

A backdoor attack against quantum neural networkswith limited information

Chen-Yi Huang(黄晨猗), and Shi-Bin Zhang(张仕斌)
Citation: Chin. Phys. B, 2023, 32 (**10**): 100306. DOI: 10.1088/1674-1056/acd8ab
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Thermodynamic limit of the *XXZ* central spin model with an arbitrary central magnetic field

Fa-Kai Wen(温发楷) and Kun Hao(郝昆)
Chin. Phys. B, 2023, 32 (**9**): 090307. DOI: 10.1088/1674-1056/acb426

Commensurate and incommensurate Haldane phases for a spin-1 bilinear-biquadratic model

Yan-Wei Dai(代艳伟), Ai-Min Chen(陈爱民), Xi-Jing Liu(刘希婧), and Yao-Heng Su(苏耀恒)
Chin. Phys. B, 2023, 32 (**9**): 090304. DOI: 10.1088/1674-1056/acac0c

Degenerate polarization entangled photon source based on a single Ti-diffusion lithium niobate waveguide in a polarization Sagnac interferometer

Yu Sun(孙宇), Chang-Wei Sun(孙昌伟), Wei Zhou(周唯), Ran Yang(杨然), Jia-Chen Duan(端家晨), Yan-Xiao Gong(龚彦晓), Ping Xu(徐平), and Shi-Ning Zhu(祝世宁)
Chin. Phys. B, 2023, 32 (**8**): 080308. DOI: 10.1088/1674-1056/acd8b3

Variational quantum simulation of thermal statistical states on a superconducting quantum processer

Xue-Yi Guo(郭学仪), Shang-Shu Li(李尚书), Xiao Xiao(效骁), Zhong-Cheng Xiang(相忠诚), Zi-Yong Ge(葛自勇), He-Kang Li(李贺康), Peng-Tao Song(宋鹏涛), Yi Peng(彭益), Zhan Wang(王战), Kai Xu(许凯), Pan Zhang(张潘), Lei Wang(王磊), Dong-Ning Zheng(郑东宁), and Heng Fan(范桁)
Chin. Phys. B, 2023, 32 (**1**): 010307. DOI: 10.1088/1674-1056/aca7f3

Purification in entanglement distribution with deep quantum neural network

Jin Xu(徐瑾), Xiaoguang Chen(陈晓光), Rong Zhang(张蓉), and Hanwei Xiao(肖晗微)
Chin. Phys. B, 2022, 31 (**8**): 080304. DOI: 10.1088/1674-1056/ac6330

A backdoor attack against quantum neural networks with limited information

Chen-Yi Huang(黄晨猗)[1,2] and Shi-Bin Zhang(张仕斌)[1,2,†]

[1] *College of Cyberspace Security, Chengdu University of Information Technology, Chengdu* 610225, *China*
[2] *Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu* 610225, *China*

(Received 14 February 2023; revised manuscript received 17 May 2023; accepted manuscript online 25 May 2023)

Backdoor attacks are emerging security threats to deep neural networks. In these attacks, adversaries manipulate the network by constructing training samples embedded with backdoor triggers. The backdoored model performs as expected on clean test samples but consistently misclassifies samples containing the backdoor trigger as a specific target label. While quantum neural networks (QNNs) have shown promise in surpassing their classical counterparts in certain machine learning tasks, they are also susceptible to backdoor attacks. However, current attacks on QNNs are constrained by the adversary's understanding of the model structure and specific encoding methods. Given the diversity of encoding methods and model structures in QNNs, the effectiveness of such backdoor attacks remains uncertain. In this paper, we propose an algorithm that leverages dataset-based optimization to initiate backdoor attacks. A malicious adversary can embed backdoor triggers into a QNN model by poisoning only a small portion of the data. The victim QNN maintains high accuracy on clean test samples without the trigger but outputs the target label set by the adversary when predicting samples with the trigger. Furthermore, our proposed attack cannot be easily resisted by existing backdoor detection methods.

Keywords: backdoor attack, quantum artificial intelligence security, quantum neural network, variational quantum circuit

PACS: 03.67.–a, 03.67.Lx, 03.67.Ac **DOI:** 10.1088/1674-1056/acd8ab

1. Introduction

Machine learning, particularly with the rapid advancement of deep learning models, has made remarkable strides in applications such as computer vision[1] and speech recognition.[2] Concurrently, the field of quantum computing has revolutionized traditional computing theories and practices. The growth of machine learning and quantum computing has stimulated researchers to enhance classical machine learning algorithms[3,4] by leveraging quantum-mechanical properties, such as superposition and entanglement.

In the era of noisy intermediate-scale quantum (NISQ) computers,[5] variational quantum algorithms[6] are considered to be suitable for NISQ devices due to their potential noise resilience. These algorithms offer the possibility of achieving the quantum advantage in solving practical problems. One specific approach is the use of variational quantum circuits, which consist of tunable parameters that can be optimized during the training process and have shown significant progress in supervised learning tasks,[7,8] unsupervised learning tasks,[9,10] and quantum learning tasks.[11]

Despite these advancements, recent studies[12–19] have shown that, like their classical deep learning counterparts,[20,21] quantum neural networks (QNNs) based on variational quantum algorithms lack robustness against adversarial attacks. Specifically, adversaries can introduce carefully crafted adversarial perturbations to the original samples, leading the QNN to misclassify.[14–16] These manipulated samples are referred to as adversarial examples,[20] with attacks typically occurring during the inference (testing) phase of quantum machine learning models.

However, the security risks to machine learning systems extend beyond the inference stage. During the training phase, adversaries can manipulate or compromise the accuracy of machine learning models by injecting malicious data samples in a type of attack known as a poisoning attack.[22,23] In contrast, Gu *et al.*[24] proposed the backdoor attack, where the aim is not to degrade the model's accuracy but to poison the training data by embedding a hidden backdoor into the target model during training. This backdoor is activated during the inference process. If the backdoor trigger is present in a given input, the model consistently predicts a specific target label (even if incorrect), posing a more covert and severe security threat.

While numerous studies have considered the security of quantum machine learning systems, they primarily focus on adversarial issues.[12–19] The impact of backdoor attacks on quantum machine learning systems has not been studied as thoroughly as adversarial attacks. In the realm of machine learning, which relies heavily on big data and computational resources, users often need to source public datasets from third parties to build effective models. As a result, adversaries can launch backdoor attacks by injecting backdoors into these public datasets, a scenario likely to be common in future quantum

[†]Corresponding author. E-mail: cuitzsb@cuit.edu.cn
© 2023 Chinese Physical Society and IOP Publishing Ltd

machine learning applications.[25]

However, designing backdoor attacks for quantum systems is challenging, and conventional backdoor attacks[24] aimed at classical neural networks are hard to apply to QNNs. Firstly, conventional backdoor attacks[24,26] require selecting a trigger (e.g., a fixed-size square trigger) to embed into the victim's classical neural network. Given the hardware limitations and noise interference, current QNNs have small input dimensions, making even a small fixed trigger (such as one requiring a single qubit) hard to embed into the victim's QNN input while maintaining the attack's stealthiness. Secondly, most backdoor attacks require changing the labels of training samples, and this attack modification may not be feasible in real-world scenarios. Chu *et al.*[27] recently made a significant contribution by proposing a circuit-level backdoor attack targeting QNNs, which can be implemented by inserting a few quantum gates into the victim's variational quantum circuit. However, this method requires the adversary to have complete knowledge of the QNN architecture, activate specific quantum gates, and is only successful against QNNs that employ angle encoding.

In this paper, inspired by previous studies[28,29] that employed universal adversarial perturbations as backdoor triggers, we propose a backdoor attack method against QNNs. This attack can be effectively implemented even when the adversary only has knowledge of the victim's QNN dataset. Our method has the following advantages: (1) It does not rely on a fixed backdoor trigger pattern and is difficult to detect. (ii) It can attack effectively without needing to alter the training process. (iii) It does not require changing the labels in the training dataset. (iv) In the most extreme scenarios, the adversary does not need to possess specific structural information about the QNN. Contrastingly, prior methods that used universal adversarial perturbations as backdoor triggers required placing triggers in specific areas of the data,[27] or intentionally mislabeling poisoned data.[29] We suggest generating a trigger by solving an optimization problem, which can associate non-target class data with target class data. This trigger will then be added to the data used to train the QNN. The victim's QNN will inadvertently embed a backdoor, which the adversary can activate using the trigger. We also evaluate defensive measures against the attack. The most popular backdoor defense detection methods, such as spectral signature detection[30] and activation clustering,[31] struggle to counteract our attack.

The rest of this paper is organized as follows: In Section 2, we introduce related work. In Section 3, we detail our threat model. Section 4 presents the entire method of the proposed backdoor attack. Section 5 introduces and analyzes experimental results. Finally, we conclude our work in Section 6.

2. Related work

2.1. Supervised machine learning

Classification tasks, specifically pattern recognition under supervised learning, are common problems in quantum machine learning. To formalize this problem, let $\mathcal{X}$ denote a set of inputs and $\mathcal{Y}$ a set of outputs. The aim of supervised learning is to develop a classifier $Q_\theta\colon \mathcal{X} \to \mathcal{Y}$ that maps each input to its corresponding target label $y \in \mathcal{Y}$, where θ represents the parameters of the classifier Q_θ. Supervised learning involves two stages: training and testing. During the training stage, given a dataset $\mathcal{D}_N = (x_1, y_1), \ldots, (x_N, y_N)$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, the learning algorithm's objective is to find optimal model parameters θ that minimize a predefined loss function, i.e.,

$$\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(Q_\theta(x_i), y_i), \tag{1}$$

where $\mathcal{L}$ is the predefined loss function. During the testing stage, the trained model Q_{θ^*} takes test examples as input and generates predictions.

2.2. Quantum neural network

Recent advancements in quantum computers' development have paved the way for utilizing quantum computing to solve classification problems, with QNNs deemed highly suitable for implementing classification tasks on NISQ computers.[5,32] A QNN typically consists of three parts: the encoding circuit, the parameterized circuit, and the measurement. QNNs require quantum states as inputs to the quantum circuits. To use quantum circuits for classical data classification, it is necessary to transform the given classical data input of dimension $\mathcal{D}$ into an n-qubit system in the quantum computer. This transformation is typically achieved through the quantum feature map $\varphi\colon \mathcal{D} \to \mathcal{H}$, where $\mathcal{H}$ is the Hilbert space. Various methods exist to encode data into quantum states, including amplitude encoding[7,14,15] and angle encoding.[33] However, for quantum datasets, encoding circuits are unnecessary, and data can be directly input into the parameterized circuit.

Once the data is encoded into a quantum state $|\psi_{\text{in}}\rangle$, which contains the complete information of the input sample to be classified, a series of unitary transformations involving multiple layers of parameterized quantum circuits can be applied to obtain the variational state

$$|\Psi\rangle = \prod_{\ell=1}^{M} V_\ell U_\ell \theta_\ell (|\psi_{\text{in}}\rangle), \tag{2}$$

where M represents the number of layers in the variational circuit, V_ℓ represents the non-parameterized unitary operation in the ℓ-th layer, $U_\ell\theta_\ell$ represents the unitary operation with variational parameters in the ℓ-th layer, and θ_ℓ represents all parameters in the ℓ-th layer. For a given QNN, the goal of training is

to find the optimal θ that minimizes the loss function between predictions and actual labels. It is worth noting that, similar to the universal approximation theorem in classical deep learning models,[34] a variational circuit with a sufficient number of layers can approximate any desired function.

Finally, the predicted labels for the input data are determined by measuring the expectation values of selected observables.

2.3. Adversarial attacks

Adversarial attacks are common security threats in quantum machine learning systems.[12–19] These attacks can be categorized as untargeted or targeted. Given a QNN model Q_{θ^*} with parameters θ^* and a clean input sample $x \in \mathcal{X}$, let us assume that the sample x is correctly classified by the model, i.e., $Q_{\theta^*}(x) = y$. In the case of non-targeted attacks, the goal of the adversary is to find a perturbation δ that causes the QNN to make a different prediction than the clean input x, i.e., $Q_{\theta^*}(x+\delta) \neq y$, where the new input $x+\delta$ is called an adversarial example x_{adv}.[20] The adversary needs to ensure that the adversarial perturbation is small and does not fundamentally affect the original input. Conversely, in the case of targeted attacks, the adversary's goal is to add an adversarial perturbation δ^t to the clean sample x to obtain an adversarial example x^t_{adv} with a specific label predicted by the QNN, i.e., $Q_{\theta^*}(x^t_{\text{adv}}) = y^t$, where $x^t_{\text{adv}} = x+\delta^t$ and y^t is a target label different from the correct label. Generally, adversarial perturbations are specific, different inputs will result in different perturbations. However, for universal adversarial perturbations,[13,35] the perturbations for different inputs are the same.

2.4. Backdoor attack

In backdoor attacks,[24] an adversary introduces a perturbation, known as a backdoor trigger, into the target model during the training phase of a machine learning system. The backdoor model behaves as expected on clean samples; however, it predicts samples containing the backdoor trigger as the class specified by the adversary. Formally, the adversary aims to use a backdoor trigger δ_{b} and a set of backdoor parameters θ^*_{b} to make the backdoor model $Q_{\theta^*_{\text{b}}}$ accurately predict the label y corresponding to the clean sample $x \in \mathcal{X}$, i.e., $Q_{\theta^*_{\text{b}}}(x) = y$. Nevertheless, when predicting a sample $x+\delta_{\text{b}}$ with a backdoor trigger δ_{b}, the model makes a target class prediction predetermined by the adversary, i.e., $Q_{\theta^*_{\text{b}}}(x+\delta_{\text{b}}) = y^t$, where y^t is the label of the target class sample.

Based on whether the injected samples maintain consistent features and labels, backdoor attacks can typically be classified into dirty-label[24] or clean-label attacks.[26] For dirty-label attacks, the adversary first adds the backdoor trigger to non-target class training samples and changes the labels of these samples to the target label for training. When the backdoor model is presented with input containing the backdoor trigger, the model's predictions can be manipulated during the inference process. However, this attack can be thwarted by manually checking or using data filtering methods to remove the contaminated input-label pairs. To improve the stealthiness of the backdoor attack, Turner *et al.*[26] proposed the clean-label backdoor attack. In that attack, the adversary does not need to modify the sample's labels, ensuring that the input of the poisoned sample aligns with its label. Thus, the clean-label attack is a more covert method.

Previous work[36] on quantum system backdoor attacks focused on quantum communication systems. Recently, Chu *et al.*[27] proposed a circuit-level backdoor attack against QNNs, where they assumed that the adversary could control the victim QNN's encoding layer and add quantum gates as backdoors. This approach implies that the adversary needs to understand the specific architecture of the encoding circuit in the QNN, and their method exclusively targets angle encoding.[27,33]

Both backdoor and adversarial attacks necessitate the addition of perturbations to clean inputs to deceive the model. However, there are key differences between the two types of attacks: adversarial attacks only manipulate the target model's inference process, while backdoor attacks occur during the training process. Additionally, in a backdoor attack, the adversary knows the specific attack perturbation, while the adversarial perturbation needs to be obtained by optimizing based on the target model's output. Several studies have attempted to combine universal adversarial perturbation techniques[35] with backdoor attacks. Zhao *et al.*[28] proposed using universal adversarial perturbations as a more powerful backdoor trigger for attacking video recognition models. However, their triggers required placement in a fixed region of the data. Zhang *et al.*[29] utilized targeted universal adversarial perturbations as triggers, but their attacks were conducted in the context of dirty label settings. Recently, Weng *et al.*[37] experimentally discovered that enhancing a model's adversarial robustness may make the model more susceptible to backdoor attacks.

Compared to existing quantum adversarial attacks,[12,13] our objective is not to design more effective adversarial attacks but rather to inject adversarial perturbations into QNNs in a way that remains undetected. Unlike the attack proposed by Chu *et al.*,[27] our method does not require the adversary to understand the specific architecture of the QNN or be confined to a specific encoding method.

3. Threat model

In this section, we define our threat model, which encompasses adversaries, attack scenarios, attack objectives, and at-

tack metrics.

3.1. Adversaries

In classical machine learning, adversaries can inject backdoors at any stage of deep models. They can activate the backdoor by modifying the neurons within the network[38] or changing the network's weights.[39] However, given that quantum computers are costly machines developed and operated by a select few, these direct manipulation attacks on the original model are impractical. Therefore, we adopt the more conservative assumptions of backdoor attacks:[24,26] adversaries can add backdoors to the dataset to accomplish the backdoor attack without knowledge of or changes to other training components such as the model architecture, training loss, or training schedule.

3.2. Attack scenarios

Due to the costs associated with preparing quantum bits and quantum gates, and the impact of noise on quantum systems, ordinary users typically lack the capacity to construct quantum models. Consequently, they often resort to training the QNNs through quantum cloud computing platforms rather than local training, with the server returning the results to the user. As machine learning models are data-driven, individuals or enterprises may not possess sufficient training data to ensure optimal model performance. Thus, users may need to gather data from third parties, such as public repositories, before deploying quantum models. Adversaries are able to embed backdoor triggers in these training data. For backdoor attacks under quantum machine learning, adversaries may or may not know the internal structure and parameter settings of the target quantum model. Even if adversaries are aware of the quantum model's internal structure, they may lack the means to construct a quantum model.

In the standard backdoor attack,[24] adversaries are permitted to inject a small number of poisoned samples into the training set of the victim model. During the inference process, adversaries use samples containing backdoor triggers to activate the backdoor, thereby altering the predicted results of the victim model, while the model behaves normally on clean data. Adversaries can modify the label attributes in the test set, such that the source label of the poisoned sample differs from the target label. Given that users are likely to manually check these obviously mislabeled input-label pairs when training quantum machine learning models, the attack may fail. Therefore, we execute attacks under the setting of clean-label backdoors,[26] where the source label of the poisoned sample aligns with its target label.

3.3. Attack objectives and attack metrics

Adversaries have three main objectives. The first is to ensure that the backdoor trigger is sufficiently covert, to avoid arousing the suspicion of ordinary users who otherwise may not select the dataset for training the QNN. The second objective is to secure a successful attack. Adversaries need to evade backdoor detection[30,31] as much as possible. The third objective is to inject the backdoor into the QNN without significantly reducing the accuracy of clean data. The QNN should maintain comparable accuracy on clean data, otherwise, users may re-collect data or refuse to train the model. Existing literature identifies two primary performance metrics for backdoor attacks. The first is clean accuracy (CA), which refers to the model's accuracy on clean test data. The second is attack success rate (ASR), representing the percentage of non-target samples with triggers that the model predicts as target samples. The higher the percentage, the more effective the attack strategy. For a given poison ratio, a successful backdoor attack should have a high ASR and not affect the CA.

4. Proposed attack method

4.1. Attack process

Figure 1 depicts the detailed workflow of the proposed attack when the target label (target class) is set to 7. The victim model is a generic QNN architecture comprising an encoding layer S_x, a variational circuit block $U(\theta)$, and a measurement layer. The proposed attack method includes three main components. The initial step is trigger generation: we utilize non-target class samples and a proxy model to generate a backdoor trigger. The subsequent step is trigger insertion: we randomly select some data from the target class to implant the trigger, thus creating poisoned samples. Once the user utilizes the poisoned data to train the quantum model, the trigger will be embedded in the trained model. The final step is inference: after the victim QNN is trained with the poisoned samples, the model will recognize samples with the trigger as the target class 7. Simultaneously, the model can correctly identify clean samples that lack the trigger.

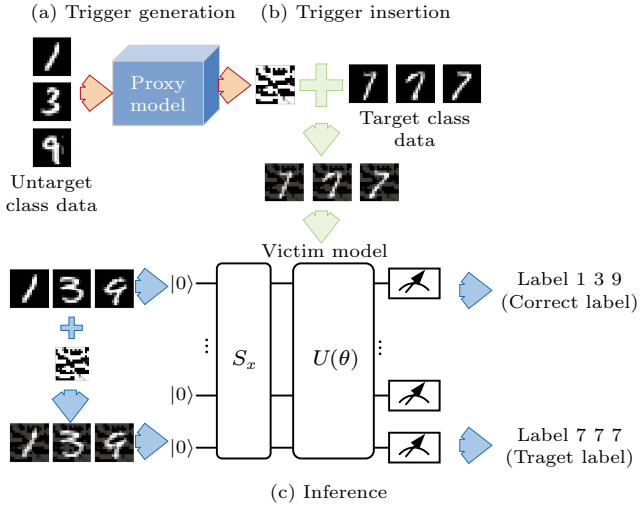


Fig. 1. Workflow of the proposed attack method.

In the following sections, we will treat the trigger generation process as an optimization problem and discuss in detail how to effectively solve this problem. Subsequently, we will further elaborate on how to embed the triggers and activate them to attack QNNs.

4.2. Problem formulation

Existing backdoor attack methods struggle to directly attack QNNs. The primary reason is that backdoor attacks typically utilize a fixed-size patch trigger, which inevitably diminishes the stealthiness of the attack due to the small input dimension of QNNs. Additionally, the patch trigger is irrelevant to the data and target class, which may result in abnormal data distribution and be detected by backdoor detection methods.[30,31] To address these challenges, we strive to find an adaptive trigger that can connect the data and target class and is relevant to specific data.

To achieve this, we aim to optimize and find the target and non-target classes in the feature space. Specifically, we assume access to the victim model Q_θ trained on the target dataset, and given the target class sample $x_\mathrm{t} \in D_\mathrm{t}$ and the non-target class sample $x_\mathrm{ut} \in D_\mathrm{ut}$, we solve the following optimization problem:

$$\delta_\mathrm{b}^* = \arg\min_{\delta \in \Delta} \sum_i Q_\theta(x_\mathrm{ut}^i + \delta) - Q_\theta(x_\mathrm{t}), \tag{3}$$

where Δ introduces a small region to ensure that the perturbation is minimal and does not fundamentally alter the input data. For a given target dataset, the above optimization will produce an imperceptible δ_b^* that associates the data in D_ut with D_t. Moreover, δ_b^* contains the most representative features in D_t, as adding it to data unrelated to D_t under Q_θ will maximize the prediction as the target class.

However, adversaries typically cannot access the parameters θ in Q_θ, and even if they understand the architecture of the victim model, they may be unable to use Q_θ to generate δ_b^* due to a lack of NISQ computing capabilities. Drawing inspiration from black-box attack techniques,[12,13,35,40] we propose using a proxy model P in place of Q_θ to generate δ_b^*.

4.3. Generate backdoor trigger

Backdoor triggers play a crucial role in backdoor attacks. As previously discussed, an effective trigger should facilitate a connection between non-target class data and target class data. To achieve this, we propose the fuzzy admix, inspired by mixup[41] and fuzzy sets,[42] to combine original non-target class data x with data selected from the target class x_t:

$$\tilde{x} = \mu(x)x + \frac{1}{n}\sum_{i=1}^{n} \nu(x)x_\mathrm{t}^i, \tag{4}$$

where n is the number of samples randomly sampled from the target class, we set it to 3 in this paper, $\mu(x)$ and $\nu(x)$ represent the membership and non-membership of x, and $\nu(x) = 1-\mu(x)$. We use the Gaussian membership function as the membership function, and its output is defined as follows:

$$\mu(x) = \mathrm{e}^{-\frac{(x-c)^2}{\sigma^2}}, \tag{5}$$

where c and σ are the mean and variance in the Gaussian membership function. In this way, the mixed data contains information from both the target class and the non-target class.

Fuzzy admix differs from mixup in that fuzzy admix combines the main part x with a small portion of x_t while maintaining the label of x, whereas mixup combines the labels of the two. As shown in Fig. 2(a), x represents the input data of the non-target class, x_t represents the randomly sampled data of the target class, x_0 represents all points with values of 0, and $\tilde{x}$ represents the possible transformed points. The red lines and green curved lines represent all possible data regions obtained by mixup and fuzzy admix, respectively. Compared with the linear interpolation operation of mixup, fuzzy admix can generate more diverse data by controlling the nonlinear relationship between x and x_t through the membership function. Figure 2(b) shows part of the MNIST dataset[43] using fuzzy admix with x_t set to 7, and it can be seen that the mixed images will have the characteristics of the target class. The advantage of doing this is that a connection between non-target class data and target class data has already been established before generating the backdoor trigger.

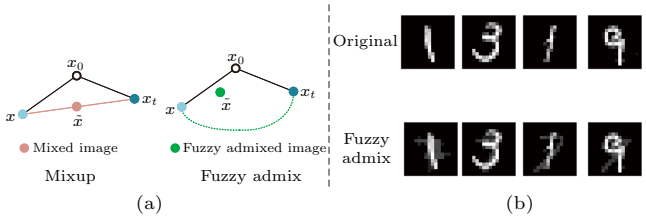


Fig. 2. The impact of fuzzy admix on input data. (a) Schematic diagram of the influence of mixup and fuzzy admix on the input space. (b) Comparison between original images and data mixed with fuzzy admix.

To tackle the optimization problem posed in Eq. (3), we require a method capable of generating perturbations of a sufficient smallness to transition non-target class inputs into the target class region. A natural idea is to consider using the adversarial perturbation generated by adversarial attack techniques to solve this optimization problem. However, current most adversarial attack techniques[12,21,44] are only able to produce distinct perturbations for individual inputs, meaning that the perturbation is unique to a specific input. This does not fulfill the needs of backdoor triggers. Therefore, we design an algorithm to find a backdoor trigger that, when added to a non-target class input, will cause the input to be misclassified by the proxy model as the target class.

Algorithm 1 provides all the details of the trigger generation process. It takes as input the proxy model P, non-target class data $\mathcal{D}_\mathrm{ut}$, target class data D_t, target class label y_t, maximal iterations I_max, fooling rate threshold T, and perturbation

algorithm G. The algorithm proceeds with two nested loops. The inner loop generates $\delta_{\rm tb}$ by mixing randomly selected x_i via fuzzy admix to get mixed data, and then employs G to obtain the perturb $\delta_{\rm tb}$. We adopt the quantum-adopted FGSM (Q-FGSM)[14] and BIM (Q-BIM)[14] methods for backdoor trigger generation. Here, $\delta_{\rm tb}$ is updated using projection to satisfy $|\delta_{\rm tb}|_\infty \leq \varepsilon$, projecting $\delta_{\rm tb}$ into the l_∞-norm and bounding each dimension to $[-\varepsilon, +\varepsilon]$. The outer loop calculates the fooling rate and determines whether generating perturbations should continue. We set the default fooling rate to 0.6. The algorithm's output is the backdoor trigger $\delta_{\rm tb}$.

As for the architecture of the proxy model, it depends on the adversary's knowledge of the victim QNN. The adversary is better off using the same architecture as the victim QNN for the attack. However, when the adversary is restricted to only classical resources, they can still use a classical model to generate the trigger. This is because, after solving the optimization problem in Eq. (3), the backdoor trigger can generate features that are identical to the target class and enable cross-model attacks.

Algorithm 1: Trigger generation

Input: The proxy model P, data of non-target class $\mathcal{D}_{\rm ut}$, data of target class $\mathcal{D}_{\rm t}$, attack target class label $y_{\rm t}$, max number of iteration $I_{\max}$, threshold of fooling rate T, perturbation generation algorithm G

Output: The backdoor trigger $\delta_{\rm tb}$

Initialize $j=0,\ f=0,\ \delta_{\rm tb}=0$;
while $j \leq I_{\max}$ and $f \leq T$ **do**
  for each sample $x_i \in \mathcal{D}_{\rm ut}$ **do**
    $x_{\rm adv} \leftarrow x_i + \delta_{\rm tb}$;
    if $P(x_{\rm adv}) \neq y_{\rm t}$ **then**
      Calculate the admixed data $\tilde{x}_i$ according to equation 4;
      $\delta_{\rm i} \leftarrow G(P,\ \tilde{x}_i,\ y_{\rm t})$;
      $x_{\rm adv} \leftarrow x_i + \delta_{\rm i}$;
      if $P(x_{\rm adv}) = y_{\rm t}$ **then**
        $\delta_{\rm tb} \leftarrow {\rm Proj}(\delta_{\rm i})$
      end
    end
  end
  $f = |\mathcal{D}_{\rm ut}|^{-1} \Sigma_{x_i \in X_{\rm ut}} [P(x_i + \delta_{\rm tb}) = y_{\rm t}]$;
  $j = j+1$;
end

4.4. Trigger embedding and activation

Once the backdoor trigger is generated, we randomly select a subset of samples from the target class, applying the backdoor trigger to their input features without altering their original labels. The number of poisoned samples is determined by the poisoning ratio, which represents the percentage of poisoned samples in the target class. In other words, the ratio is the proportion between the number of poisoned samples and the total number of samples in the target class. Subsequently, we provide the victim QNN with the poisoned data, along with the remaining clean data, for training. Upon completion of the training, the backdoor will be embedded within the victim model.

During the inference phase, triggers are appended to the non-target class data to activate the backdoor. The QNN, now embedded with the trigger, will classify these non-target class samples as belonging to the target class, thereby successfully executing the backdoor attack.

5. Numerical simulation

Our evaluation mainly focuses on the following aspects: the effectiveness of our proposed method in different QNN models; the impact of various design choices in our attack, including the impact of ε, the impact of the proxy model, and the impact of fuzzy admix; and the effectiveness of current defense detection mechanisms against our attack.

5.1. Experimental setup

To assess the effectiveness of our proposed attack method, we opted for several QNN models, including three commonly general multi-layer variational classifiers (denoted as QNN1,[7] QNN2,[32] QNN3[45]), and a quantum convolutional neural network (QCNN[46]). These models have been extensively leveraged in studies probing the vulnerability of quantum machine learning systems.[12,13] To illustrate that our attack method is not restricted to a specific QNN encoding scheme, we selected QNN2, which employs angle encoding,[33] while the other QNNs used amplitude encoding.[7,12,13] Given the unique architecture of QCNN, we considered only binary classification problems for it.

We used the MNIST dataset[43] to evaluate our proposed attack method, randomly selecting a total of 1000 training samples and 1000 test samples from the digits 1, 3, 7, and 9, and selected digits 1 and 9 specifically for the QCNN experiment. Due to the limited number of qubits available in

NISQ computers, we downscaled the image size of MNIST from 28×28 pixels to 16×16 pixels and normalized them for QNNs employing amplitude encoding. For QNN2, which uses angle encoding, we applied principal component analysis to reduce the 28×28 MNIST images to 4×4 dimensions. These processing steps are similar to the previous works.[12,13,27] For simplicity and convenience, we designated digit 7 as the target class for the backdoor attack on QNN structures, and digit 9 as the target class for the QCNN.

For training the QNNs, we adopted the quantum version of cross-entropy[12,13] as the loss function and the Adam optimizer[47] to adjust the parameters in the circuit with an aim to minimize the loss function on the training set. This process was conducted with a batch size of 64, a learning rate of 0.003, and over 40 epochs. To calculate the partial derivatives of the parameters in the variational circuit, we employed the "parameter shift rule"[45,48] to obtain the required gradients. For most experiments, we set the poisoning rate at 0.4, ε at 0.2, and the parameters in Eq. (5) as $(c, \sigma) = (1, 2)$.

Regarding the proxy model used by the adversary, we assumed that the adversary possesses knowledge of the entire structure of the quantum classifier. However, in situations where the adversary lacks such knowledge or the capability to construct a QNN, they may resort to using a classical model as a proxy. In such instances, we use a simple feedforward neural network (FNN) as the proxy model, adhering to the same architecture as outlined in Ref. [14]. For quantum computing simulations, we used the Pennylane[49] framework, while classical machine learning operations were implemented using the TensorFlow[50] framework.

5.2. Comparison with baseline backdoor attack

In this paper, we choose the clean-label backdoor attack method[26] as our baseline for comparison. For the baseline attack, we experiment with $l_2 = 1.5$ perturbations in ℓ_p-bounded adversarial example attacks, as the l_2-norm has demonstrated superior performance in the baseline attack experiments. The backdoor triggers for the baseline attack involve adding a fixed square trigger of size 3×3 to the lower right corner of the image. Figure 3 showcases examples of the baseline and the poisoned sections of the images chosen for this paper.

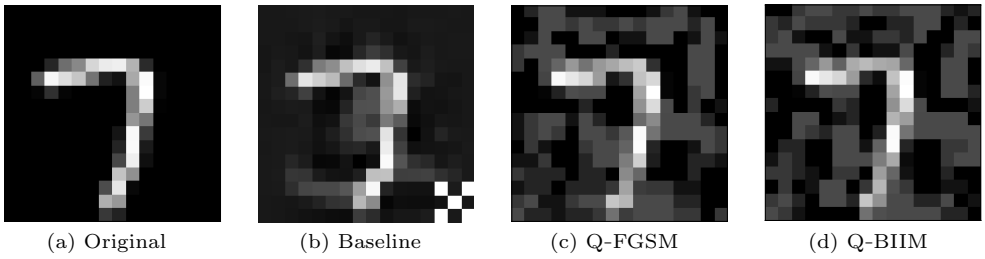


Fig. 3. Examples of different poisoned data (original images from the MNIST dataset[43] and three poisoned versions of the image using the attack strategies of the baseline and ours, from left to right).

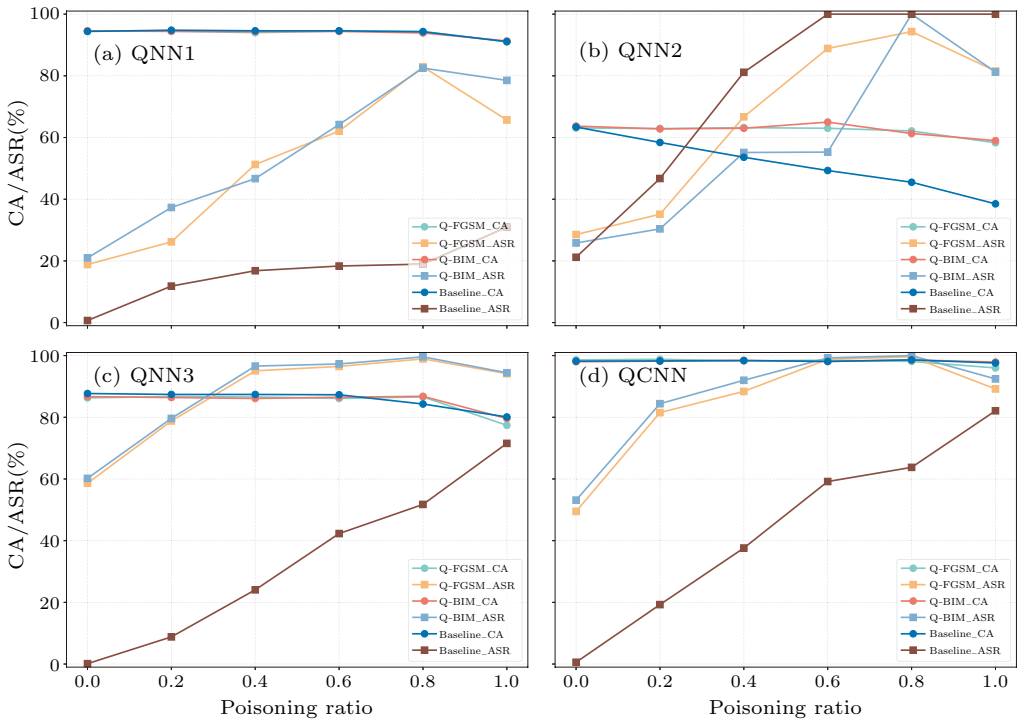


Fig. 4. Comparison of our method with the baseline in terms of CA and ASR.

As mentioned earlier, CA and ASR are common metrics employed to evaluate the performance of backdoor attacks. A successful backdoor attack should achieve a high ASR while minimizing the impact on CA. Here, we evaluated the CA and ASR of different model structures at various poisoning rates, as depicted in Fig. 4.

For QNNs that use amplitude encoding schemes, our attack's ASR is significantly higher than the baseline. However, in contrast to the baseline, our method's ASR does not increase monotonically with the poisoning rate. Instead, it initially rises, then subsequently falls. The ASR experiences a rapid surge when the poisoning rate ranges from 0 to 0.8. Yet, when this range is exceeded, the ASR begins to decline.

For QNNs that use angle encoding schemes, barring a poisoning rate of 0, the baseline attack's ASR is higher than our method's. However, as the poisoning rate increases, the CA of the baseline drastically decreases, reaching only 53.60% at a poisoning rate of 0.4, and plummeting to a mere 38.50% when the poisoning rate is 1. This level of performance degradation is unacceptable to users and could lead to the failure of the baseline attack, especially when users scrutinize the training set or when data is recollected. In contrast, our method maintains a relatively stable CA.

Except for the case in which the poisoning rate is 1, our attacks do not significantly impact the CA. However, at a poisoning rate of 1, the CA experiences a notable decrease. This observation aligns with the findings of Turner *et al.*[26] We conjecture that this decrease occurs when the poisoning rate is excessively high, leaving almost no clean data information in the target class. Consequently, QNNs fail to learn the original features of the target class, and the model tends to learn the new features of the backdoor triggers instead. Therefore, in the absence of the backdoor trigger, the model is prone to making incorrect predictions, leading to a decrease in performance.

5.3. Impact of perturbation ε

We further investigated the impact of different sizes of ε on ASR. We used Q-FGSM[14] to generate backdoor triggers and set $\varepsilon = [0.1, 0.2, 0.3, 0.4]$ to test on the clean model of QNN3 (i.e., the case in which the poisoning rate is 0) and the backdoor model to further demonstrate the effectiveness of our triggers. The experimental results are shown in Fig. 5, where ASR increases with the increase of ε. Before the backdoor embedding (i.e., clean model), our backdoor triggers achieve some ASR. After the backdoor embedding, the backdoor model can achieve higher ASR than the clean model. The ASR difference between the clean model and the backdoor model indicates that our triggers can help embed backdoors in QNNs. Additionally, CA is almost unaffected by the increase of ε.

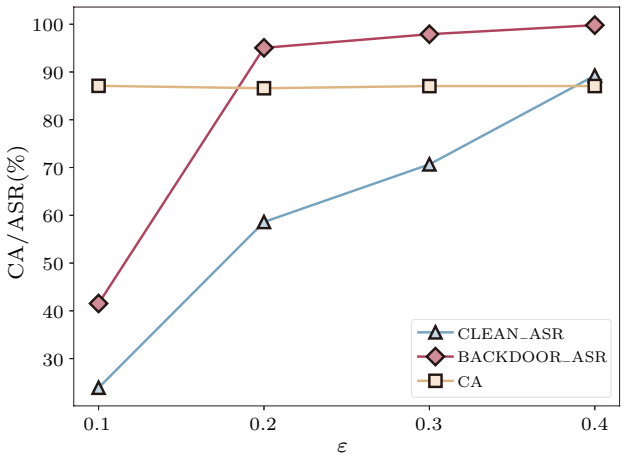


Fig. 5. The impact of different ε.

5.4. Impact of proxy model

As discussed above, when adversaries cannot use quantum resources, they may consider using classical models for attacks. Table 1 shows the ASRs of different QNN models when using FNN as the proxy model. For comparison, we also use the same classical model to generate perturbations for the baseline attack. Since the proxy model is a classical neural network structure, we use the fast gradient sign method (FGSM)[13] and the projected gradient descent (PGD)[44] to generate triggers. Our attack improves ASR to a certain extent compared to the baseline.

Table 1. ASR results using FNN as the proxy model.

Proxy model	Victim model	Perturbation generation		Baseline
		FGSM	PGD	
FNN	QNN1	87.26%	69.54%	35.69%
	QNN2	72.54%	79.43%	58.96%
	QNN3	73.49%	74.69%	32.13%
	QCNN	95.81%	95.79%	60.53%

The above evaluation results mean that adversaries can implement our proposed attacks without collecting detailed information about the victim QNN. In particular, due to the limited number of qubits in NISQ computers, users may need to reduce the dimensionality of the data for training. This preprocessing method for the data limits the ability of the baseline attack, while our method is not affected by this limitation.

5.5. Impact of fuzzy admix

We chose three sets of hyperparameters from different Gaussian membership functions and QNN3 to investigate the effect of fuzzy admix on attack performance. We selected three different values $(c, \sigma) = (0.1, 0.1), (1, 1), (1, 2)$ for experimentation. Additionally, we compared attacks that did not use fuzzy admix for data mixing. The results are listed in Table 2. In most cases, compared to attacks that did not use fuzzy admix, using fuzzy admix helps to improve ASR. Furthermore, the trigger attack performance was best when

$(c,\sigma) = (1,2)$. This indicates that fuzzy admix helps the algorithm find better backdoor triggers for attacks. However, it is difficult to observe this when choosing the set of parameters $(c,\sigma) = (0.5,0.5)$. This may be due to the information of the target class data dominating too much when selecting inappropriate hyperparameters, resulting in the trigger generation algorithm unable to obtain information from non-target class data.

Table 2. The impact of hyperparameters in fuzzy admix and without fuzzy admix.

Hyperparameters	CA	ASR
0.1,0.1	86.50%	64.20%
1,1	86.50%	92.77%
1,2	86.60%	95.06%
Without fuzzy admix	86.20%	78.63%

5.6. Evaluation for backdoor detection

Here, we evaluate whether the proposed backdoor trigger can bypass backdoor detection methods. Spectral signature detection[30] and activation clustering[31] are two popular backdoor trigger detection methods. Spectral signature detection can detect and remove poisoned data and requires a hyperparameter to be set as the upper limit for deleting suspicious data, which we set to 1.5. Activation clustering uses the relative size metric to identify whether the current model is poisoned. To avoid redundant experiments, we only report the detection results for the perturbations generated by QNN3. Table 3 lists the precision, recall, and F1 score of the two detection methods.

For the spectral signature detection method, very few poisoned samples are detected and removed, and we speculate that this is due to differences between some clean data and poisoned data. However, as shown in Fig. 4, even if some poisoned samples are removed, our attack can achieve a high attack success rate at a small poisoning rate, as empirically observed.

As for the detection results of poisoned models using activation clustering, zero indicates that the method did not find the model to be poisoned. In other words, activation clustering fails to detect the attack proposed in this paper. We speculate that this is because we did not modify the labels in the dataset, and our trigger was generated specifically for the dataset. These two factors do not lead to any anomalies in our poisoned data.

Table 3. Detection results of spectral signature detection[30] and activation clustering.[31]

Perturbation generation	Spectral signature detection			Activation clustering		
Q-FGSM	2.66%	20.44%	4.71%	0.00%	0.00%	0.00%
Q-BIM	3.21%	25.00%	5.70%	0.00%	0.00%	0.00%

6. Conclusions

This paper presents a backdoor attack strategy against QNNs, which is achieved by solving an optimization problem in the training dataset to generate triggers. Remarkably, in the most extreme cases, this approach neither requires the adversary to have knowledge of the specific architecture of the QNNs nor mandates alterations to the dataset labels. It can achieve a high success rate of backdoor attacks while having minimal impact on the performance of the quantum model. We believe that this type of attack reveals an important vulnerability in QNNs. With the development of quantum machine learning, we hope that our results will contribute to further exploration of more backdoor attack methods and encourage the development of defense models to ensure the security of future quantum machine learning systems.

Acknowledgments

We thank Jinge Yan for helpful discussion. This work was supported by the National Natural Science Foundation of China (Grant No. 62076042), the National Key Research and Development Plan of China, Key Project of Cyberspace Security Governance (Grant No. 2022YFB3103103), the Key Research and Development Project of Sichuan Province (Grant Nos. 2022YFS0571, 2021YFSY0012, 2021YFG0332, and 2020YFG0307).

References

[1] Krizhevsky A, Sutskever I and Hinton G 2017 *Commun. ACM* **60** 84
[2] Graves A, Mohamed A R and Hinton G 2013 *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 26–30, 2013, Vancouver, BC, Canada, p. 6645
[3] Dunjko V and Briegel H J 2018 *Rep. Prog. Phys.* **81** 074001
[4] Li W and Deng D L 2022 *Sci. China Phys. Mech. Astron.* **65** 220301
[5] Preskill J 2018 *Quantum* **2** 79
[6] Cerezo M, Arrasmith A, Babbush R, Benjamin S C, Endo S, Fujii K, McClean J R, Mitarai K, Cincio L and Coles P J 2021 *Natl. Rev. Phys.* **3** 625
[7] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 *Phys. Rev. A* **101** 032308
[8] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 *npj Quantum Inf.* **4** 65
[9] Dallaire-Demers P L and Killoran N 2018 *Phys. Rev. A* **98** 012324
[10] Zoufal C, Lucchi A and Woerner S 2019 *npj Quantum Inf.* **5** 103
[11] Morales M E, Tlyachev T and Biamonte J 2018 *Phys. Rev. A* **98** 062333
[12] Lu S, Duan L M and Deng D L 2020 *Phys. Rev. Res.* **2** 033212
[13] Gong W and Deng D L 2022 *Natl. Sci. Rev.* **9** 130
[14] Liu N and Wittek P 2020 *Phys. Rev. A* **101** 062331
[15] Liao H, Convy I, Huggins W J and Whaley K B 2021 *Phys. Rev. A* **103** 042427
[16] Ren W, Li W, Xu S, *et al.* 2022 *Nat. Comput. Sci.* **2** 711
[17] Weber M, Liu N, Li B, Zhang C and Zhao Z 2021 *npj Quantum Inf.* **7** 76
[18] Guan J, Fang W and Ying M 2020 arXiv:2008.07230 [quant-ph]
[19] Du Y, Hsieh M H, Liu T, Tao D and Liu N 2021 *Phys. Rev. Res.* **3** 023153

[20] Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I and Fergus R 2013 arXiv:1312.6199 [cs.CV]

[21] Goodfellow I J, Shlens J and Szegedy C 2014 arXiv:1412.6572 [stat.ML]

[22] Muñoz-González L, Biggio Battista, Demontis A, Paudice A, Wongrassamee V, Lupu E C and Roli F 2017 *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Dallas, Texas, USA, p. 27

[23] Shafahi A, Huang W R, Najibi M, Suciu O, Studer C, Dumitras T and Goldstein T 2018 *Advances in Neural Information Processing Systems*, December 2018, Montreal, Canada, p. 139

[24] Gu T, Dolan-Gavitt B and Garg S 2017 arXiv:1708.06733 [cs.CR]

[25] Chen S Y C and Yoo S 2021 *Entropy* **23** 460

[26] Turner A, Tsipras D and Madry A 2019 *arXiv:1912.02771 [stat.ML]*

[27] Chu C, Jiang L, Swany M and Chen F 2023 arXiv:2302.08090 [quant-ph]

[28] Zhao S, Ma X, Bailey J, Chen J and Jiang Y G 2020 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020, Seattle, United States, p. 14443

[29] Zhang Q, Ding Y F, Tian Y Q, Guo J M, Yuan M and Jiang Y 2021 *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, July 2021, New York, NY, USA, p. 127

[30] Tran B, Li J and Madry A 2018 *Advances in Neural Information Processing Systems*, Montreal, Quebec, Canada, p. 8011

[31] Chen B, Carvalho W, Baracaldo N, Ludwig H, Edwards B, Lee T, Molloy L and Srivastava B 2018 arXiv:1811.03728 [cs.LG]

[32] Farhi E and Neven H 2018 arXiv:1802.06002 [quant-ph]

[33] LaRose R and Coyle B 2020 *Phys. Rev. A* **102** 032420

[34] Hornik K, Stinchcombe M and White H 1989 *Neural Netw.* **2** 359

[35] Moosavi-Dezfooli S M, Fawzi A, Fawzi O and Frossard P 2017 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, Hawaii, USA, p. 86

[36] Makarov V, Bourgoin J P, Chaiwongkhot P, Gagné M, Jennewein T, Kaiser S, Kashyap R, Legré M, Minshull C and Sajeed S 2016 *Phys. Rev. A* **94** 030302

[37] Weng C H, Lee Y and Wu S H 2020 *Advances in Neural Information Processing Systems*, 2020, p. 11973

[38] Liu Y, Lee W C, Tao G, Ma S, Aafer Y and Zhang X 2019 *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, November 11–15, 2019, London, United Kingdom, p. 1265

[39] Rakin A S, He Z and Fan D 2020 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020, Seattle, United States, p. 13198

[40] Liu Y, Chen X, Liu C and Song D 2016 arXiv:1611.02770 [cs.LG]

[41] Zhang H, Cisse M, Dauphin Y N and Lopez-Paz D 2017 arXiv:1710.09412 [cs.LG]

[42] Zadeh LA 1965 *Inf. Control* **8** 338

[43] LeCun Y 1998 *THE MNIST DATABASE of handwritten digits*

[44] Madry A, Makelov A, Schmidt L, Tsipras D and Vladu A 2019 arXiv:1706.06083 [stat.ML]

[45] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 *Phys. Rev. A* **98** 032309

[46] Cong I, Choi S and Lukin M D 2019 *Nat. Phys.* **15** 1273

[47] Kingma D P and Ba J 2014 arXiv:1412.6980 [cs.LG]

[48] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 *Phys. Rev. A* **99** 032331

[49] Bergholm V, Izaac J, Schuld M, *et al.* 2018 arXiv:1811.04968 [quant-ph]

[50] Abadi M 2016 *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, September 2016, Nara, Japan, p. 1