D_F(\rho, \sigma_{adv})$.

This relation is illustrated in Fig. 10.2. Each quantity in inequality (10.3) has a distinct role in robustness analysis:

- $\varepsilon_{\mathrm{RLB}}(\rho)$: a certified lower bound used for formal robustness guarantees;
- $\varepsilon^*(\rho)$: the exact robustness radius, computable via SDP;
- $\varepsilon_{\mathrm{RUB}}(\rho)$: an empirical upper bound derived from adversarial attacks.

Tightness Assessment. The gap $\Delta := \varepsilon_{\mathrm{RUB}}(\rho) - \varepsilon_{\mathrm{RLB}}(\rho)$ quantifies the precision of the robustness estimation. In the superconducting hardware experiments presented later [20], this certified lower bound is compared against an empirical robustness upper bound $\varepsilon_{\mathrm{RUB}}(\rho)$, obtained using gradient-based adversarial attacks (introduced in the next section). The observed gap between the two bounds is typically less than 3×10^{-3}, demonstrating that $\varepsilon_{\mathrm{RLB}}(\rho)$ provides a tight and practically useful certificate of robustness.

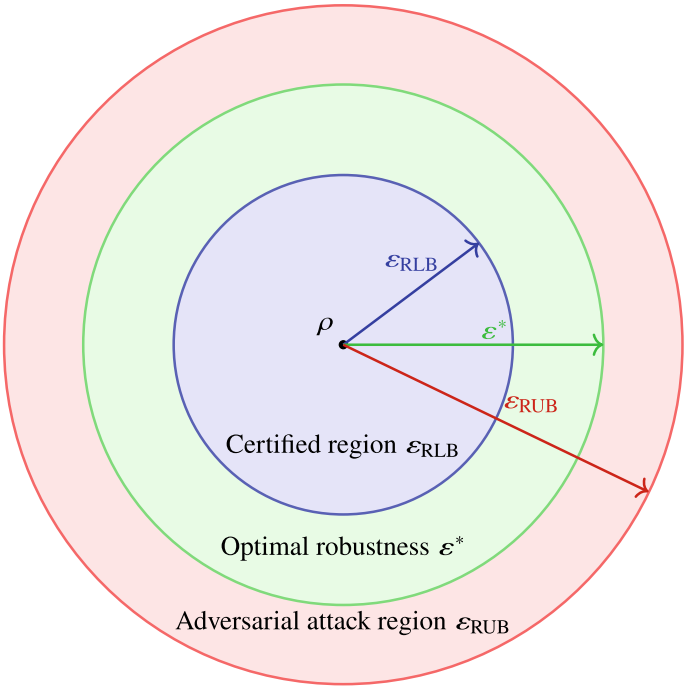

Fig. 10.2 Visualization of robustness bounds as nested fidelity-distance regions around the input state ρ. The certified lower bound $\varepsilon_{\mathrm{RLB}}$ defines a guaranteed-safe region, the upper bound $\varepsilon_{\mathrm{RUB}}$ corresponds to successful adversarial attacks, and the optimal robustness ε^* lies in between

10.4 Robustness Verification Algorithms

In this section, we present several algorithms for formally verifying the robustness of quantum classifiers, building upon the theoretical lower and optimal bounds introduced in the previous section.

10.4.1 State Robustness Verification

We begin by considering the robustness of a specific quantum input state ρ under a quantum classifier $\mathcal{A}$. The robustness verification problem (Problem 1) is to determine whether $\mathcal{A}$ is ε-robust at ρ for a given threshold $\varepsilon > 0$.

Note that once the exact robustness radius $\varepsilon^*(\rho)$ is computed, verifying ε-robustness reduces to a simple comparison: $\mathcal{A}$ is ε-robust at ρ if and only if $\varepsilon \leq \varepsilon^*(\rho)$. This observation, together with Theorem 1, leads to Algorithm 1, which simultaneously determines the robustness status of ρ and computes the exact robustness radius $\varepsilon^*(\rho)$.

Algorithm 1 StateRobustnessVerifier($\mathcal{A}$, ε, ρ)

Input: $\mathcal{A} = (\mathcal{E}, \{M_c\}_{c\in C})$ is a quantum classifier, $\varepsilon < 1$ is a real number, ρ is an input state of $\mathcal{A}$
Output: **true** indicates $\mathcal{A}$ is ε-robust at ρ or **false** with an adversarial example σ indicates $\mathcal{A}$ is not ε-robust at ρ
1: **for each** $c \in C$ and $c \neq \mathcal{A}(\rho)$ **do**
2: By a SDP solver, compute $\varepsilon_c^*(\rho)$ with an optimal state σ_c in the SDP of Theorem 1
3: **end for**
4: Let $\varepsilon^*(\rho) = \min_c \varepsilon_c^*(\rho)$ and $c^* = \arg\min_c \varepsilon^*(\rho)$
5: **if** $\varepsilon^*(\rho) > \varepsilon$ **then**
6: **return true**
7: **else**
8: **return false** and σ_{c^*}
9: **end if**

The primary computational cost of Algorithm 1 lies in solving semidefinite programs (SDPs), as shown in Line 2. These SDP instances scale as $O(N^{6.5})$ when solved using interior-point methods [31], where N is the dimension of the Hilbert space $\mathcal{H}$. Since the algorithm requires solving an SDP for each incorrect class label (i.e., $|C| - 1$ times), the overall computational complexity is given by the following:

Theorem 4 *The worst-case complexity of Algorithm 1 is $O(|C| \cdot N^{6.5})$, where N is the dimension of the input state ρ, and $|C|$ is the number of class labels.*

Algorithm 2 RobustnessVerifier($\mathcal{A}$, ε, T)

Input: $\mathcal{A} = (\mathcal{E}, \{M_c\}_{c\in C})$ is a well-trained quantum classifier, $\varepsilon < 1$ is a real number, $T = \{(\rho_i, l_i)\}$ is a dataset indicating ρ_i be correctly classified into class l_i.
Output: The robust accuracy RA and a set $R = \{< \sigma_j, i_j >\}$, where for each j, σ_j is an ε-adversarial example of ρ_{i_j}; R can be an empty set if $\mathcal{A}$ is ε-robust at any state in T.
1: $R = \emptyset$ be an empty set. *// Recording adversarial examples and corresponding indexes of states in dataset T*
2: **for each** $(\rho_i, l_i) \in T$ **do**
3: Let $p_c = \mathrm{Tr}[M_c\mathcal{E}(\rho_i)]$ be the measurement probability of observing outcome c for input ρ.
4: Obtain the lower bound $\varepsilon_{\mathrm{RLB}}(\rho_i) = \min_{c\neq c^*} \frac{1}{2}\left(\sqrt{p_{c^*}} - \sqrt{p_c}\right)^2$ *// Applying the robust bound in Theorem 2*
5: **if** $\varepsilon > \varepsilon_{\mathrm{RLB}}(\rho)$ **then**
6: **if** StateRobustnessVerifier ($\mathcal{A}$, ε, ρ_i) == **false then**
7: σ be the output state of StateRobustnessVerifier ($\mathcal{A}$, ε, ρ_i)
8: $R = R \cup \{(\sigma, i)\}$
9: **end if**
10: **end if**
11: **end for**
12: **return** $RA = 1 - \frac{|R|}{|T|}$, R *// $|R| = 0$ if R is a empty set*

10.4.2 Classifier Robustness Verification

We now turn our attention to verifying the robustness of a quantum classifier $\mathcal{A}$ as a whole. Algorithm 2 is developed for this purpose by combining the exact robustness computation procedure (Algorithm 1) with the certified lower bound technique from Theorem 2.

One key advantage of formal robustness verification in classical machine learning is its ability to identify counterexamples, i.e., adversarial examples, for specific inputs (see, e.g., [25, 32–34]). This advantage is preserved in the quantum setting through Algorithm 2, which enables both verification and counterexample generation. In particular, this facilitates a natural extension of adversarial training [35] to quantum machine learning. Specifically, when a quantum state ρ fails the ε-robustness check, the algorithm automatically identifies an adversarial example σ. By augmenting the training dataset with the misclassified pair (σ, l), where l is the correct label, the classifier $\mathcal{A}$ can be retrained to enhance its robustness. This iterative refinement process allows quantum models to gradually become more resilient to adversarial perturbations.

To analyze the complexity of Algorithm 2, we begin by noting from Theorem 1 that evaluating the robustness of a quantum classifier $\mathcal{A}$—specifically, computing its robust accuracy and identifying adversarial examples—requires invoking Algorithm 1 on each quantum state in the given dataset. Since the cost of a single call to Algorithm 1 is $O(|C| \cdot N^{6.5})$, the overall complexity for a dataset T of size $|T|$ is

$$O(|T| \cdot |C| \cdot N^{6.5}),$$

where $|C|$ is the number of classes and $N = \dim(\mathcal{H})$ is the dimension of the input state space.

However, the certified robustness lower bound provided in Theorem 2 offers a way to significantly reduce this computational cost. This bound can be computed efficiently in $O(|C| \cdot N^5)$ time, which corresponds to performing $|C| \cdot N^2$ matrix multiplications for $N \times N$ density matrices by noting that the number of Kraus operator in a super-operator $\mathcal{E}$ can be limited to be less than N^2.

A practical strategy for efficient robustness verification is thus as follows: we first use the certified lower bound to pre-screen the training dataset T, identifying all potentially non-robust states and collecting them in a subset $T' \subseteq T$. We then apply the exact robustness-checking Algorithm 1 only to the states in T', using a separate set R to record the discovered adversarial examples and their indices.

This two-stage approach reduces the overall complexity to

$$O(|T'| \cdot |C| \cdot N^{6.5}),$$

where typically $|T'| \ll |T|$, as empirically confirmed by our simulation results in [18]. This demonstrates the practical effectiveness of using the robust lower bound for accelerating formal verification of robustness in quantum classifiers.

10.4.3 Approximate Robustness Verification

Finally, we introduce an efficient algorithm (Algorithm 3) for under-approximating the robust accuracy computed by Algorithm 2. Specifically, Algorithm 3 serves as a lightweight subroutine that avoids calling an SDP solver whenever a potential non-robust state can be directly detected using the robustness lower bound established in Theorem 2.

Algorithm 3 UnderRobustAccuracy($\mathcal{A}$, ε, T)

Input: $\mathcal{A} = (\mathcal{E}, \{M_c\}_{c\in C})$ is a well-trained quantum classifier, $\varepsilon < 1$ is a real number, $T = \{(\rho_i, l_i)\}$ is a dataset indicating ρ_i be correctly classified into class l_i.

Output: A under-approximation of robust accuracy URA

1: $r = 0$. *// Record the number of potential non-robust states*
2: **for each** $(\rho_i, l_i) \in T$ **do**
3: Let $p_c = \mathrm{Tr}[M_c\mathcal{E}(\rho_i)]$ be the measurement probability of observing outcome c for input ρ.
4: Obtain the lower bound $\varepsilon_{\mathrm{RLB}}(\rho_i) = \min_{c\neq c^*} \frac{1}{2}\left(\sqrt{p_{c^*}} - \sqrt{p_c}\right)^2$ *// Applying the robust bound in Theorem 2*
5: **if** $\varepsilon > \varepsilon_{\mathrm{RLB}}(\rho_i)$ **then**
6: $r = r + 1$
7: **end if**
8: **end for**
9: **return** URA$= 1 - \frac{r}{|T|}$.

We have empirically compared Algorithms 2 and 3 in terms of verification time and output accuracy across various quantum classifiers simulated on classical hardware, as reported in [18]. The results demonstrate that Algorithm 3 exhibits favorable scalability and achieves significantly faster runtimes than Algorithm 2, while still providing accurate approximations of robust accuracy. These findings confirm the practical tightness of the proposed robustness lower bound.

10.4.4 Verification Algorithm Comparison

To conclude this section, we summarize the robustness verification algorithms based on different robustness bounds in Table 10.1. These algorithms vary in computational complexity and verification guarantees. In practice, the condition $|T'| \ll |T|$ often holds, confirming the utility of robustness lower bounds in efficiently identifying non-robust quantum states.

Thanks to the linearity of quantum learning models—an outcome of the foundational postulates of quantum mechanics—robustness verification for quantum classifiers can often be performed efficiently, with polynomial-time complexity in the dimension of the input state. This is in stark contrast to classical machine learn-

Table 10.1 Summary of robustness verification algorithms based on different bounds

Robustness verification algorithms			
	Robustness lower bound	Robustness optimal bound	Mixed strategy
Method	Matrix multiplication (MM)	Semidefinite programming (SDP)	MM & SDP
Complexity	$O(\|T\| \cdot \|C\| \cdot N^5)$	$O(\|T\| \cdot \|C\| \cdot N^{6.5})$	$O(\|T'\| \cdot \|C\| \cdot N^{6.5})$
Robust accuracy	Under-approximate	Exact	Exact

ing models, such as deep neural networks (DNNs), which are highly nonlinear and non-convex. In such classical settings, even verifying simple properties can be NP-complete [21].

However, this computational advantage in the quantum setting diminishes when verification is restricted to *pure states*. Unlike the convex set of mixed states, the set of pure states is non-convex. Consequently, the optimization problem underlying the computation of the robustness radius is no longer a semidefinite program (SDP) but becomes a *quadratically constrained quadratic program* (QCQP)—an optimization class where both the objective and constraints are quadratic [18]. Solving QCQPs is NP-hard in general.

To address this challenge, Algorithm 1 can be adapted to pure-state robustness verification by replacing the SDP solver in Line 2 with a QCQP solver. The resulting variant can be integrated into Algorithm 2 to compute robust accuracy and identify adversarial examples constrained to pure states. This adaptation has been evaluated in case studies such as MNIST classification, where input samples are encoded as pure quantum states [18].

10.5 VeriQR: A Tool for Robustness Verification

To translate the robustness verification algorithms (Algorithms 2 and 3) developed in the previous section into practical workflows, we introduce VeriQR [19], the first dedicated software tool for certifying adversarial robustness in quantum machine learning. VeriQR bridges formal verification, adversarial evaluation, and robustness enhancement, providing a flexible and efficient platform for real-world quantum applications.

VeriQR is available at https://github.com/Veri-Q/VeriQR.

As illustrated in Fig. 10.3, the core capabilities of VeriQR include

1. *Tool Integration*: A built-in parser compatible with OpenQASM, supporting frontends such as Qiskit, Cirq, and MindSpore Quantum;
2. *Noisy Simulation*: Injection of various noise models, including random noise, depolarizing, bit-flip, phase-flip, and user-defined noise;

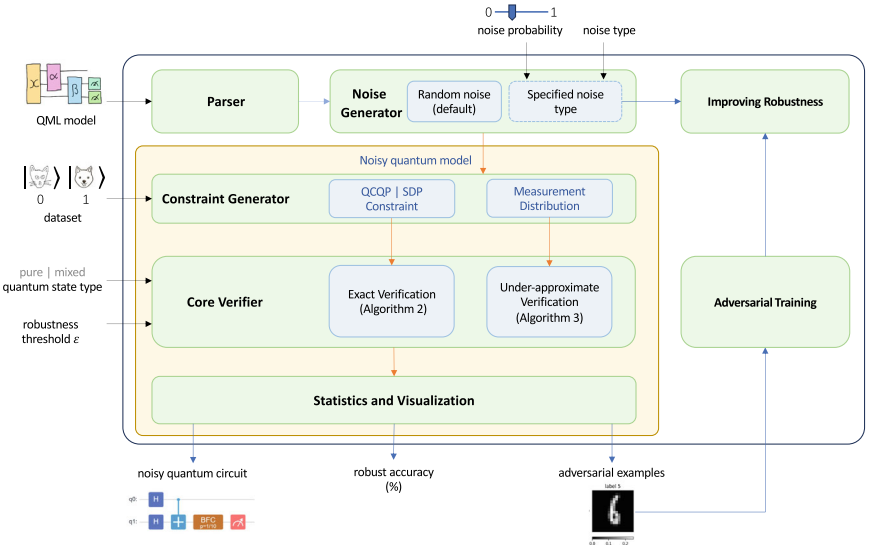

Fig. 10.3 System architecture of VERIQR

3. *Robustness Verification*: Formal verification of adversarial robustness for quantum classifiers based on certified bounds;
4. *GUI and Visualization*: Interactive interface for diagnostics and visualizing robustness landscapes of quantum learning models;
5. *Robustness Improvement*: Enhancement of classifier robustness via adversarial training using counterexamples identified through formal verification or adding specified quantum noise.

In the following, we detail these features.

10.5.1 Setups and Inputs

VERIQR is a graphical user interface (GUI) tool implemented in C++, leveraging the widely adopted Qt framework for GUI development [36].

As shown in the upper left corner of Fig. 10.3, users begin by importing a QML model along with a dataset of quantum states and corresponding ground truth labels, drawn from either the training or testing phase. VERIQR supports models in the following formats, each representing a quantum circuit with a final measurement stage:

1. *NumPy data file (.npz format)*: This format packages the quantum circuit, measurement operators, and dataset into a single file. It is particularly suitable for users familiar with classical machine learning and formal methods but less expe-

rienced in quantum computing. VERIQR includes several example .npz models, enabling new users to quickly begin robustness verification.
2. *OpenQASM 2.0 file (.qasm format)*: OpenQASM is a widely accepted standard for describing quantum circuits, originally introduced by IBM [37]. QML models trained using various quantum platforms can be converted into this format for unified robustness analysis within VERIQR.

Once a model is loaded, users can configure verification parameters via the control panel, as depicted in the left and top portions of Fig. 10.3. The configurable options include:

1. **Noise Model**. Select among several quantum noise types, including *depolarizing*, *phase-flip*, and *bit-flip* noise [38]. Random noise is enabled by default. Additionally, users may define custom noise models or apply combinations of different types.
2. **Quantum State Type**. Specify whether the input quantum states are *mixed* or *pure*. By default, VERIQR assumes mixed states, as they better represent realistic NISQ system states.
3. **Robustness Threshold**. Set the perturbation tolerance parameter ε for robustness verification. This determines the minimum fidelity distance required to guarantee robustness certification.

10.5.2 Verifying Robustness

As shown in the center of Fig. 10.3, the robustness verification workflow in VERIQR consists of five key modules:

(1) **Parser**. This module reads the quantum classifier file and constructs the corresponding quantum circuit object for downstream analysis.
(2) **Noise Generator**. Taking a quantum circuit as input, this module supports two noise injection modes:

   – In the default mode, random noise is applied to each qubit at random positions within the circuit, with randomly sampled noise probabilities, simulating the effects of real-world quantum hardware noise.
   – Alternatively, users can specify noise types (e.g., depolarizing, bit-flip, phase-flip, or custom Kraus operators) and inject them at the end of the circuit, a commonly adopted assumption. This functionality also enables robustness enhancement, as discussed in [39].

(3) **Constraint Generator**. This module generates the necessary mathematical constraints from the (possibly noisy) quantum circuit and dataset, preparing them for the verification engine.
(4) **Core Verifier**. Given the generated constraints, a perturbation threshold ε, and the type of quantum state (mixed or pure), this module selects an appropriate

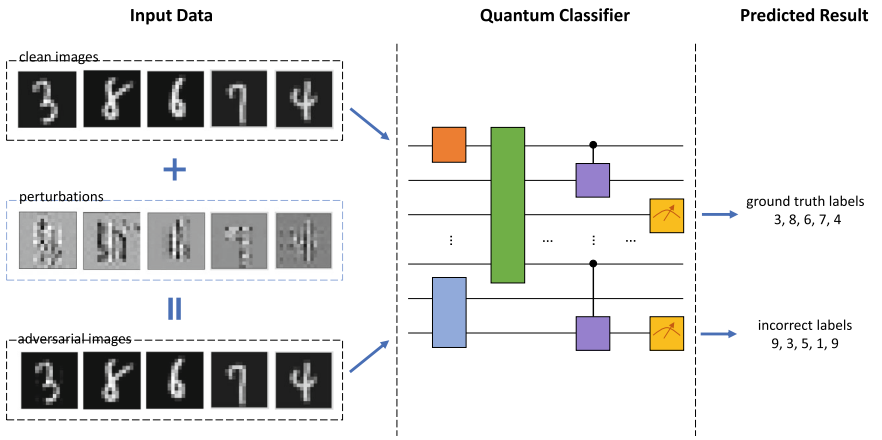

Fig. 10.4 The adversarial examples and the corresponding adversarial perturbations found by VERIQR in MINIST handwritten digit classification

solver: an SDP solver for mixed states or a QCQP solver for pure states [18]. It executes both exact and under-approximate verification procedures via Algorithms 2 and 3, respectively, to certify ε-robustness.

(5) **Statistics and Visualization**. This module visualizes verification results in the GUI. It reports the robust accuracy of the classifier and stores detected adversarial examples in a NumPy file for further analysis or adversarial training. Additionally, it displays the quantum circuit diagrams before and after noise injection. For MNIST digit classification tasks, VERIQR also presents images of adversarial examples corresponding to digits selected by the user, as illustrated in Fig. 10.4.

10.5.3 Improving Robustness

VERIQR supports two complementary strategies to enhance the adversarial robustness of QML models: adversarial training and the deliberate injection of specific quantum noise.

Adversarial Training. VERIQR extends classical adversarial training techniques to the quantum domain. When the ε-robustness of a state ρ with true label l fails, the embedded verification algorithms automatically generate an adversarial example σ. The pair (σ, l) can then be added to the training dataset, allowing the QML model to be retrained with adversarial samples. This iterative procedure effectively improves the model's robustness against future adversarial perturbations.

Injection of Specific Noise. Prior studies [40–42] have shown that carefully introducing quantum noise at selected points within a circuit can enhance robustness, potentially outperforming random noise. VERIQR provides users with the ability

to inject either standard or user-defined noise models—using Kraus operators—at specific positions in the quantum circuit. This functionality enables systematic exploration and optimization of noise-assisted robustness strategies for QML models.

10.6 Experimental Benchmark on Superconducting Hardware

To evaluate the practical hardware effectiveness of our robustness verification framework, the first experimental benchmark of adversarial robustness in QML was conducted on real superconducting quantum hardware [20]. This section outlines the experimental setup, summarizes the benchmark results, and highlights insights obtained from physical validation.

The formal framework was validated by executing QML classifiers on a 20-qubit superconducting processor [20]. As shown in Fig. 10.5a, the processor comprises 72 qubits and 126 couplers arranged in a 2D lattice; 20 qubits with high fidelity, highlighted in green, were selected for the experiments.

Two types of classification tasks were considered—classical and quantum—both implemented using trained quantum neural network (QNN) classifiers. As depicted in Fig. 10.5b, each QNN architecture consists of three components: a state preparation circuit that encodes data into a quantum input state, a variational circuit that performs the learning task, and a basis transformation followed by quantum measurement to extract output predictions.

1. **Classical Task**: The classification of handwritten characters "Q" and "T" was performed using the *EMNIST dataset* [43], as illustrated in Fig. 10.5c.
2. **Quantum Task**: The task of *Linear Cluster State Excitation Identification (LCEI)* [11, 18] involved distinguishing between excited and non-excited 20-qubit cluster states. These states were generated using a synthetic dataset, with excitation determined by the rotation angle α of an $R_x(\alpha)$ operation applied at the end of the cluster state preparation circuit (Fig. 10.5d).

For both tasks, classification was based on the expectation value of the output qubit with respect to the Pauli-Z operator, σ_z [17]. The resulting expectation value was converted into a probability via the transformation $p = (\langle\sigma_z\rangle + 1)/2$, with a threshold of $p = 0.5$ used to define the decision boundary.

For each classification task, adversarial robustness benchmarking is conducted on a well-trained binary quantum classifier using 10 randomly selected input quantum states (5 from each class). The experimental procedure involves the following steps:

1. **Measurement**: For each selected input quantum state ρ, execute the quantum classifier circuit on the quantum hardware to obtain the binary measurement outcome probabilities p_1, $p_2 = 1 - p_1$. Without loss of generality, it is assumed that $p_1 \geq p_2$ to facilitate subsequent analysis.

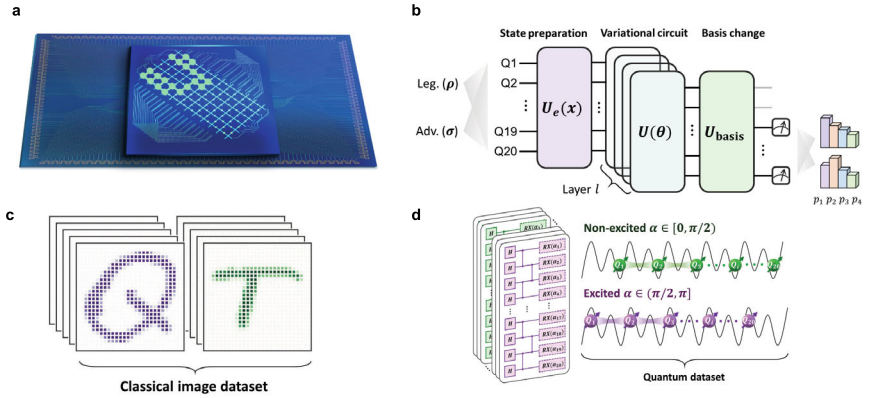

Fig. 10.5 Experimental schematic for QNN robustness evaluation. **a** Schematic diagram of the superconducting quantum processor, comprising 72 qubits and 126 couplers arranged in a 2D lattice. The 20 qubits selected for the experiment are highlighted in green. **b** Architecture of the quantum neural network (QNN) classifier, including the state preparation circuit, an l-layer variational circuit, and pre-measurement basis transformation gates. **c** Sample visualization of handwritten letters "Q" and "T" from the EMNIST dataset, used for the classical image classification task. **d** Quantum circuit used to generate the LCEI dataset, showing the application of an $R_x(\alpha)$ rotation following the linear cluster state. States are labeled as "excited" or "non-excited" based on the rotation angle α

2. **Robustness Lower Bound**: Compute the robustness lower bound $\varepsilon_{\text{RLB}}(\rho)$ based on the measured probabilities using Theorem 2.
3. **Robustness Upper Bound**: Apply the Mask FGSM attack (as described in Sect. 10.3.3) on the quantum hardware to estimate the robustness upper bound $\varepsilon_{\text{RUB}}(\rho)$ and identify the corresponding adversarial example σ_{adv}.
4. **Adversarial Training**: Retrain the quantum neural network classifier by incorporating the adversarial examples of all 10 quantum states into the training set, using their correct labels, in order to enhance the model's robustness.
5. **Adversarial Analysis**: Compare the robustness bounds before and after adversarial training to evaluate the tightness and stability of the certification framework.

As shown in Fig. 10.6, the experimental results validate two key aspects of the robustness verification framework:

1. **Tightness of Robustness Bounds**: Fig. 10.6a and c compare the experimentally measured robustness upper bounds ε_{RUB} (denoted as R_{UB}) against the certified robustness lower bounds ε_{RLB} (denoted as R_{LB}), computed from five randomly selected samples per class. In both EMNIST and LCEI tasks, the upper bounds consistently exceed the corresponding lower bounds, with average gaps of 2.96×10^{-3} and 3.25×10^{-3}, respectively. These results demonstrate the near-optimality of the Mask FGSM attack strategy and affirm the tightness of the certified lower bounds derived in Theorem 2.

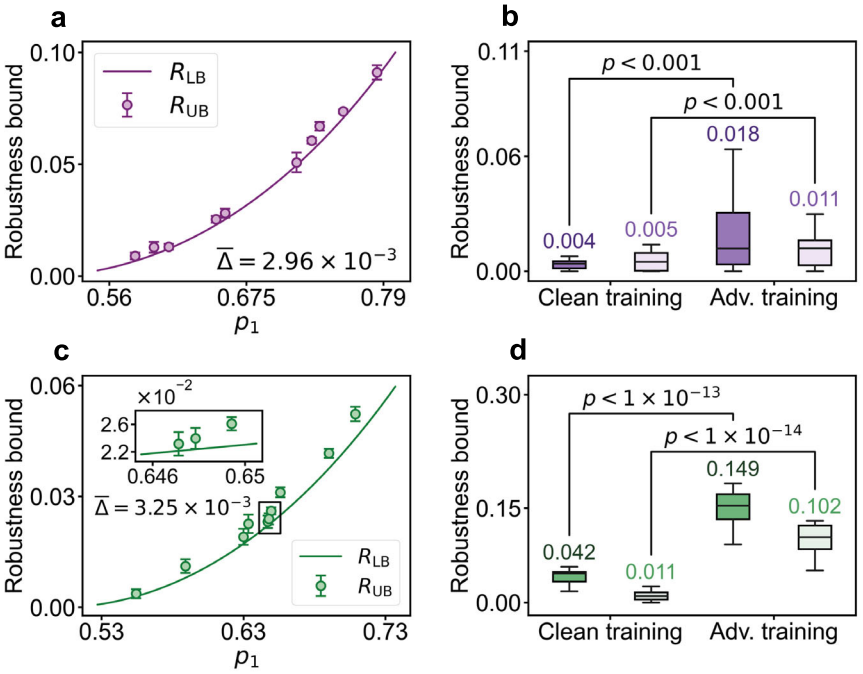

Fig. 10.6 Robustness bound verification experiments. **a**, **c**, Comparison of the experimental upper bound R_{UB} (from 10 randomly selected samples, 5 per class) versus theoretical R_{LB}. Error bars indicate the root mean square error from fitting $D(\rho, \sigma)$. $\overline{\Delta}$ denotes the average gap between R_{UB} and R_{LB} of the 10 samples. **b**, **d**, The robustness bounds for critical samples under clean and adversarial training. Adversarial training significantly increased the average robustness lower bound, with dark and light colors denoting distinct classes. p-value ≤ 0.001, indicating statistically significant differences. Panels **a** and **b** correspond to EMNIST dataset, while **c** and **d** to LCEI

2. **Improvement through Adversarial Training**: Critical samples, which are defined as the 20% of instances with the lowest robustness under clean training, were used to evaluate the effect of adversarial training. As depicted in Fig. 10.6b and d, adversarial training significantly increased the mean certified robustness lower bound for these critical samples, by a factor of 4.22 in EMNIST and 4.74 in LCEI. This improvement underscores the utility of adversarial training in enhancing the robustness of QMLs and mitigating misclassification under adversarial perturbations.

10.7 Summary and Outlook

This review has presented a comprehensive overview of our recently developed formal verification framework for certifying the adversarial robustness of quantum machine learning (QML) classifiers. The framework unifies rigorous theoretical foundations, algorithmic advances, toolchain development, and experimental validation into an end-to-end pipeline for trustworthy QML on quantum devices.

At the theoretical level, the framework introduces two certified robustness bounds for quantifying the adversarial robustness radius of a QML classifier at a given quantum input state:

- **Optimal Robustness Bound.** This bound exactly quantify the adversarial robustness radius and can be computable via semidefinite programming (SDP) when the input state is mixed, and via quadratically constrained quadratic programming (QCQP) for pure states.
- **Robustness Lower Bound.** This efficiently computable bound, derived from measurement statistics, certifies a region around a given input state where the classifier's output remains unchanged. It is especially valuable for identifying potentially non-robust inputs with low computational overhead and is well suited for hardware-level evaluation, as it does not require access to the internal evolution of the quantum classifier.

These certified bounds enable both exact and under-approximate verification. Furthermore, they are connected through the **robustness upper bound**, an empirical robustness estimate derived from adversarial attacks (e.g., the Mask FGSM method). This empirical bound serves as a benchmark for evaluating the tightness of the certified lower bound in practical scenarios.

To operationalize the theoretical results, the software tool VERIQR has been developed, serving as the first dedicated platform for robustness verification of QML models. VERIQR supports exact and approximate verification, simulates various noise models, enables adversarial training, and integrates with major quantum programming frameworks (e.g., Qiskit, Cirq). A graphical user interface (GUI) supports intuitive robustness diagnostics and visualization.

Notably, this framework has been experimentally validated on a 20-qubit superconducting quantum processor. These experiments constitute the first hardware-based benchmark of adversarial robustness in QML. The results show that:

- The certified robustness lower bounds closely match empirical upper bounds across classification tasks.
- Adversarial training substantially improves the robustness guarantees of quantum classifiers, particularly for inputs most vulnerable to perturbation.

With these results, our framework marks a promising step toward trustworthy QML on noisy intermediate-scale quantum (NISQ) hardware and opens promising directions for scalable and hardware-integrated robustness certification. In the following, we outline several key avenues for future research and development.

1. Hardware-integrated robustness verification. While the current implementation of VERIQR operates within classical simulation environments, a critical next step is to extend the framework for execution directly on quantum hardware. This would enable robustness certification under real-world conditions, accounting for hardware-specific noise characteristics, calibration drift, and temporal fluctuations. Such integration would bridge the practical gap between formal robustness guarantees and physical device behavior, enabling more faithful certification of QML models in practical deployment scenarios.

2. Certification of dynamically trained quantum classifiers. Most existing robustness verification approaches, including those presented in this review, assume a fixed and pre-trained quantum classifier. However, many real-world applications require models that learn from data in real time or adapt to adversarial perturbations. A promising direction is to develop verification methods that support dynamic or online QML training. In this setting, the quantum model evolves continuously, and robustness certification must be integrated into the training process to ensure that robustness guarantees are preserved or improved over time. This requires the design of adaptive certification techniques that can operate in tandem with quantum training loops, like their classical counterparts [44].

3. Beyond robustness: certifying fairness, interpretability, and privacy. While this framework focuses on adversarial robustness, many other aspects of trustworthiness are equally important in quantum AI systems. Future research may extend the verification methodology to properties such as fairness across quantum finance, interpretability of quantum decision-making, and differential privacy under quantum data access. Initial efforts in these directions have already begun [41, 45, 46], leveraging the robustness-based verification framework. A unified formal foundation for verifying such multifaceted trust properties would significantly strengthen the overall reliability of QML.

4. Cross-platform robustness benchmarking. The experimental results in this work are based on superconducting quantum hardware. Expanding the robustness benchmarking to other quantum platforms, such as trapped ions, neutral atoms, or photonic systems, would offer deeper insights into the relationship between hardware noise characteristics and model robustness. Establishing standard robustness benchmarks across different architectures will be essential for evaluating and comparing the trustworthiness of QML implementations on heterogeneous quantum systems.

These directions outline a broader agenda for building verifiable, trustworthy, and scalable quantum AI systems. As quantum computing advances toward practical and widespread use, formal verification tools like those presented in this framework may play an important role. By embedding trust guarantees into the lifecycle of quantum machine learning from model design and training to deployment and adaptation, future research can ensure that quantum AI not only performs well, but also behaves reliably and predictably in complex and uncertain environments. It is hoped that this review will foster broader engagement from the formal methods and quantum AI communities toward this critical goal.

References

1. Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S (2017) Quantum machine learning. Nature 549(7671):195–202
2. Cerezo M, Verdon G, Huang H-Y, Cincio L, Coles PJ (2022) Challenges and opportunities in quantum machine learning. Nat Comput Sci 2(9):567–576
3. Havlíček V, Córcoles AD, Temme K, Harrow AW, Kandala A, Chow JM, Gambetta JM (2019) Supervised learning with quantum-enhanced feature spaces. Nature 567:209–212
4. Herrmann J, Llima SM, Remm A, Zapletal P, McMahon NA, Scarato C, Swiadek F, Andersen CK, Hellings C, Krinner S, Lacroix N, Lazar S, Kerschbaum M, Zanuz DC, Norris GJ, Hartmann MJ, Wallraff A, Eichler C (2022) Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases. Nat Commun 13:12
5. Gong M, Huang HL, Wang S, Guo C, Li S, Wu Y, Zhu Q, Zhao Y, Guo S, Qian H, Ye Y, Zha C, Chen F, Ying C, Yu J, Fan D, Wu D, Su H, Deng H, Rong H, Zhang K, Cao S, Lin J, Xu Y, Sun L, Guo C, Li N, Liang F, Sakurai A, Nemoto K, Munro WJ, Huo YH, Lu CY, Peng CZ, Zhu X, Pan JW (2023) Quantum neuronal sensing of quantum many-body states on a 61-qubit programmable superconducting processor. Sci Bullet 68:906–912
6. Tacchino F, Macchiavello C, Gerace D, Bajoni D (2019) An artificial neuron implemented on an actual quantum processor. npj Quantum Inf 5:12
7. Huang HL, Du Y, Gong M, Zhao Y, Wu Y, Wang C, Li S, Liang F, Lin J, Xu Y, Yang R, Liu T, Hsieh MH, Deng H, Rong H, Peng CZ, Lu CY, Chen YA, Tao D, Zhu X, Pan JW (2021) Experimental quantum generative adversarial networks for image generation. Phys Rev Appl 16:8
8. Huang K, Wang Z-A, Song C, Xu K, Li H, Wang Z, Guo Q, Song Z, Liu Z-B, Zheng D, et al (2021) Quantum generative adversarial networks with multiple superconducting qubits. npj Quantum Inf 7(1):165
9. Zhang C, Lu Z, Zhao L, Xu S, Li W, Wang K, Chen J, Wu Y, Jin F, Zhu X, et al (2024) Quantum continual learning on a programmable superconducting processor. arXiv:2409.09729
10. Chen J, Wu Y, Yang Z, Xu S, Ye X, Li D, Wang K, Zhang C, Jin F, Zhu X et al (2025) Quantum ensemble learning with a programmable superconducting processor. arXiv:2503.11047
11. Broughton M, Verdon G, McCourt T, Martinez AJ, Yoo JH, Isakov SV, Massey P, Halavati R, Niu MY, Zlokapa A et al (2020) Tensorflow quantum: a software framework for quantum machine learning. arXiv:2003.02989
12. Lu S, Duan LM, Deng DL (2020) Quantum adversarial machine learning. Phys Rev Res 2:8
13. Liu N, Liu N, Liu N, Wittek P, Wittek P, Wittek P, Wittek P (2020) Vulnerability of quantum classification to adversarial perturbations. Phys Rev A 101:6
14. Franco N, Sakhnenko A, Stolpmann L, Thuerck D, Petsch F, Rüll A, Lorenz JM (2024) Predominant aspects on security for quantum machine learning: Literature review. In: 2024 IEEE international conference on quantum computing and engineering (QCE), vol 1. IEEE, pp 1467–1477
15. Huang L, Joseph AD, Nelson B, Rubinstein BIP, Tygar JD (2011) Adversarial machine learning. In: Proceedings of the 4th ACM workshop on security and artificial intelligence, pp 43–58
16. Goodfellow IJ, Shlens J, Szegedy C (2015) Explaining and harnessing adversarial examples. In: Bengio Y, LeCun Y (eds) 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015, conference track proceedings
17. Nielsen MA, Chuang IL (2010) Quantum computation and quantum information. Cambridge University Press
18. Guan J, Fang W, Ying M (2021) Robustness verification of quantum classifiers. In: International conference on computer aided verification. Springer, pp 151–174
19. Lin Y, Guan J, Fang W, Ying M, Su Z (2024) A robustness verification tool for quantum machine learning models. In: International symposium on formal methods. Springer, pp 403–421
20. Zhang H-F, Chen Z-Y, Wang P, Guo L-L, Wang T-L, Yang X-Y, Zhao R-Z, Zhao Z-A, Zhang S, Du L et al (2025) Experimental robustness benchmark of quantum neural network on a superconducting quantum processor. arXiv:2505.16714

21. Katz G, Barrett C, Dill DL, Julian K, Kochenderfer MJ (2017). Reluplex: an efficient smt solver for verifying deep neural networks. In: International conference on computer aided verification. Springer, pp 97–117
22. Lomuscio A, Maganti L (2017) An approach to reachability analysis for feed-forward relu neural networks. arXiv:1706.07351
23. Tjeng V, Xiao KY, Tedrake R (2019) Evaluating robustness of neural networks with mixed integer programming. In: 7th international conference on learning representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019. OpenReview.net
24. Bastani O, Ioannou Y, Lampropoulos L, Vytiniotis D, Nori AV, Criminisi A (2016) Measuring neural net robustness with constraints. In: Lee DD, Sugiyama M, von Luxburg U, Guyon I, Garnett R (eds) Advances in neural information processing systems 29: annual conference on neural information processing systems 2016, 5–10 Dec 2016, Barcelona, Spain, pp 2613–2621
25. Tran H-D, Bak S, Xiang W, Johnson TT (2020) Verification of deep convolutional neural networks using imagestars. In: International conference on computer aided verification. Springer, pp 507–526
26. Ren K, Zheng T, Qin Z, Liu X (2020) Adversarial attacks and defenses in deep learning. Engineering 6(3):346–360
27. Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. arXiv:1412.6572
28. Wierichs D, Izaac J, Wang C, Lin CY-Y (2022) General parameter-shift rules for quantum gradients. Quantum 6:677
29. Mitarai K, Negoro M, Kitagawa M, Fujii K (2018) Quantum circuit learning. Phys Rev A 98:9
30. Schuld M, Bergholm V, Gogolin C, Izaac J, Killoran N (2019) Evaluating analytic gradients on quantum hardware. Phys Rev A 99(3):032331
31. Zhang RY, Lavaei J (2018) Sparse semidefinite programs with near-linear time complexity. In: 2018 IEEE conference on decision and control (CDC). IEEE, pp 1624–1631
32. Elboher YY, Gottschlich J, Katz G (2020) An abstraction-based framework for neural network verification. In: International conference on computer aided verification. Springer, pp 43–65
33. Fremont DJ, Chiu J, Margineantu DD, Osipychev D, Seshia SA (2020) Formal analysis and redesign of a neural network-based aircraft taxiing system with verifai. In: International conference on computer aided verification. Springer, pp 122–134
34. Kwiatkowska MZ (2019) Safety verification for deep neural networks with provable guarantees (invited paper). In: Fokkink WJ, van Glabbeek R (eds) 30th international conference on concurrency theory, CONCUR 2019, 27–30 Aug 2019, Amsterdam, The Netherlands, vol 140. LIPIcs. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, pp 1:1–1:5
35. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2018) Towards deep learning models resistant to adversarial attacks. In: 6th international conference on learning representations, ICLR 2018, Vancouver, BC, Canada, April 30–May 3, 2018, conference track proceedings. OpenReview.net
36. Blanchette J, Summerfield M (2006) C++ GUI programming with Qt 4. Prentice Hall Professional
37. Cross AW, Bishop LS, Smolin JA, Gambetta JM (2017) Open quantum assembly language. arXiv:1707.03429
38. Nielsen MA, Chuang IL (2001) Quantum computation and quantum information. Phys Today 54(2):60
39. Lin Y, Guan J, Fang W, Ying M, Su Z (2024) Veriqr: a robustness verification tool for quantum machine learning models. arXiv:2407.13533
40. Yuxuan D, Hsieh M-H, Liu T, Tao D, Liu N (2021) Quantum noise protects quantum classifiers against adversaries. Phys Rev Res 3(2):023153
41. Guan J, Fang W, Ying M (2022) Verifying fairness in quantum machine learning. In: Computer aided verification: 34th international conference, CAV 2022, Haifa, Israel, Aug 7–10, 2022, proceedings, part II. Springer, pp 408–429
42. Huang J-C, Tsai Y-L, Yang C-HH, Su C-F, Yu C-M, Chen P-Y, Kuo S-Y (2023) Certified robustness of quantum classifiers against adversarial examples through quantum noise. In:

ICASSP 2023-2023 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 1–5
43. Cohen G, Afshar S, Tapson J, Van Schaik A (2017) Emnist: extending mnist to handwritten letters. In: 2017 international joint conference on neural networks (IJCNN). IEEE, pp 2921–2926
44. Huasong Meng M, Bai G, Teo SG, Hou Z, Xiao Y, Lin Y, Dong JS (2022) Adversarial robustness of deep neural networks: a survey from a formal verification perspective. IEEE Trans Dependable Secure Comput
45. Guan J, Fang W, Huang M, Ying M (2023) Detecting violations of differential privacy for quantum algorithms. In: Proceedings of the 2023 ACM SIGSAC conference on computer and communications security, pp 2277–2291
46. Guan J (2025) Optimal mechanisms for quantum local differential privacy. In: Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, pp 3737–3749