IOP Publishing New J. Phys. 25 (2023) 083019 https://doi.org/10.1088/1367-2630/ace8b4

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics

Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
18 April 2023

REVISED
4 July 2023

ACCEPTED FOR PUBLICATION
19 July 2023

PUBLISHED
10 August 2023

PAPER

Enhancing adversarial robustness of quantum neural networks by adding noise layers

Chenyi Huang[1,2] and **Shibin Zhang**[1,2,*]

[1] School of Cybersecurity, Chengdu University of Information Technology, Chengdu 610225, People's Republic of China
[2] Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu 610225, People's Republic of China
* Author to whom any correspondence should be addressed.

E-mail: cuitzsb@cuit.edu.cn

Keywords: quantum adversarial machine learning, quantum machine learning, adversarial examples, adversarial training

Abstract

The rapid advancements in machine learning and quantum computing have given rise to a new research frontier: quantum machine learning. Quantum models designed for tackling classification problems possess the potential to deliver speed enhancements and superior predictive accuracy compared to their classical counterparts. However, recent research has revealed that quantum neural networks (QNNs), akin to their classical deep neural network-based classifier counterparts, are vulnerable to adversarial attacks. In these attacks, meticulously designed perturbations added to clean input data can result in QNNs producing incorrect predictions with high confidence. To mitigate this issue, we suggest enhancing the adversarial robustness of quantum machine learning systems by incorporating noise layers into QNNs. This is accomplished by solving a Min-Max optimization problem to control the magnitude of the noise, thereby increasing the QNN's resilience against adversarial attacks. Extensive numerical experiments illustrate that our proposed method outperforms state-of-the-art defense techniques in terms of both clean and robust accuracy.

1. Introduction

The rapid advancement of machine learning, particularly in deep neural networks, has led to unprecedented progress in various fields, such as computer vision [1], natural language processing [2], and autonomous driving [3]. Simultaneously, quantum machine learning [4], a new computational paradigm that combines quantum computing with machine learning, is swiftly emerging. It leverages quantum parallelism and non-classical correlations, including quantum entanglement, to potentially accelerate or revolutionize existing classical algorithms [5, 6]. Importantly, the fusion of these disciplines can lead to symbiotic advancements and offer fresh perspectives for tackling a wide range of challenging problems. For example, classical machine learning has been effectively utilized to address quantum many-body problems in the fields of physics and chemistry [7], along with propelling the progress of quantum simulation technology [8]. Simultaneously, the amalgamation of physics concepts and traditional machine learning techniques has exhibited considerable potential in solving practical quantum problems [9].

With the development of quantum machine learning on noisy intermediate-scale quantum devices [10], quantum neural networks (QNNs) [11–15] have surfaced as a promising approach to executing machine learning tasks on these devices. By fusing quantum computing with deep neural network models, this approach permits the embedding of input data into Hilbert spaces and the execution of classical machine learning tasks such as classification [11, 15, 16], generative modeling [13, 17, 18], among others [19, 20]. QNNs consist of parameterized gate operations in quantum circuits, with classical optimization methods like gradient descent employed for optimizing the quantum gate parameters (e.g. Pauli rotation angles) for particular tasks and determining the optimal parameters [19, 20]. Notably, these networks show potential

quantum computational advantages when processing certain quantum synthesized data and solving discrete logarithm problems [21]. However, similar to their classical counterparts, quantum machine learning systems also exhibit a lack of robustness against adversarial attacks [22–25]. More specifically, for QNNs addressing classification problems, adversaries can generate small perturbations that result in erroneous and high-confidence classification outputs through adversarial attack algorithms. These manipulated inputs, known as adversarial examples [26], pose a significant challenge for deploying QNNs in security-sensitive applications such as quantum state recognition [24], quantum topological phase recognition [22], and medical diagnosis [24]. This security threat has also been experimentally demonstrated on superconducting quantum devices [24].

Various methods have been proposed in classical machine learning to improve robustness against adversarial attacks, including adversarial training [27, 28] and model regularization with randomized noise [29–31]. Motivated by these works, recent research has investigated different approaches to enhance the adversarial robustness of quantum classifiers. Lu *et al* [22] proposed quantum adversarial training to enhance the robustness of quantum classifiers, but adversarial training inevitably results in a decrease in accuracy on clean datasets [32]. Du *et al* [33] investigated the use of depolarizing noise in quantum circuits to enhance the adversarial robustness of quantum models. However, the model's accuracy declines as noise increases beyond a threshold, and in practice, noise cannot render quantum models entirely robust [33].

To alleviate these issues, we introduce an innovative framework incorporating the concept of 'noise layer'. The noise layers randomly rotate the quantum bits on the Bloch sphere and optimize the controlled random rotation angles through end-to-end training. Our framework generates adversarial examples within QNNs using noise layers, aiming to maximize the inner minimax problem in adversarial training. Furthermore, to tackle the outer minimization problem, we suggest employing the combined loss to guide the QNN's learning to balance accuracy on both clean and adversarial data. Our method does not rely on any assumptions about the structure of QNN, and can be easily implemented on QNN with minimal additional parameters and computational overhead. In comparison to quantum adversarial training [22], our strategy achieves better accuracy on clean data while maintaining competitive robust accuracy on adversarial data.

2. Related work

2.1. QNN classifier

Classification tasks in supervised learning represent an important branch of quantum machine learning. These tasks involve a decision process in which a model is trained on labeled data, enabling it to predict the corresponding labels for a set of input data. In the context of quantum classification, the objective is to learn an algorithm $\mathcal{A}:\rho\to y$ that maps input quantum data $\rho\in\mathcal{H}$, where $\mathcal{H}$ is a subspace of the Hilbert space, to a label $y\in\mathcal{Y}$ in $\mathcal{Y}$, with $\mathcal{Y}$ being a finite, countable set of labels. For simplicity, we assume the input quantum states are pure states. To learn $\mathcal{A}$, a quantum classifier must be trained on a given training set T. The training set $T=(\rho_i,y_i)_{i=1}^M$ consists of $M<\infty$ pairs (ρ_i,y_i), where ρ_i and y_i are the input state and the corresponding label, respectively. Given a parameterized model $f(\theta,\rho)$, where θ is a set of adjustable parameters, learning is performed by optimizing θ to minimize the empirical risk,

$$\min_{\theta}\frac{1}{M}\sum_{i=1}^{M}\mathcal{L}\left(f(\theta,\rho_i),y_i\right), \tag{1}$$

where $\mathcal{L}$ is a pre-defined loss function.

A multitude of QNN models have been developed for classification tasks, drawing inspiration from classical deep learning models [11, 12, 15, 16]. These QNNs consist of a series of quantum circuits that comprise parameterized gates [19]. For classical data, the data needed for classification is first encoded into quantum states through specific state preparation procedures or feature mapping processes [34]. Common methods primarily include amplitude encoding [22, 35] and angle encoding [14, 36]. Amplitude encoding is a method that normalizes the input data and associates it with the probability amplitude of the quantum state. This method can map classical vectors to the Hilbert space of dimension $d=2^n$, where n is the total number of quantum bits used for characterization. On the other hand, angle encoding is more intuitive, where each feature corresponds to a quantum bit, thus requiring n quantum bits to represent n features. Once the data is encoded into the quantum state ρ, a series of parameterized gates will be applied and optimized for the classification task. For quantum data, the data can be directly fed into the QNN circuit.

Various structures of QNNs have been proposed, including quantum convolutional neural networks (QCNNs) [12, 37], hierarchical quantum classifiers [16], tensor networks [38], and others. Finally, in order

to extract information from ρ for prediction, it is necessary to measure the expectation values of selected observables. In the case of binary classification problems, a natural approach is to measure a single qubit, producing a binary outcome of 0 or 1 as the predicted label $\hat{y}$. However, since the measurement is a probabilistic process, the exact value can only be obtained through infinite sampling, and the classifier can only output an estimate of the probability $\hat{y}(\rho)$. It is noteworthy that while this paper focuses solely on binary classification, the same approach can be applied to multi-class problems.

2.2. Quantum adversarial attack

Recent research has revealed that techniques from classical adversarial learning can produce subtle perturbations in input data, deceiving highly accurate quantum classifiers [22–24]. Adversarial attacks can be categorized as either white-box or black-box attacks, depending on the adversary's access to the quantum model's information.

White-box attacks [26, 27] involve the adversary having complete knowledge of the quantum model's architecture and parameters, while black-box attacks [22, 23] occur when the adversary lacks any knowledge of the quantum model.

The fast gradient sign method (FGSM) [22, 27] is an effective one-step adversarial attack method where the adversarial example is generated by:

$$\rho^{\mathrm{adv}} = \rho + \epsilon \cdot \mathrm{sgn}\left(\nabla_{\rho}\mathcal{L}\left(f(\theta,\rho),y\right)\right), \tag{2}$$

where ϵ is the perturbation constraint determining the attack strength, $\mathrm{sgn}(\cdot)$ is the sign function, and ∇_{ρ} denotes the gradient of the loss with respect to the legitimate ρ with the correct label y. Various attack methods have been developed to improve attack efficiency and usability, such as the basic iterative method (BIM) [22, 23, 33, 39], a multi-step variant of FGSM, to generate adversarial examples:

$$\begin{aligned} &\rho_0^{\mathrm{adv}} = \rho, \\ &\rho_{k+1}^{\mathrm{adv}} = \Pi\left[\rho_{\mathrm{k}}^{\mathrm{adv}} + \alpha \cdot \mathrm{sgn}\left(\nabla_{\rho}\mathcal{L}\left(f\left(\theta,\rho_{\mathrm{k}}^{\mathrm{adv}}\right),y\right)\right)\right], \end{aligned} \tag{3}$$

where $\rho_{\mathrm{k}}^{\mathrm{adv}}$ denotes the adversarial example updated at step k, Π is the clipping function, which can be viewed as the projection operator of the normalized wave function in the field of quantum machine learning [22], and α is the attack step size.

One prevalent black-box attack is executed through substitute models [40], where the adversary trains a model using the target model's outputs as labels and employs the trained model to generate adversarial examples. Transferable adversarial attack [41], a variant of the substitute model attack, are a crucial technique for attacking quantum models. In transferable adversarial attacks, the adversary generates adversarial examples from a source model, which could be either a classical [22] or a quantum [23] model, to attack the target model. The source and target models can have completely different structures, but they need to be trained using real training data.

2.3. Quantum adversarial defense

Due to the security threats posed by adversarial examples [26], several defense strategies have been proposed to enhance the adversarial robustness of quantum machine learning models, such as adversarial training [22, 24] and the use of depolarizing noise [33].

Adversarial training involves treating adversarial examples as training data, which helps the model learn to differentiate between adversarial examples and improve its adversarial robustness. The standard adversarial training approach aims to obtain the optimal solution for QNN parameters θ by solving the following min-max problem:

$$\arg\min_{\theta}\left\{\arg\max_{\rho^{\mathrm{adv}}\in\mathcal{H}_{\epsilon}}\mathcal{L}\left(f\left(\theta,\rho^{\mathrm{adv}}\right),y\right)\right\}, \tag{4}$$

where the inner maximization finds perturbed data ρ^{adv}, and $\mathcal{H}_{\epsilon}$ is the set of adversarial perturbations subject to the ϵ-constraint. The outer minimization is optimized through standard QNN training. This simple and effective method has been experimentally verified on superconducting computers [24].

In classical adversarial learning, injecting noise into the model to enhance its adversarial robustness has yielded promising results. Inspired by this, Du *et al* [33] proposed using depolarizing noise in quantum circuits to improve robustness, leveraging the connection between differential privacy and adversarial robustness. Injecting noise can also be viewed as a form of model regularization.

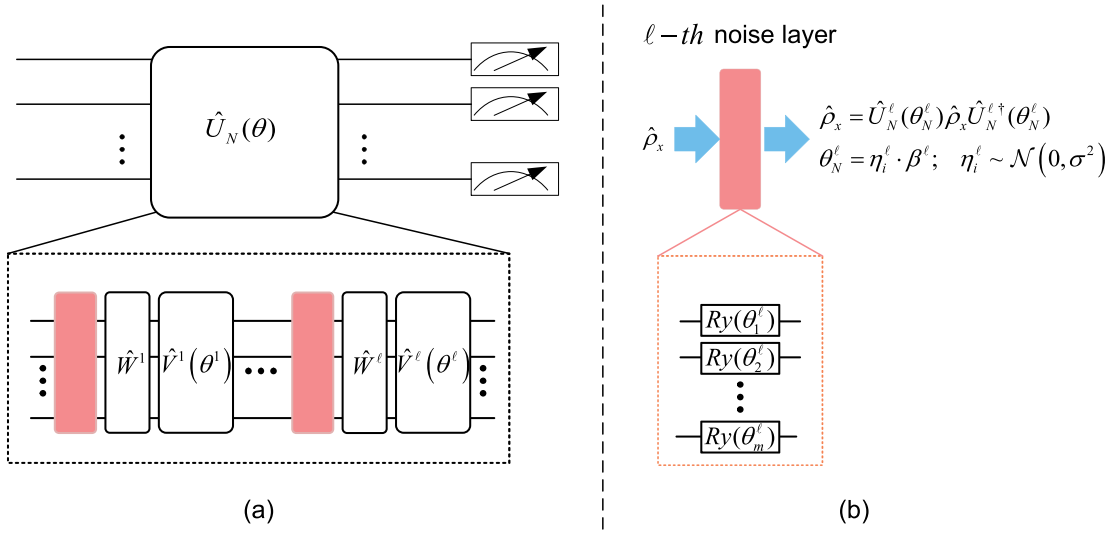


Figure 1. (a) Our proposed QNN structure with noise layers. (b) The specific structure of the noise layer, using the RY gate as an example.

3. QNNs with noise layers

In this section, we propose improving the adversarial robustness of QNNs by incorporating noise layers. We will first introduce the proposed method and then discuss the impact of this method on QNNs.

3.1. Proposed method

Our main idea is to introduce randomness into the variational circuits to achieve model regularization. We introduce noise layers that can add controllable perturbations to the quantum bits in the corresponding quantum circuits. Specifically, QNNs are generally considered as quantum simulations of classical neural networks, consisting of multiple layers of parameterized quantum circuits. Each layer includes trainable parameters for rotation gates and entangling gates, which can be written as a product of unitary gates in the following form:

$$\hat{U}(\theta) = \hat{U}(\theta_1, \ldots \theta_P) = \prod_{\ell=1}^{P} \hat{W}^\ell \hat{V}^\ell\left(\theta^\ell\right), \tag{5}$$

where P denotes the number of layers in the variational circuit, $\hat{W}^\ell$ represents non-parametric unitary operations at the ℓth layer, $\hat{V}^\ell\left(\theta^\ell\right)$ represents unitary operations with variational parameters at the ℓth layer, and θ^ℓ denotes all parameters in the ℓth layer.

To increase the randomness of QNNs, we propose adding noise layers before each QNN layer. As shown in figure 1(a), we represent the structure of QNNs with noise as $\hat{U}_N(\theta)$ and the quantum state after passing through the ℓth layer of the noise layer can be expressed as $\hat{U}_N^\ell(\theta_N^\ell)\hat{\rho}\hat{U}_N^\ell\dagger(\theta_N^\ell)$. Then, the QNN with noise layers can be written as follows:

$$\hat{U}_N(\theta) = \hat{U}_N(\theta_1, \ldots \theta_P) = \prod_{\ell=1}^{P} \hat{U}_N^\ell(\theta_N^\ell)\hat{W}^\ell \hat{V}^\ell\left(\theta^\ell\right). \tag{6}$$

The role of the noise layer is to induce random rotations on the Bloch sphere. This is achieved by incorporating single-qubit rotation gates, such as the RY gate shown in figure 1(b), to introduce random rotations and generate random variables as rotation angles. The coefficients used to control the size of each η_i^ℓ can be mathematically described as follows:

$$\theta_N^\ell = \eta_i^\ell \cdot \beta^\ell; \quad \eta_i^\ell \sim \mathcal{N}\left(0, \sigma^2\right), \tag{7}$$

where η_i^ℓ is the noise variable of the ith qubit in the ℓth layer. For simplicity, we assume that η is randomly sampled from a Gaussian distribution with zero mean and variance σ^2. However, our method can also be applied to other noise distributions, such as Gaussian–Bernoulli. β^ℓ is the coefficient used to control the size of each η_i^ℓ. Assuming that a noise layer is added throughout the entire QNN, only one β^ℓ per layer needs to

be optimized for each parameterized quantum circuit in that layer. It can be regarded as a variational parameter in the QNN and optimized through gradient descent.

Like most classifiers, the QNN with noise layers consists of two independent processes, training and inference. The algorithms for training and inference are presented in algorithms 1 and 2, respectively.

Algorithm 1. Training of quantum neural network with noise layer.

Input: The model with parameter θ, the constraint coefficient β, the combined loss function $\mathcal{L}'$, the training set $T = \{(\rho_i, y_i)\}_{i=1}^{M}$, the learning rate r, the batch size n_b, and the optimizer f_o
Output: The trained model

Initialization: generate random initial parameters for θ
for *number of training iterations* **do**
  Randomly choose n_b samples $(\rho_1, y_1), \ldots, (\rho_{n_b}, y_{n_b})$ in T
  Random variables are generated by (7)
  Set up rotation angles of additional single-quantum rotation doors with random variables
  Compute gradients (noise gradients) $G \leftarrow \frac{1}{n_b}\sum_{k=1}^{n_b} \nabla\mathcal{L}'\left(h\left(\rho_{(\mathrm{k})}; \boldsymbol{\theta}, \boldsymbol{\beta}\right), \mathbf{y}_{(\mathrm{k})}\right)$
  Updates: $\theta, \beta \leftarrow f_o(\theta, \beta, \mathrm{r}, \mathrm{G})$
end
Output the trained model

Algorithm 2. Testing of quantum neural network with noise layer.

Input: The trained model with parameter θ, the constraint coefficient β, the given testing sample ρ, the number of measurements $\mathcal{J}$, the counter $count = 0$
Output: Predicted class label $\hat{y}$

Initialization: $count \leftarrow 0$;
for $i = 1$ **to** $\mathcal{J}$ **do**
  Random variables are generated by (7)
  Set up rotation angles of additional single-quantum rotation doors with random variables
  Measure the output qubits and obtain the measurement result y_i
  if $y_i == 1$ **then**
    $count \leftarrow count + 1$;
  end
end
Calculate the frequency of the predicted class label: $f_{\hat{y}[\rho]} \leftarrow count/\mathcal{J}$
if $f_{\hat{y}[\rho]} > 0.5$ **then**
  Predicted class label $\hat{y} = 1$;
end
else
  Predicted class label $\hat{y} = 0$;
end
Output the predicted class label

However, when training QNNs with noisy layers, we found that the optimizer tends to minimize the noise, leading to β convergence to values close to zero (as shown in table 2). To address this, we combine noise injection with adversarial training to avoid overfitting to clean data and successfully defend against adversarial attacks. As previously mentioned, adversarial training aims to solve the min-max problem in (4). In QNNs with noisy layers, solving the internal maximization problem aims to find adversarial examples for a given data point, which is obtained by solving the following equation:

$$\rho^{\mathrm{adv}*} = \arg\max_{\delta\in\Delta} \mathcal{L}\left(f_\epsilon\left(\theta, \rho+\delta\right), y\right), \tag{8}$$

where Δ denotes a sufficiently small region to ensure that the adversarial perturbation is small and does not fundamentally alter the original input, and $f_\epsilon(\theta, \rho)$ represents the model with noisy layers. Unlike standard adversarial example generation, our $\rho^{\mathrm{adv}*}$ is generated with noise injection. Moreover, when solving the minimization problem, in order to avoid the model only minimizing the loss for perturbed data during the outer minimization process and losing information about clean data, we do not minimize only the adversarial loss given by the adversarial example in (4), but instead use the combined loss $\mathcal{L}'$, which is a weighted sum of the losses on clean and perturbed data. The combined loss $\mathcal{L}'$ is described as follows:

$$\mathcal{L}' = \gamma \times \mathcal{L}\left(f_\epsilon\left(\theta, \rho^{\mathrm{adv}*}\right), y\right) + (1-\gamma) \times \mathcal{L}\left(f_\epsilon\left(\theta, \rho\right), y\right), \tag{9}$$

where γ is a hyperparameter that controls the weight of the loss for clean data and perturbed data. In this paper, we set it to 0.5. The advantage of this approach is that, compared to the loss function of (4), the model in adversarial training no longer focuses solely on the 'current' training sample (whether it is a clean or adversarial sample). With the loss function defined by (9), when training on clean samples, the model takes into account the impact of adversarial examples; conversely, when training on adversarial examples, the model also considers the impact of clean samples.

3.2. Analysis

First, during the training process of QNNs with noisy layers, noise is randomly generated at each gradient descent step. Empirical evidence [42, 43] suggests that this regularization-like technique assists the optimization algorithm in finding parameters that are robust to adversarial perturbations. Liu *et al* [42] theoretically demonstrated that adding random noise is equivalent to Lipschitz regularization. By controlling the noise coefficient, one can balance the model's robustness with the training loss. Moreover, similar to classical adversarial machine learning, quantum adversarial attacks require access to the model's gradient information [22, 24]. Due to the peculiarities of variational circuits, finite-difference methods and the parameter-shift rule [14, 19, 44, 45] are commonly used to estimate the gradients of QNNs. The parameter-shift rule is particularly suitable for implementation on current quantum devices. Under normal circumstances, the required partial derivatives can be numerically calculated using finite-difference schemes, as shown below:

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \frac{\mathcal{L}(\theta+\Delta)-\mathcal{L}(\theta-\Delta)}{2\Delta}, \tag{10}$$

where Δ is a small random perturbation vector. To estimate the gradient, this method requires evaluating the loss function twice for each parameter. Due to the presence of noise layers, adversaries are affected by the randomly generated noise perturbations when computing the two evaluations of the loss function, resulting in a significant discrepancy between the computed gradient and the true gradient. Although the parameter-shift rule can accurately obtain the gradients of circuits, this method also requires the execution of circuits twice to estimate the gradient. Similarly, adversaries cannot accurately obtain the gradient of circuits with noise layers. Since most attacks require the computation or estimation of gradients, noise layers help to perturb the gradients and deceive adversarial attacks, thereby reducing the success rate of adversarial attacks.

Second, conventional adversarial training [22, 28] aims to associate the same class label with a sample and its surrounding adversarial examples to help the model achieve a certain level of certified robustness. However, this approach can lead to severe overfitting on the training data, resulting in a decrease in the model's generalization ability on natural samples. Additionally, the noise injection in QNNs increases the model's randomness, inevitably leading to a decrease in accuracy. To address these issues, we introduce the combined loss function $\mathcal{L}'$ to successfully train QNNs. We optimize $\mathcal{L}'$ to find an appropriate value of β that balances the model's prediction accuracy with its robustness against adversarial attacks. This helps avoid excessive or insufficient noise, which can both affect the model's accuracy and increase the risk of overfitting to the training data.

Finally, to better understand the potential sample space generated by QNNs with embedded noise layers, we randomly selected 100 samples of the digits 1 and 9 from the MNIST dataset and performed unsupervised PCA feature reduction to reduce the size of the space to 2, demonstrating the effect of adversarial perturbations on QNNs with noise layers. We represent data that the model can correctly classify with circles and data that it cannot correctly classify with crosses. In figure 2(a), almost all data can be correctly classified by the QNN trained with standard methods. In figure 2(b), we use $\epsilon = 0.15$ to generate adversarial examples corresponding to randomly selected samples, and approximately 65% of the samples cannot be correctly classified. In figure 2(c), we perform quantum adversarial training with $\epsilon = 0.15$ and then generate corresponding adversarial examples. Most of these samples can be correctly classified by the model trained with adversarial training, but the performance of the model in predicting clean samples will be degraded. This is because, in order to help the model learn more complex decision boundaries, adversarial training assigns the same class labels to adversarial examples as to clean training samples, which causes the model to overfit to the training data and destroys the classification performance of clean samples. In figure 2(d), we also use $\epsilon = 0.15$ to perform adversarial training in the QNN with noise layers, and compared with standard adversarial training, the model shows better performance on both clean and adversarial data. Compared with figure 2(c), due to the interference of the noise layer, the 'distance' between the adversarial examples generated for the model and the original samples is closer. Specifically, the noise layer adds additional complexity to the generation of adversarial examples, making it more difficult for adversarial attacks to significantly deviate from the original samples. This is an ideal result because it ensures that the model does not overfit to adversarial examples and produce distorted and adverse decision boundaries.

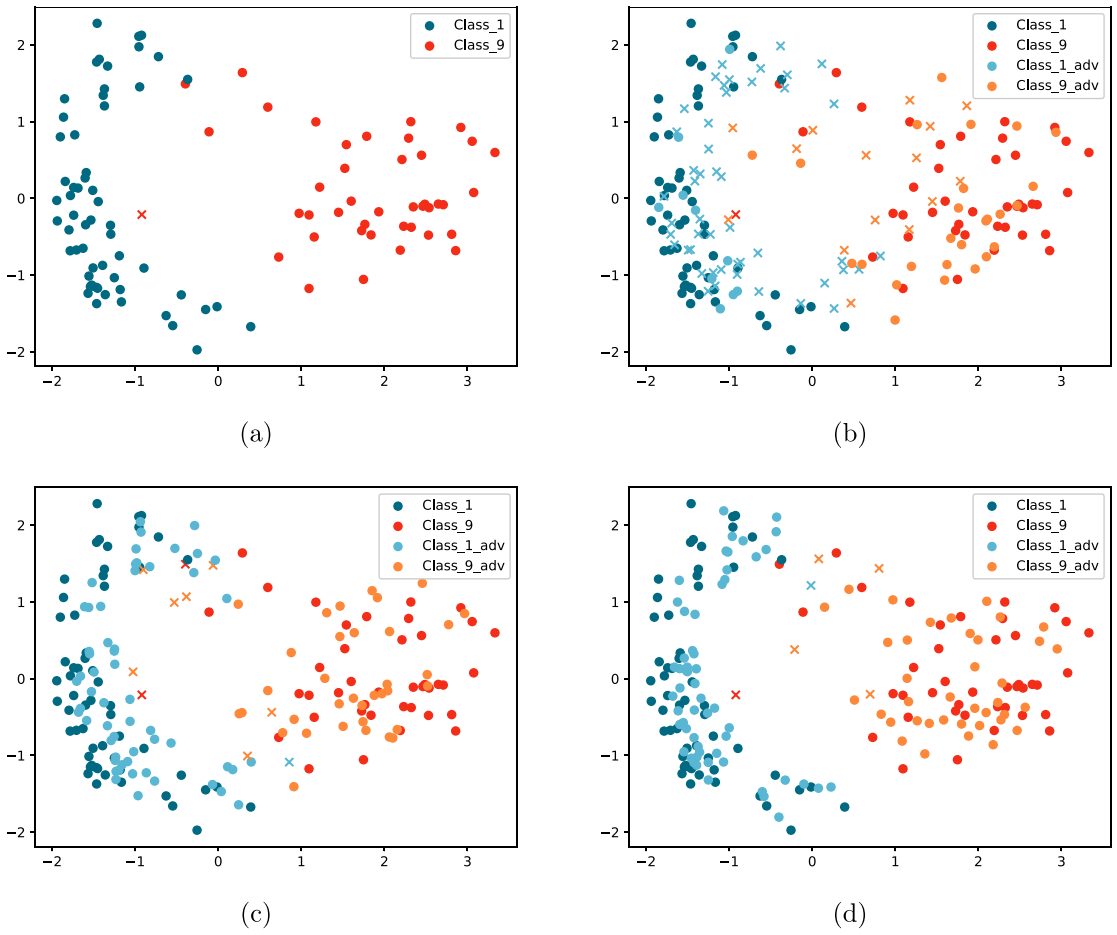


Figure 2. Two-dimensional visualization of the embedding space of the QNN classifier trained on the MNIST dataset.

4. Numerical experiments

In this section, we assess the effectiveness of our defensive strategy through numerical simulations using Pennylane [46]. At the outset, we implemented our approach on the 'moons' dataset sourced from scikit-learn [47], validating its compatibility with encoding techniques such as amplitude and angle encoding, and the influence of noise layers on the Hilbert space. Subsequently, we extended our experiments to encompass the ground state quantum dataset of the one-dimensional (1D) transverse field Ising model and the MNIST dataset [48].

4.1. Experimental setup

Dataset. For the synthetic dataset [47], we generated 1000 data points, introducing the noise level of 0.03, and classified these into class 0 and class 1. We reserved twenty percent of this data for testing. For the quantum dataset, following previous research [22, 23], we classified the ground state of the 1D transverse field Ising model as follows:

$$H_{\text{Ising}} = -\sum_{n=1}^{N-1} \sigma_n^z \sigma_{n+1}^z - J_x \sum_{n=1}^{N} \sigma_n^x, \tag{11}$$

where J_x represents the strength of the transverse field, and σ_n^z and σ_n^x denote the Pauli matrices of the nth spin. A quantum phase transition occurs at $J_x = 1$, between the paramagnetic phase for $0 < J_x < 1$ and the ferromagnetic phase for $J_x > 1$. We sampled an Ising model with a system size of 4 for various J_x values (from 0 to 2), and determined the corresponding ground states. These ground states and their labels constructed the dataset of 999 data points, from which we randomly selected 40% for testing. For the MNIST dataset [48], to mimic quantum computation with limited classical resources, we reduced the image size from 28×28 pixels to 16×16 pixels and normalized the data. The training involved 400 samples of digits 1 and 9 from the training set, while the testing used 1000 samples from the test set.

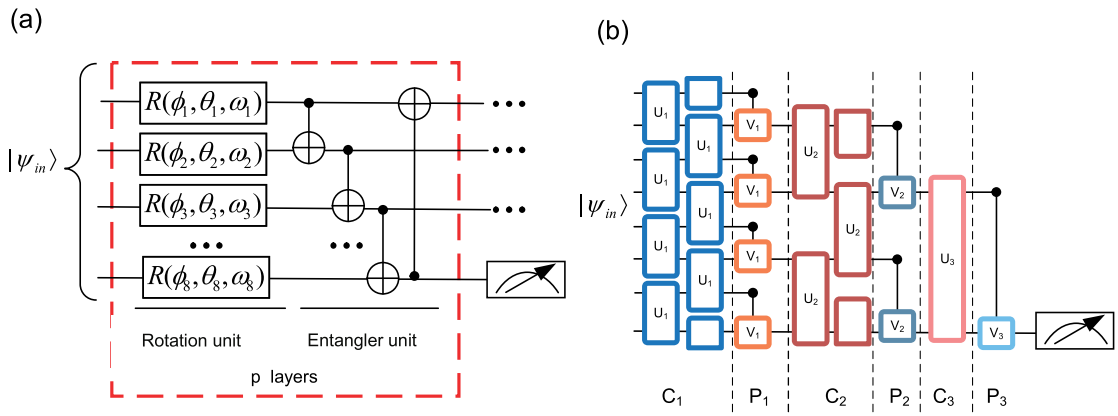


Figure 3. The structure of the QNNs used in numerical experiments. (a) A schematic diagram of the QNN structure with eight input quantum bits. (b) A schematic diagram of the QCNN structure with eight input quantum bits.

Network structure and hyperparameters. We adopted the general multi-layer variational neural network structure for the QNNs, aligning with previous research [22, 23]. Unless otherwise mentioned, we used amplitude encoding [22, 35] to transform classical data into quantum states. The circuit depth of the QNN was set to 10. We used the parameter shift rule [14, 19, 44, 45] to obtain the required gradients, and the quantum version of cross-entropy [22] served as the loss function. The Adam optimizer [49] minimized the loss function with a batch size of 64 and a learning rate of 0.005. We conducted training over 100 epochs for synthetic and quantum datasets, and 200 epochs for the MNIST dataset. In section 4.4.4, we also opted to utilize the QCNN [12]. Diagrams of both types of QNNs (QNN and QCNN) are illustrated in figure 3. The QNN comprises P layers, with each layer containing a rotation unit and an entangling unit. The rotation unit carries out arbitrary single-qubit operations, while the entangling unit consists of controlled NOT (CNOT) gates. Quantum data is converted into classes through the positive operator-valued measure (POVM) measurement of the Z-axis of the Bloch sphere. Furthermore, the σ^2 in (7) was set to $\frac{2\pi}{2^5}$, and the initial value of β was set to 0.25.

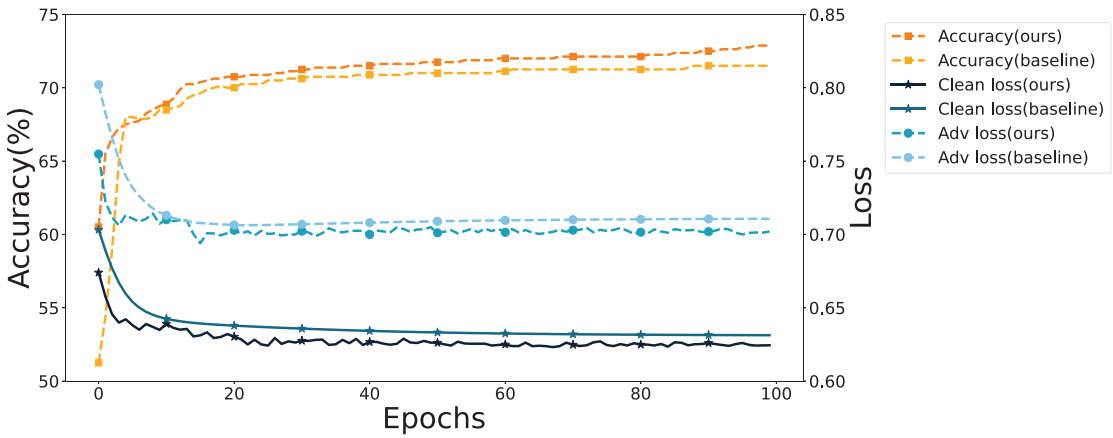
Attack settings. The attacks were assumed to be aware of the QNN's randomization and were modeled according to the quantum adversarial training setup by Lu *et al* [22], using BIM to generate adversarial examples with an iteration count of 3 and a step size of 0.05. Unless otherwise specified, experiments were evaluated based on the attack performance with $\epsilon = 0.15$. For the black-box attack, we used classical network architectures identical to that of [22], with the final layer of the network outputting only two classes. We utilized a model based on a fully-connected neural network (FNN) and a convolutional neural network (CNN). The FNN includes two hidden layers (comprising 512 and 53 neurons respectively), two Dropout layers to mitigate overfitting, and a Softmax output layer with two units. The CNN contains three convolution layers (with 64, 128, and 64 filters and filter sizes of 8×8, 4×4, and 2×2 respectively), followed by a Flatten layer, and ultimately an output layer with two units.

Baseline. To the best of our knowledge, quantum adversarial training [22] serves as the most prevalent defense mechanism in quantum adversarial learning [22, 24, 50], which we use as the baseline defense model. To train the baseline model, at each epoch, we first generate the corresponding adversarial examples from clean samples and use them together as the training dataset for retraining. Additionally, we discuss several defense mechanisms, such as Gaussian data augmentation [51] and the use of depolarizing noise to protect quantum classifiers [33].

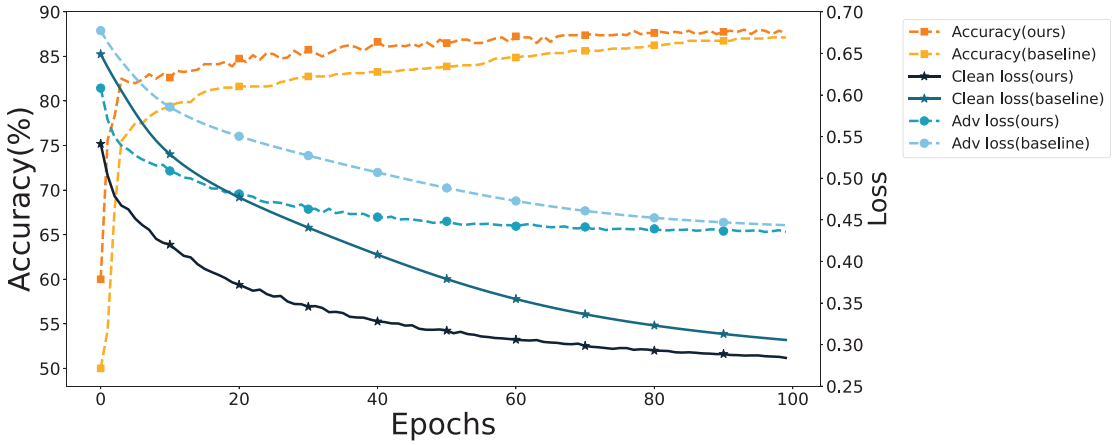
4.2. Synthetic dataset

We employed amplitude encoding [22, 35] and angle encoding [14, 36] to convert data from the synthetic dataset into quantum states. The training of QNNs was accomplished using both baseline training and our proposed training. As displayed in figure 4, our approach significantly reduced the training loss when compared to the baseline, irrespective of the encoding and data type (clean or adversarial). Notably, the smaller training loss often implies a more robust model [52]. As evidenced in table 1, our strategy improved the robustness of the synthetic dataset by approximately 3%–4%, relative to the baseline.

Figure 5 illustrates the data distribution in Hilbert space at varying stages of amplitude encoding, where adversarial data is produced by FGSM [22, 27]. Figures 5(a) and (b) depict the training set distribution in Hilbert space before training, for models with and without the noise layer. Figures 5(c) and (d) display the adversarial data distribution from the test set in Hilbert space, after standard QNNs training, irrespective of



(a) Amplitude encoding.



(b) Angle encoding.

Figure 4. Training loss and accuracy of QNNs for classifying synthetic datasets during training.

Table 1. Accuracy of FGSM and BIM on clean and adversarial data under attack on synthetic datasets. Best results are shown in bold.

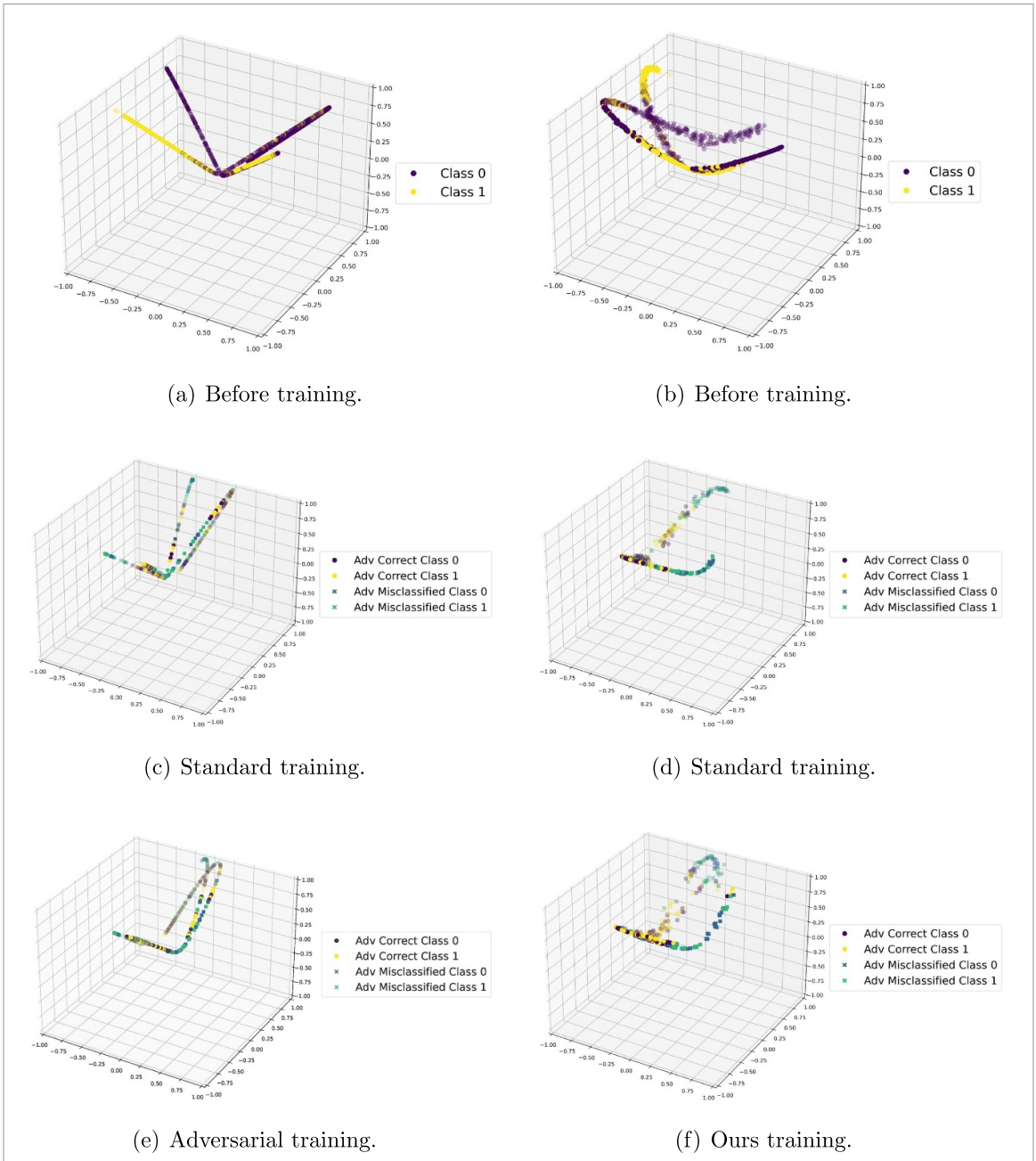
Model	Encoding	Clean	FGSM	BIM
Baseline	Amplitude	**74.00%**	55.00%	54.50%
	Angle	84.50%	74.50%	71.00%
This work	Amplitude	**74.00%**	**57.00%**	**55.50%**
	Angle	**89.00%**	**78.50%**	**74.00%**

the noise layer's presence. Finally, figures 5(e) and (f) present the adversarial data distribution from the test set in Hilbert space, after training with both the baseline and our method.

Insights from figures 5(a) and (b) suggest that the training data presents a 'linear' distribution in the Hilbert space in QNNs without the noise layer. Nonetheless, data in Hilbert space appears more dispersed when the QNN incorporates the noise layer, compared to the model without it. This attribute might aid in accurately classifying as many unseen training points as possible [36], and could be beneficial for subsequent training [53].

4.3. Quantum dataset

Contrary to classical data inputs, QNNs can directly categorize quantum data. Figure 6 contrasts the learning curves of three models on the quantum dataset: 1) the baseline QNN, 2) the QNN with the noise layer but without the combined loss computed using (9), and 3) the QNN with the noise layer that employs the combined loss calculated with (9)—our proposed approach. Absent the combined loss, the training loss barely differs from the baseline, on both clean and adversarial data. Nevertheless, our strategy significantly reduces loss on both data types compared to the baseline.



(a) Before training. (b) Before training.

(c) Standard training. (d) Standard training.

(e) Adversarial training. (f) Ours training.

Figure 5. The distribution of data at different stages in Hilbert space.

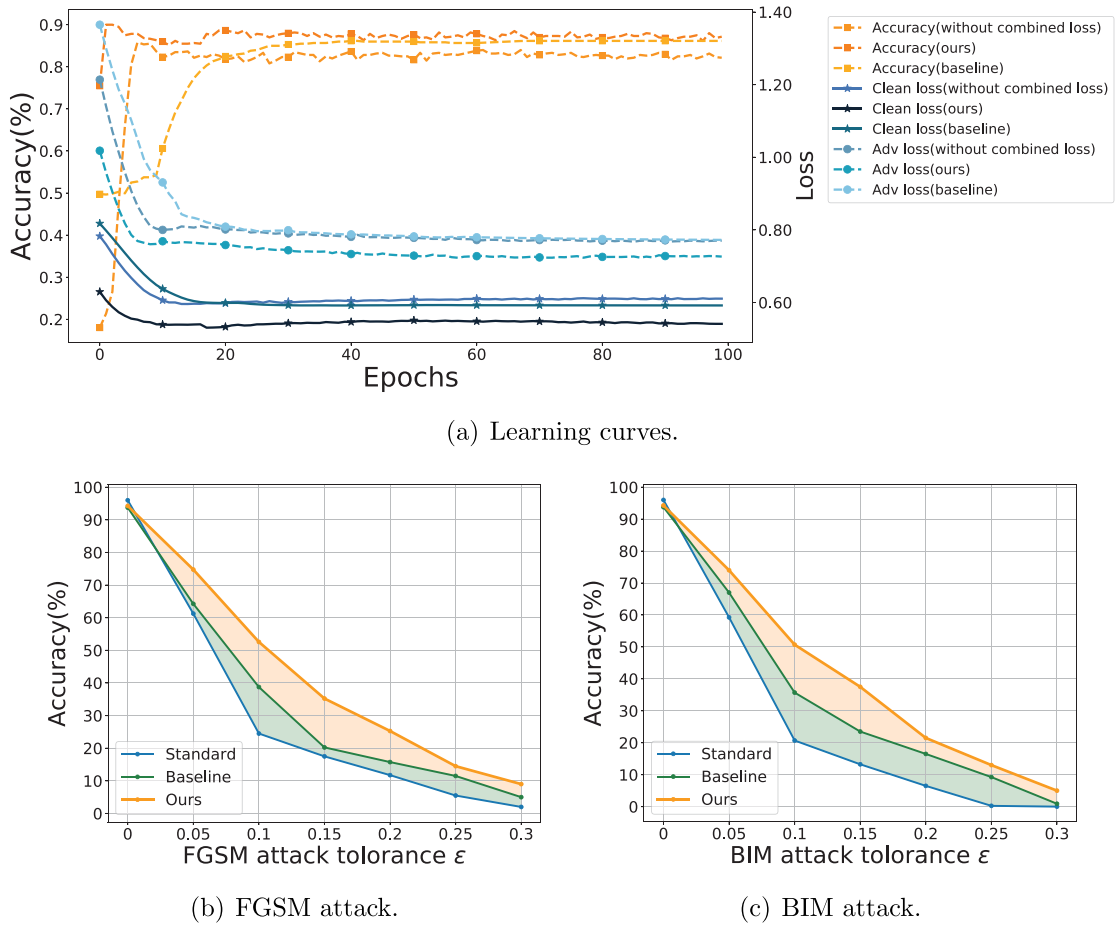
Subsequently, figures 6(b) and (c) offer the performance comparison of the standard training model, the baseline, and our model on the quantum dataset test set, under the adversarial perturbations of FGSM [22, 27] and BIM [22, 39]. These diagrams clearly demonstrate that adversarial training of the baseline model has somewhat enhanced the model's robustness. However, notably, our method surpasses the baseline.

4.4. MNIST dataset

In this section, we initially delve into QNNs with noise layers, investigating from three distinct perspectives utilizing the MNIST dataset [48]: (1) factors influencing the limiting coefficient β, (2) the implications of injecting noise layers at different stages, and (3) the effect of σ^2 in (7). Subsequently, we evaluate their robustness under white-box and black-box settings.

4.4.1. Factors affecting the constraint coefficient β

As discussed in section 3.1, training the constraint coefficient β requires the combined loss. We conducted training experiments on QNNs under different conditions, including QNNs without using the combined loss and without noise injection (referred to as QNN-V), QNNs without using the combined loss but with noise injection (referred to as QNN-N), QNNs using the combined loss but without noise injection (referred



(a) Learning curves.

(b) FGSM attack.

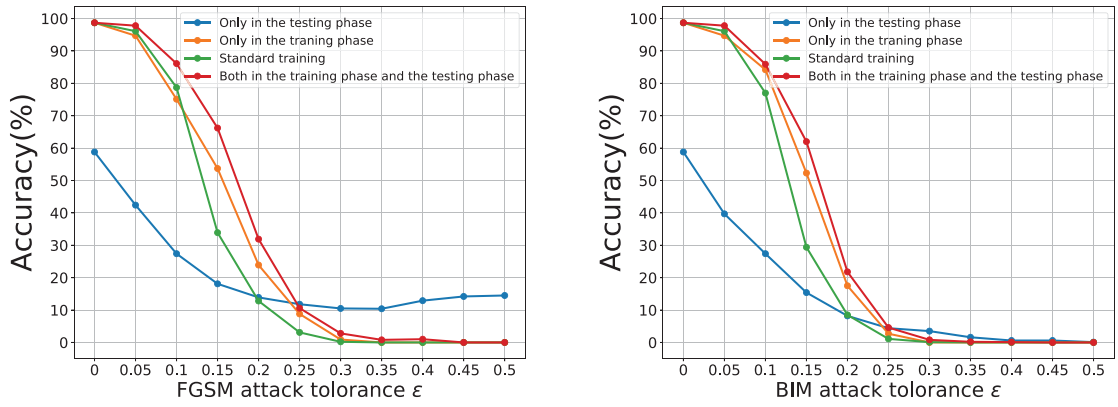
(c) BIM attack.

Figure 6. Numerical simulation results on quantum datasets.

Table 2. Convergence of constraint coefficients and performance of QNN with noisy layers. (Top) Constraint coefficients under various training methods. (Bottom) Accuracy of testing clean and adversarial perturbed data under FGSM and BIM attacks. Best results are shown in bold.

Layer index	QNN-V	QNN-N	QNN-C	QNN-CN
Layer 1	0.2580	0.0007	0.2304	0.0236
Layer 2	0.2667	0.0009	0.2569	0.0047
Layer 3	0.2287	0.0031	0.2378	0.0040
Layer 4	0.2606	0.0008	0.2861	0.0142
Layer 5	0.2534	0.0000	0.2562	0.0023
Layer 6	0.2724	0.0048	0.2919	0.0101
Layer 7	0.2465	0.0016	0.2837	0.0210
Layer 8	0.2462	0.0006	0.2134	0.0017
Layer 9	0.2637	0.0007	0.2353	0.0163
Layer 10	0.2529	0.0006	0.2241	0.0188
Clean	98.60%	**98.90%**	98.80%	98.70%
FGSM	32.90%	31.25%	48.20%	**66.20%**
BIM	28.20%	26.70%	42.20%	**62.00%**

to as QNN-C), and QNNs using the combined loss and with noise injection (referred to as QNN-CN), to compare the convergence of training with noise. As shown in table 2, during normal training (i.e. QNN-V), the value of β fluctuates around 0.25, indicating that the optimizer barely optimizes β. When only noise is injected (i.e. QNN-N), β converges to a negligible value. In contrast, when trained with the combined loss and noise injection (i.e. QNN-CN), the optimizer can optimize β to an appropriate value for controlling the noise level. Furthermore, noise injection is crucial for optimization, as shown in table 2. Using only the combined loss without noise injection (i.e. QNN-C) also leads to the failure of optimizing β.



(a) FGSM attack. (b) BIM attack.

Figure 7. Model's robust accuracy under different adversarial perturbations on the MNIST dataset with noise layers at different stages.

4.4.2. Impact of noise injection phase

In this section, we demonstrate the necessity of using noise layers during both the training and testing phases to improve a model's adversarial robustness. We experimented with four different models: the standard model without any noise layers, the model with noise layers only during testing, the model with noise layers only during training, and the model with noise layers during both phases. For the models with noise layers, we applied the combined loss to ensure optimal performance.

Figure 7 displays the impact of different adversarial attacks using FGSM and BIM on the performance of the four models on the MNIST test set. It should be noted that when $\epsilon = 0$, the robust accuracy degrades to the accuracy of clean samples.

As mentioned earlier, the primary purpose of noise injection is to fool gradient-based adversarial attacks. The simplest idea is to pretrain the QNN and then add noise layers to each layer of the QNN during the testing phase. Unfortunately, as shown in figure 7, this method, adding noise layers only during the testing phase, leads to a significant decrease in both accuracy and robustness of the QNN. Moreover, we found that adding noise layers during the testing stage is important. In figure 7, we compare models that only add noise layers during the training stage with those that add noise layers during both training and testing stages. We found that the former had much lower robustness performance compared to the latter. These results suggest that if the noise layer is not used at some stage, the model's adversarial robustness will be reduced.

In summary, our experimental results demonstrate that injecting noise layers during both the training and testing phases and using the combined loss to train the control coefficient β are crucial for improving the adversarial robustness of QNNs.

4.4.3. Effect of σ^2

Figure 8 highlights the impact of both the standard training and our proposed approach on the QNNs' accuracy for clean and adversarial data at different σ^2 values on the MNIST dataset. In the majority of instances, QNNs trained using our method demonstrate substantial stability across various σ^2 values.

Nonetheless, when $\sigma^2 = \frac{2\pi}{2^1}$, a significant decrease in the QNN's accuracy is observed, regardless of the sample type. This observation suggests that the incorporation of the noise layer could potentially impair the QNN's performance at notably high σ^2 values. Despite this, it is critical to remember that $\sigma^2 = \frac{2\pi}{2^1}$ signifies a relatively large value. Therefore, we believe that within a reasonable spectrum of σ^2, the noise's impact on the QNN's performance remains marginal or negligible.

4.4.4. White-box robustness

We evaluated the robustness of QNNs with added noise layers from two perspectives: circuit depth and network architecture. For different circuit depths, we conducted experiments on defensive baseline models and QNNs with noise layers added before each rotation unit layer. Additionally, we selected the QCNN depicted in figure 3 as an alternative network structure to demonstrate the generality of our approach, with the noisy layer added before the quantum convolutional layer. Figure 9 illustrates the impact of circuit depth on robust accuracy. It should be noted that we only trained the QNN with a circuit depth of 40 for 100 epochs. The experimental results of the QCNN are listed in table 3.

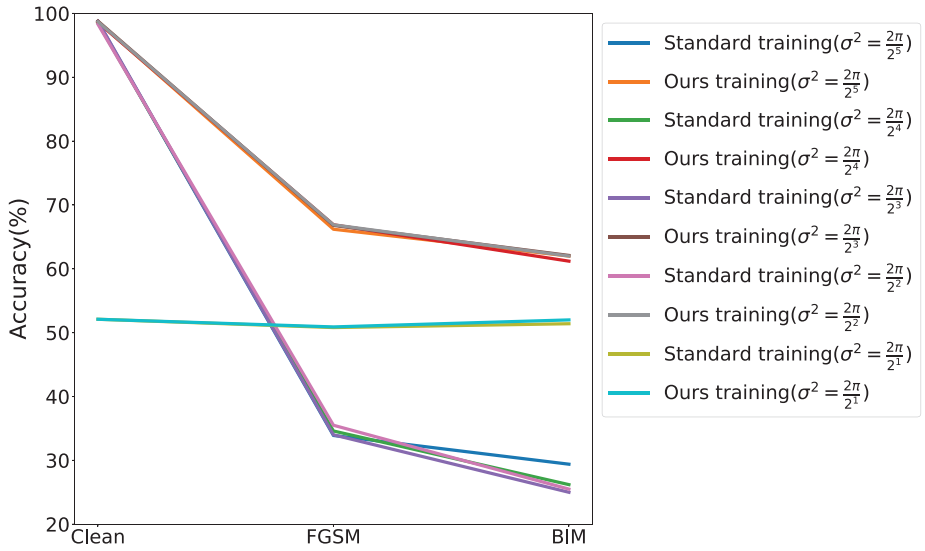


Figure 8. Robust accuracy on the MNIST dataset for different values of σ^2.

As shown in figure 9, our method achieved better robustness and had better clean accuracy compared to the baseline. We observed that increased circuit depths contributed to improved model robustness, consistent with the findings of Lu *et al* [22]. Furthermore, when the testing ϵ is greater than the trained value ($\epsilon = 0.15$), the model's robustness drops sharply, aligning with the conclusion of Madry *et al* [28]. However, our method offers greater resilience against adversarial attacks, delaying the onset of robustness collapse. Table 3 demonstrates that our method also yields a significant improvement in the adversarial robustness of the QCNN model.

4.4.5. Black-box robustness

In this subsection, we evaluate our proposed approach against transferable attacks [22, 24]. Transferable attacks involve training a source model to generate adversarial examples that can be used to attack a target model. Considering that adversaries may not have access to quantum devices and might only have classical resources at their disposal, we employ classical models to generate adversarial examples for the source model, including the FNN and the CNN mentioned in the section 4.1 attack settings. We adopt the network architecture, training process, and adversarial example generation methods (such as FGSM, BIM, and the momentum iterative method (MIM) proposed by [54]) as suggested by [22].

The black-box robustness of all defense models is presented in table 4. Similarly, our proposed method demonstrates superior robustness compared to adversarial training, highlighting the effectiveness of our approach in countering transferable attacks.

4.5. Discussion

As discussed in section 2.3, several adversarial defense approaches have been proposed to secure quantum classifiers for practical applications. Among them, quantum adversarial training [22, 24] is the most popular and reliable method, which we use as the baseline for our defense model. Additionally, we compare our proposed method with other similar approaches [33, 51].

The distinction between Gaussian data augmentation [51] and our method is that the former only adds Gaussian noise to images during training, while we add noise layers to the network to defend against adversarial attacks during both training and testing phases. During training, the noise layers help the optimizer find robust variation parameters for perturbed inputs. During testing, the noise can perturb gradients to deceive gradient-based attacks. Furthermore, Gaussian data augmentation cannot defend against adaptive attacks [55]: once the adversaries are aware of the image enhancement process, they can generate adversarial examples to attack the model. In contrast, our experiments are conducted under the assumption that the adversaries know the randomization process.

Protecting quantum classifiers using quantum noise [33] is a related method that injects noise at different locations in QNNs. This method adds depolarizing noise to QNNs and introduces the concept of differential privacy to defend against adversarial attacks. Although this method provides theoretical guarantees, using differential privacy for defense will compromise the accuracy of clean data. Moreover, in classical adversarial

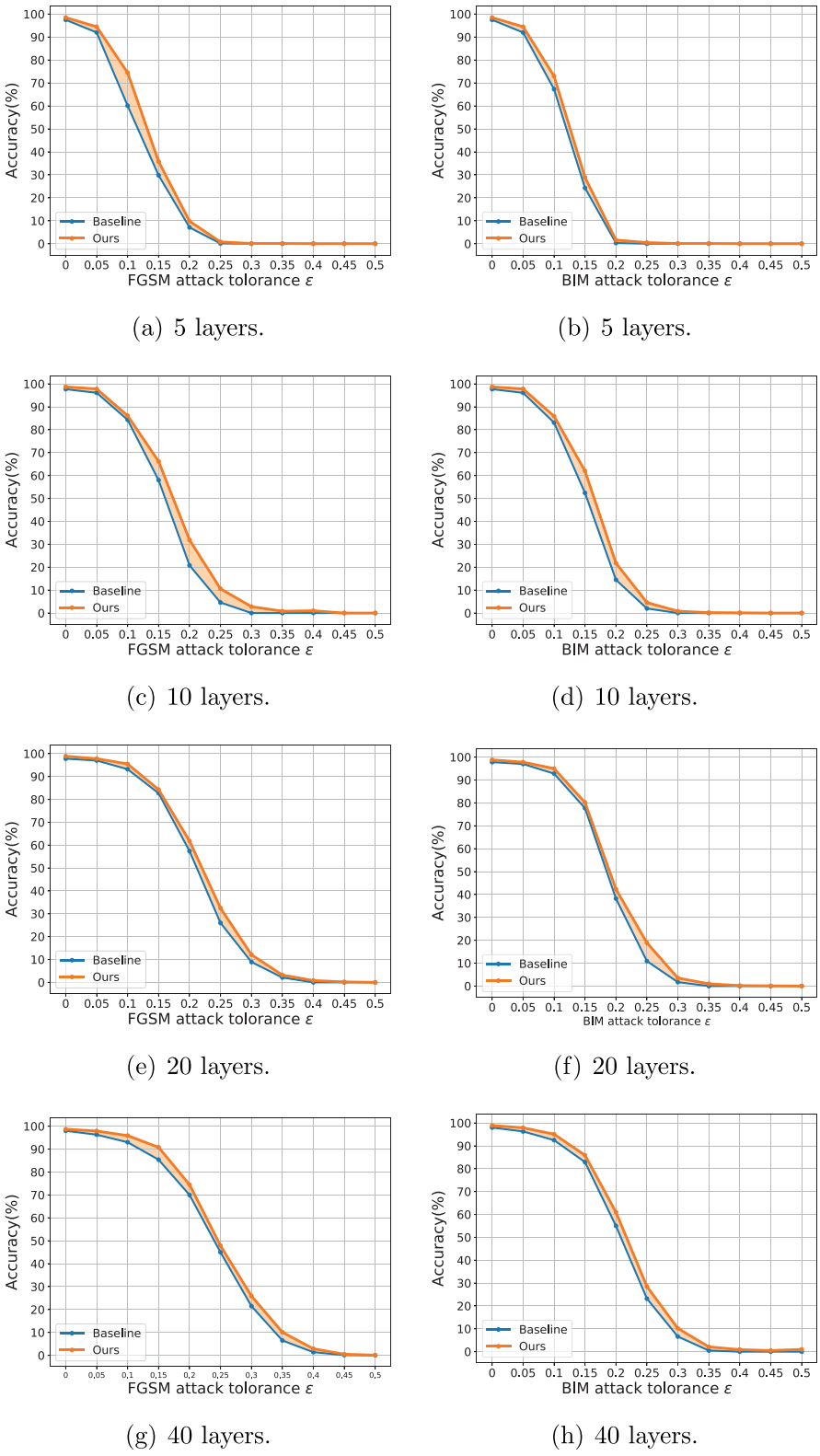


Figure 9. Robust accuracy of QNNs with varying circuit depths on the MNIST dataset under different levels of attack. Our approach clearly outperforms the baseline.

machine learning, differential privacy performs poorly in resisting attacks based on the L_∞-norm. Additionally, they require manual configuration of the noise level; although our method also requires setting the noise level hyperparameters, it can automatically adjust the noise level through trainable constraint coefficients.

Table 3. Accuracy of clean and adversarial perturbation data under FGSM and BIM attacks on QCNN. Best results are shown in bold.

	No defense	Baseline	This work
Clean	97.75%	97.10%	**98.75%**
FGSM	20.50%	55.00%	**61.35%**
BIM	17.00%	45.30%	**53.25%**

Table 4. Robust accuracy under transfer attack on MNIST dataset. Best results are shown in bold.

Attack model	Defense method	FGSM	BIM	MIM
CNN	No defense	63.90%	66.80%	67.00%
	Baseline	95.70%	96.00%	95.10%
	This work	**96.60%**	**97.60%**	**96.60%**
FNN	No defense	44.60%	43.90%	37.80%
	Baseline	82.80%	86.50%	84.10%
	This work	**88.40%**	**92.20%**	**90.20%**

5. Conclusion and future work

In this paper, we propose adding noise layers in QNNs as a regularization technique aimed at improving the generalization of quantum models for both clean and robust performance. The noise intensity within these layers can be controlled by solving a min-max optimization problem during adversarial training. Our numerical experimental results demonstrate that our method effectively withstands white-box and black-box attacks and outperforms the state-of-the-art defense methods in terms of accuracy on both clean and adversarial perturbed data.

It is important to note that while adversarial training is an effective approach for enhancing model robustness and maintaining high accuracy on both clean and adversarial data, it is computationally demanding. Besides computing gradients for updating variational parameters, generating adversarial examples requires multiple gradient computations. In fact, developing a robust model through adversarial training can take approximately 3 to 30 times longer than standard model training. In future work, we plan to investigate methods to improve quantum model stability and reduce computational overhead. We believe that further research in this area is crucial for the development of safe and reliable quantum AI technology.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 62076042), the National Key Research and Development Plan of China, Key Project of Cyberspace Security Governance (No. 2022YFB3103103), the Key Research and Development Project of Sichuan Province (Nos. 2022YFS0571, 2021YFSY0012, 2021YFG0332, 2020YFG0307).

ORCID iD

Chenyi Huang https://orcid.org/0000-0002-7664-0720

References

[1] Simonyan K and Zisserman A 2014 Very deep convolutional networks for large-scale image recognition (arXiv:1409.1556)
[2] Sutskever I, Vinyals O and Le Q V 2014 Sequence to sequence learning with neural networks *Advances in Neural Information Processing Systems* vol 27
[3] Chen C, Seff A, Kornhauser A and Xiao J 2015 Deepdriving: Learning affordance for direct perception in autonomous driving *Proc. of the IEEE int. conf. on computer vision* pp 2722–30 (available at: https://openaccess.thecvf.com/content_iccv_2015/html/Chen_DeepDriving_Learning_Affordance_ICCV_2015_paper.html)
[4] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning *Nature* **549** 195–202
[5] Ciliberto C, Herbster M, Davide Ialongo A, Pontil M, Rocchetto A, Severini S and Wossnig L 2018 Quantum machine learning: a classical perspective *Proc. R. Soc.* A **474** 20170551
[6] Dunjko V and Briegel H J 2018 Machine learning and artificial intelligence in the quantum domain: a review of recent progress *Rep. Prog. Phys.* **81** 074001

[7] Huang H-Y, Kueng R, Torlai G, Albert V V and Preskill J 2022 Provably efficient machine learning for quantum many-body problems *Science* **377** eabk3333
[8] Xiao T, Huang J, Li H, Fan J and Zeng G 2022 Intelligent certification for quantum simulators via machine learning *npj Quantum Inf.* **8** 138
[9] Xiao T, Fan J and Zeng G 2022 Parameter estimation in quantum sensing based on deep reinforcement learning *npj Quantum Inf.* **8** 2
[10] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[11] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)
[12] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273–8
[13] Zoufal C, Lucchi A and Woerner S 2019 Quantum generative adversarial networks for learning and loading random distributions *npj Quantum Inf.* **5** 103
[14] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309
[15] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 Circuit-centric quantum classifiers *Phys. Rev.* A **101** 032308
[16] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 Hierarchical quantum classifiers *npj Quantum Inf.* **4** 65
[17] Benedetti M, Garcia-Pintos D, Perdomo O, Leyton-Ortega V, Nam Y and Perdomo-Ortiz A 2019 A generative modeling approach for benchmarking and training shallow quantum circuits *npj Quantum Inf.* **5** 45
[18] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324
[19] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001
[20] Cerezo M *et al* 2021 Variational quantum algorithms *Nat. Rev. Phys.* **3** 625–44
[21] Qian Y, Wang X, Du Y, Wu X and Tao D 2022 The dilemma of quantum neural networks *IEEE Trans. Neural Netw. Learn. Syst.* 1–13
[22] Lu S, Duan L-M and Deng D-L 2020 Quantum adversarial machine learning *Phys. Rev. Res.* **2** 033212
[23] Gong W and Deng D-L 2022 Universal adversarial examples and perturbations for quantum classifiers *Natl Sci. Rev.* **9** nwab130
[24] Ren W *et al* 2022 Experimental quantum adversarial learning with programmable superconducting qubits *Nat. Comput. Sci.* **2** 711–7
[25] Liu N and Wittek P 2020 Vulnerability of quantum classification to adversarial perturbations *Phys. Rev.* A **101** 062331
[26] Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I and Fergus R 2013 Intriguing properties of neural networks (arXiv:1312.6199)
[27] Goodfellow I J, Shlens J and Szegedy C 2014 Explaining and harnessing adversarial examples (arXiv:1412.6572)
[28] Madry A, Makelov A, Schmidt L, Tsipras D and Vladu A 2017 Towards deep learning models resistant to adversarial attacks (arXiv:1706.06083)
[29] Li B, Chen C, Wang W and Carin L 2019 Certified adversarial robustness with additive noise *Advances in Neural Information Processing Systems* vol 32
[30] Cohen J, Rosenfeld E and Kolter Z 2019 Certified adversarial robustness via randomized smoothing *Int. Conf. on Machine Learning* (PMLR) pp 1310–20
[31] Lecuyer M, Atlidakis V, Geambasu R, Hsu D and Jana S 2019 Certified robustness to adversarial examples with differential privacy *2019 IEEE Symp. on Security and Privacy (SP)* (IEEE) pp 656–72
[32] Tsipras D, Santurkar S, Engstrom L, Turner A and Madry A 2018 Robustness may be at odds with accuracy (arXiv:1805.12152)
[33] Du Y, Hsieh M-H, Liu T, Tao D and Liu N 2021 Quantum noise protects quantum classifiers against adversaries *Phys. Rev. Res.* **3** 023153
[34] Schuld M, Sweke R and Jakob Meyer J 2021 Effect of data encoding on the expressive power of variational quantum-machine-learning models *Phys. Rev.* A **103** 032430
[35] Schuld M, Fingerhuth M and Petruccione F 2017 Implementing a distance-based classifier with a quantum interference circuit *Europhys. Lett.* **119** 60002
[36] LaRose R and Coyle B 2020 Robust data encodings for quantum classifiers *Phys. Rev.* A **102** 032420
[37] Henderson M, Shakya S, Pradhan S and Cook T 2020 Quanvolutional neural networks: powering image recognition with quantum circuits *Quantum Mach. Intell.* **2** 2
[38] Stoudenmire E and Schwab D J 2016 Supervised learning with tensor networks *Advances in Neural Information Processing Systems* vol 29
[39] Kurakin A, Goodfellow I J and Bengio S 2018 Adversarial examples in the physical world *Artificial Intelligence Safety and Security* (Chapman and Hall/CRC) pp 99–112
[40] Papernot N, McDaniel P, Goodfellow I, Jha S and Swami A 2017 Practical black-box attacks against machine learning *Proc. 2017 ACM on Asia Conf. on Computer and Communications Security* pp 506–19
[41] Liu Y, Chen X, Liu C and Song D 2016 Delving into transferable adversarial examples and black-box attacks (arXiv:1611.02770)
[42] Liu X, Cheng M, Zhang H and Hsieh C-J 2018 Towards robust neural networks via random self-ensemble *Proc. European Conf. on Computer Vision (ECCV)* pp 369–85
[43] He Z, Siraj Rakin A and Fan D 2019 Parametric noise injection: trainable randomness to improve deep neural network robustness against adversarial attack *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 588–97
[44] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 Evaluating analytic gradients on quantum hardware *Phys. Rev.* A **99** 032331
[45] Harrow A W and Napp J C 2021 Low-depth gradient measurements can improve convergence in variational hybrid quantum-classical algorithms *Phys. Rev. Lett.* **126** 140502
[46] Bergholm V *et al* 2018 Pennylane: automatic differentiation of hybrid quantum-classical computations (arXiv:1811.04968)
[47] Pedregosa F *et al* 2011 Scikit-learn: machine learning in python *J. Mach. Learn. Res.* **12** 2825–30 (arXiv:1201.0490)
[48] LeCun Y 1998 The mnist database of handwritten digits (available at: http://yann.lecun.com/exdb/mnist/)
[49] Kingma D P and Jimmy B 2014 Adam: a method for stochastic optimization (arXiv:1412.6980)
[50] Guan J, Fang W and Ying M 2020 Robustness verification of quantum machine learning *CoRR* (available at: https://link.springer.com/chapter/10.1007/978-3-030-81685-8_7#citeas)
[51] Zantedeschi V, Nicolae M-I and Rawat A 2017 Efficient defenses against adversarial attacks *Proc. 10th ACM Workshop on Artificial Intelligence and Security* pp 39–49

[52] Du Y, Yang Y, Tao D and Hsieh M-H 2022 Demystify problem-dependent power of quantum neural networks on multi-class classification (arXiv:2301.01597)
[53] Lloyd S, Schuld M, Ijaz A, Izaac J and Killoran N 2020 Quantum embeddings for machine learning (arXiv:2001.03622)
[54] Dong Y, Liao F, Pang T, Su H, Zhu J, Hu X and Li J 2018 Boosting adversarial attacks with momentum *Proc. of the IEEE conf. on computer vision and pattern recognition*In pp 9185–93 (available at: https://openaccess.thecvf.com/content_cvpr_2018/html/Dong_Boosting_Adversarial_Attacks_CVPR_2018_paper.html)
[55] Carlini N and Wagner D 2017 Magnet and 'efficient defenses against adversarial attacks' are not robust to adversarial examples (arXiv:1711.08478)