A Numerical Gradient Inversion Attack in Variational Quantum Neural-Networks

Georgios Papadopoulos[1], Shaltiel Eloul[1], Yash Satsangi[1], Jamie Heredge[2], Niraj Kumar[2], Chun-Fu Chen[2], and Marco Pistoia[2]

Abstract—The loss landscape of Variational Quantum Neural Networks (VQNNs) is characterized by local minima that grow exponentially with increasing qubits. Because of this, it is more challenging to recover information from model gradients during training compared to classical Neural Networks (NNs). In this paper we present a numerical scheme that successfully reconstructs input training, real-world, practical data from trainable VQNNs' gradients. Our scheme is based on gradient inversion that works by combining gradients estimation with the finite difference method and adaptive low-pass filtering. The scheme is further optimized with Kalman filter to obtain efficient convergence. Our experiments show that our algorithm can invert even batch-trained data, given the VQNN model is sufficiently *'over-parameterized'*.

Index Terms—Quantum Machine Learning, VQNN, Variational, Privacy, Inversion Gradient Attacks

I. INTRODUCTION

QUANTUM Computing (QC) experiences significant technological advancements every year [1]–[6]. This progress is anticipated to influence a wide-range of scientific and industrial applications. Quantum Machine Learning (QML) is one of the domains considered to be early beneficiaries of QC, with development of algorithms such as Quantum Support Vector Machines [7] (QSVM), Quantum Deep Neural Networks [8] (QDNN), and Quantum Reinforcement Learning [9] (QRL). QML models are designed to fit patterns in data, similar to classical NNs. QMLs are generally grouped into a) models with fixed quantum circuits [10]–[12] such as quantum kernel methods, that do not have variational parameters in the quantum circuit and b) VQNN models, a flexible technique that incorporates trainable parameters within the quantum circuits (ansatz) [13], [14].

VQNNs encode classical data into quantum states and process them using quantum mechanical gates arranged in layers (Fig. 1). The output from measuring the quantum state is then used to calculate the loss function. The trainable parameters are represented as vectors and optimized by measuring gradients using the parameter shift rule [15] at each training round. The model parameters are then updated through classical optimization in the training round. VQNN models are flexible and are often combined within larger hybrid quantum-classical models [16]. This allows studying their potential applicability across various domains, including ab-initio molecular modeling [17], material analysis [18], high-energy matter [19]; and
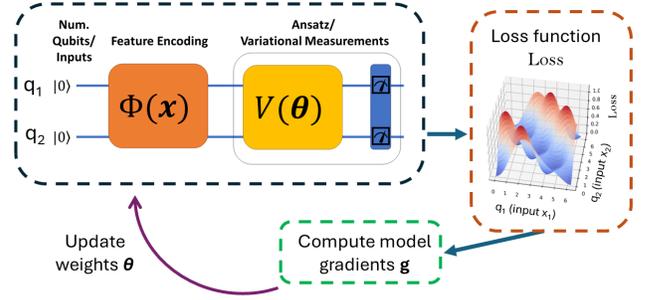
Fig. 1. Generic architecture of a VQNN. It consists of a quantum feature map $\Phi(\mathbf{x})$ that encodes classical input data $\mathbf{x}$ into a quantum state. The horizontal lines represent the qubits in the quantum circuit. The ansatz $V(\boldsymbol{\theta})$ is designed with trainable parameters $\boldsymbol{\theta}$ to explore the solution space. The trainable parameters are optimized by adjusting $\boldsymbol{\theta}$ to minimize a defined loss function (Loss).

also being explored for general data applications, recognition with ML models, financial applications, and more [20].

Training VQNNs may prove challenging since gradient-based training suffers from the vanishing gradient problem [21], [22] due to barren plateaus in the loss landscape. Another challenge is that the loss landscape is characterized by a large number of local minima. The importance of the latter is apparent even in a system with a few qubits, yet it is somewhat under-looked in the literature (but see [23]). For instance, Fig. 2 shows a sampled gradient map of a two qubits VQNN circuit. This oscillating landscape can lead to large instability in common gradient descent optimizers [23] (ADAM [24], L-BFGS [25], NatGrad [26]). Interestingly, the existence of many local minima can be leveraged for privacy-preserving training of VQNN models [27]–[29] in distributed learning setups such as *Federated Learning* (FL) [30].

Federated learning [30] are distributed training paradigms for a machine learning model, without sharing the local data among different parties/clients (C_l). FL is crucial in training on sensitive and personal data of edge devices, in communication systems such as in automotive, data-market and pricing data models [34]–[36], or recently, even being used to preserve confidentiality of drug molecule structures in an FL setup between pharmaceutical companies [37]. In a typical application, a central server initializes and distributes a global model to all clients. Each client then independently trains the model on local data, computing weight updates (see Fig. 3). Clients send only these updates back to the server. The updated weights $\boldsymbol{\Theta}_i$, shared by each client for the model

---
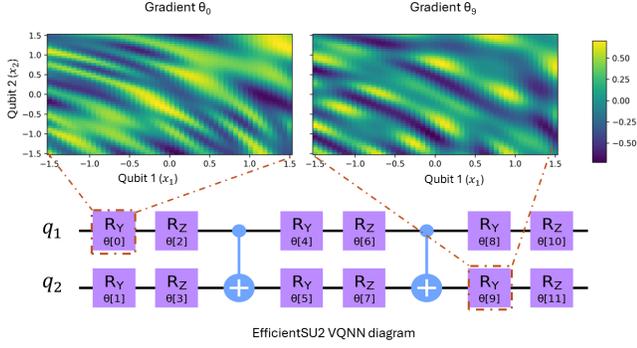
Fig. 2. The surface shows the value of the VQNN model loss gradients for each combination of input values. The ansatz VQNN model is used also in our experiments, $EfficientSU2$ [31] with Fraud data as described in Table I and with input parameters x_1 (qubit 1) and x_2 (qubit 2). $ZZFeatureMap$ is used here for embeddings [32], [33].
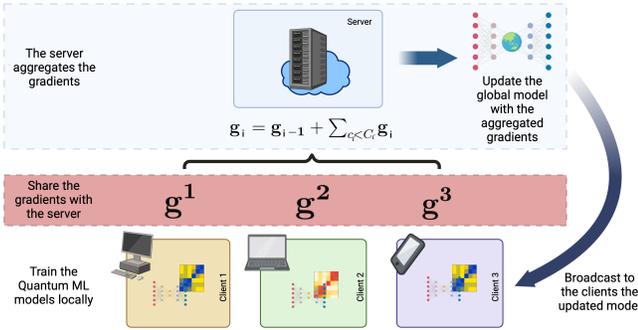
Fig. 3. In a standard pipeline of a Federated Learning model, clients (C_l) locally train the neural network on their data and share only the gradients or weights with the server, without exposing the data itself. Then the server aggregates the gradients ($\mathbf{g}$) or weights ($\boldsymbol{\theta}$) and calibrates a global model. Next, it broadcasts the updated global model back to the clients.

f (with i as the training steps/rounds), are defined as:

$$\mathbf{G}_i = -\frac{\gamma}{B} \sum_{b<B} \frac{\partial \text{Loss}(f(\mathbf{X}_{i,b}, \boldsymbol{\Theta}_i), \mathbf{Y}_{i,b})}{\partial \boldsymbol{\Theta}_i}, \qquad (1)$$

where $\mathbf{G}_i$ the gradients of each training step $i = \{1, 2, \cdots, I\}$ with I the total steps, $\mathbf{X}_i$ the features data, B the batch size, $\mathbf{Y}_i$ target values, Loss the loss function, $\boldsymbol{\Theta}_t$ the model f weights, γ the learning rate. The server then aggregates updates of the weights from clients [38], [39], $\mathbf{G}_i = \mathbf{G}_{i-1} + \sum_{c_l < C_l} \mathbf{G}_i$, and repeat the training with new rounds of data and possibly new clients. For later use, we define $\mathbf{g} = \mathbf{G}_i$ for the gradients, $\boldsymbol{\theta} = \boldsymbol{\Theta}_i$ for the model weights, $\mathbf{x} = \mathbf{X}_i$ for the data, $\mathbf{y} = \mathbf{Y}_i$ for the target values.

Gradient sharing is vulnerable to inversion attacks [40]–[42], allowing adversaries to reconstruct sensitive data from shared gradients. A typical attack involves selecting a proxy model and proxy gradients $\mathbf{g}'$ to minimize a loss function [43]–[45] between $\mathbf{g}'$ and $\mathbf{g}$, for example, $||\mathbf{g}' - \mathbf{g}||$ w.r.t a guess proxy input vector $\mathbf{x}'$ [43], [45]. Variations include using vector products instead of absolute distance [46], adding regularizations [47], or prior distributions to enhance inversion success. Results indicate that NNs are surprisingly highly vulnerable to such attack, with potentially recovering the full large batch of input data within machine-level precision [48]– [50]. Proposed mechanisms with Differential Privacy (DP) and other techniques [40], [45], [51] to reduce NN vulnerability often lower model performance, or incur high computational costs with encryption schemes [52]–[54]. Recent studies [27], [55]–[57] highlight Variational Quantum Circuits (VQCs) as resistant to gradient inversion attacks. VQC gradients, with complex multivariate Chebyshev polynomials, create a system of equations difficult to solve numerically, hindering data reconstruction. Fig. 4 illustrates the challenge of finding a global minimum for $min||\mathbf{g}' - \mathbf{g}||$.

In this paper we present a new algorithm that successfully adapts inversion attacks to invert private data when training VQNNs in federated or decenteralized learning setups. One could try inversion attacks in the VQNN scheme by numerically calculating gradients using the Finite Difference Method (FDM). However, this alone, would not be sufficient and fail to minimize $min||\mathbf{g}' - \mathbf{g}||$ to the global minimum of the original data, due to the high density of local minima. By incorporating adaptive low-pass filtering in the FDM, the attack can find the global minimum effectively by adapting the low-pass filter's window to the local minima frequency (see Fig. 5). The algorithm is found to be effective as long as the model is trainable. This paper accurately reconstructs input $\mathbf{x}$ with a numerical error of $MSE(\mathbf{x}', \mathbf{x}) \sim 1\text{e}{-}10$. Additionally, when adding a Kalman filter updates, we improve the efficiency of convergence across all trainable models and data, even recovering batch inputs.

A. Contributions

For a comprehensive privacy study, it is critical to assess the performance of the VQNN models. Therefore, this work revisits training tasks proposed on VQNNs. It is shown, that practical models which are trainable with typical optimizer (SGD or L-BFGS), can be also vulnerable to the inversion algorithm proposed here. In addition, the work includes results for benchmark dataset (MNIST [58]) and importantly, real-life sensitive data (Credit Card Fraud [59]). Through this systematic study, we provide a practical analysis of data leakage risks in sensitive QML applications. Although our work provides results on representative and typical cases, it does not intend to provide a full assessment to all potential models or datasets. Instead, it forms the base for a benchmark scheme that can be tested in a specific application or for further research on security of model training.

Specifically, our contributions are:

- **A successful gradient inversion attack on VQNN models. The attack is achieved by an optimization algorithm based on finite difference gradient estimation with an adaptive filtering.**
- **A critical study of the gradients inversion in trainable VQNN models, at different sizes and complexity, batching, and various datasets.**
- **Providing a practical benchmark attack to assess the trade-off between privacy and performance, and in comparison to NN models.**
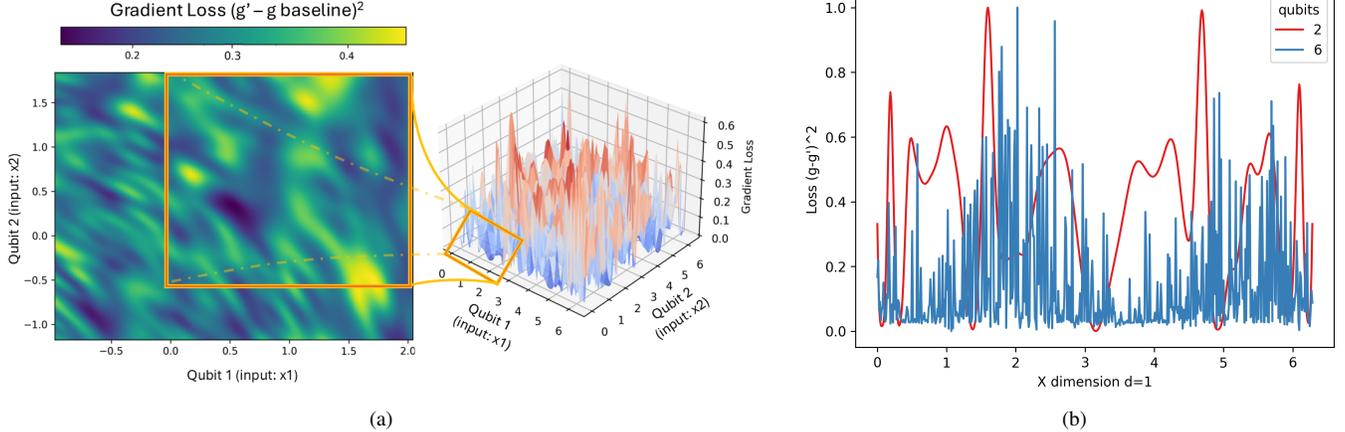
Fig. 4. The plots show the gradient loss landscape (Eq.7) with multiple local minima, which can hinder the success of gradient inversion attacks [27]–[29]. Each plot shows a different viewpoint of the multiple local minima of a VQNN model. In the 3-D and surface plots (a), the x-axis and y-axis represent the input parameters x_1 (qubit 1) and x_2 (qubit 2), ranging from 0 to 2π. The z-axis depicts the gradient loss $(\mathbf{g}' - \mathbf{g})^2$. The line plot (b) shows the multiple local minima across the dimension x_1. The VQNN model used in these plots is named 'Complex' and depicted in Table I.
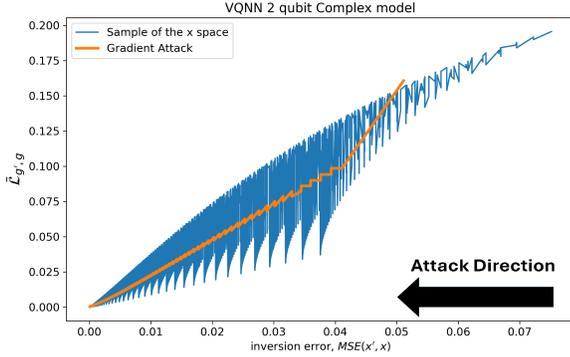
Fig. 5. Gradient inversion attack profile on two qubits 'Complex' VQNN model (thick orange line), shows a close-up near the global minimum of the inversion error with $log10$ MSE distance $(\mathbf{x}', \mathbf{x})$ versus the squared gradient loss $(\mathbf{g}' - \mathbf{g})^2$. As the attack is progressed, $\mathbf{g}'$ approaches $\mathbf{g}$ which leads to reveal the hidden $\mathbf{x}$. The fluctuating $\mathbf{x}$ space is plotted (thin blue line) by sampling small random differences of $(\mathbf{x}' - \mathbf{x})^2$.

The rest of the paper is structured as follows: Section II overviews recent work on privacy in QML followed by background on VQNN model training (Section III). Section IV, details the method for the inversion attack with the proposed optimization algorithm (Alg. 2). Section V discusses the VQNN models used, their training and setup (Table I), assumptions, and the experiments carried out. Then, we discuss results of inversion attacks in the various experiments of models, datasets, and batch. We also compare performance versus privacy, with differential privacy using noisy gradients in VQNN and NN (Section VI).

II. RELATED WORK

Many recent works explored the applicability of quantum FL setup and indicated the added value for privacy [27], [55]–[57], [60], [61]. However, none of these works demonstrate a successful attack for practical cases, and hence our paper provides an important insight to practitioners and researchers in the field. [62] uses a hybrid quantum convolutional network to preserve privacy via FL for automatic speech recognition. [63] focuses on implementing privacy-preserving QML through the application of DP. The authors propose a hybrid quantum-classical model using a VQC trained with differentially private optimization algorithms. The privacy mechanism is introduced by clipping the gradients and adding Gaussian noise to ensure DP. Compared to their work, our paper goes beyond simple noise addition for privacy by actively exploring vulnerabilities through gradient-based attacks, specifically gradient inversion. Other works in quantum DP [64] which primarily focus a theoretical framework for quantum DP using information-theoretic tools like the hockey-stick divergence. Their work does not actively explore attacks on quantum models to evaluate privacy risks. [65] employ quantum FL, which distributes training data among multiple clients and trains local models iteratively. They also integrate quantum DP by adding noise during training. However, they do not explicitly perform adversarial or propose inversion attacks to reconstruct original inputs. Their emphasis is on accuracy versus privacy under FL.

Closely related works are in a new line of research, [27], [66]. These works investigate properties of 'overparameter-ized' QML models with highly-expressive encoding (in terms of Fourier frequency spectrum) that provide an inherent protection against gradient inversion attacks. It examines these models in the context of an FL setup. The works show theoretical results backed by empirical evidence that the perpetrator encountered a considerable challenge when attempting to craft an attack on these models due to its complexity. Here, we show empirically, that despite this challenge, we can successfully invert to original data in trainable VQNN models. [61] establishes a relation between the privacy of VQNNs and the dimension of the Lie algebra of their circuit generators. This theoretical insight highlights the general susceptibility of various models to analytical closed-form solution attacks. Our research complements this study and introduces the first algorithm for inversion attacks of VQNNs. This is valuable

tool for exploring and benchmarking the resilience of VQNNs in practical applications. Several recent works also discuss the security and privacy of VQNNs [67] mostly with relation to adversarial attacks (or white-box attacks since they require access to target values) [68], [69], a group of methods that exploit gradients to create inputs (adversarial examples) that fool the model into making incorrect predictions. For example, [70] and [71] explore adversarial robustness in VQNNs. [70] analyzes vulnerabilities under scenarios involving Haar-random pure states and Gaussian latent spaces, using the quantum DP framework from [72]. They provide robustness bounds and focus on classification tasks, particularly examining prediction-change attacks and their impact on artificial datasets like MNIST [71], on the other hand, leverage quantum hypothesis testing (QHT) to establish formal robustness conditions for quantum classifiers, proposing assessments to evaluate resilience against adversarial and noise perturbations. Both works emphasize robustness evaluation not on the inversion attack type presented here, and require target labels or classification outputs.

In contrast, our work departs from these frameworks by implementing gradient-based inversion attacks that exploit quantum reversibility without relying on target labels or classification outputs. By approximating original input data in both regression and classification contexts, we provide a broader vulnerability analysis encompassing both benchmark and real-world datasets, revealing additional risks not covered in prior studies. Similar to [60], [70], [71], [73] investigate adversarial robustness in VQNNs by incorporating quantum noise (similar to DP), proposing it as a defense mechanism against adversarial attacks. Their work employs the iterative fast gradient sign method (I-FGSM) to craft adversarial examples, which rely on access to target labels for calculating perturbations that maximize the model's loss. This approach evaluates robustness by measuring the rate of mis-classification caused by these perturbations, with a focus on assessing the effects of noise on classification accuracy. However, their methodology is inherently dependent on the availability of target values and is designed for classification tasks. In contrast, our work operates without access to target labels, employing gradient inversion techniques that exploit quantum reversibility to recover input data directly.

III. BACKGROUND

A. Training Variational Quantum Neural Networks (VQNN)

Given a dataset $\{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{I}$ that consists of $\mathbf{X}_i \in \mathbb{R}^J$ input features and corresponding target values for classification $\mathbf{Y}_i \in \{0,1\}^C$ where C is the number of classes or in regression models $\mathbf{Y}_i \in \mathbb{R}$. In a model the parameters are $\boldsymbol{\theta} = \{\theta_1, \theta_2, \cdots, \theta_m, \cdots, \theta_M\}$, with $m = [1, M]$. Subsequently, training a neural network with parameter $\boldsymbol{\theta}$ that accepts the data $\mathbf{X}_i$ as input, it requires minimizing a loss function between the prediction $\hat{\mathbf{Y}}_i = f(\mathbf{X}_i, \boldsymbol{\theta})$ and the target values $\mathbf{Y}_i$:

$$\min_{\boldsymbol{\theta}} \left[\sum_{i=1}^{I} \text{Loss}(\hat{\mathbf{Y}}_i, \mathbf{Y}_i) \right] \qquad (2)$$

Algorithm 1 Training Algorithm for VQNN
Require: Dataset $\mathbf{x}$ with rows $\{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{I}$, initial parameters $\boldsymbol{\theta}_0$, learning rate η, maximum epochs I
Ensure: Optimized parameters $\boldsymbol{\theta}^*$
1: Initialize $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}_0$
2: **for** $i = 1$ to I **do**
3:     **for** each $(\mathbf{X}_i, \mathbf{Y}_i)$ in the dataset $\mathbf{x}$ **do**
4:         Encode $\mathbf{X}_i$ into quantum state:
5:         $|\Phi(\mathbf{X}_i)\rangle \leftarrow U_\Phi(\mathbf{X}_i)|0\rangle^{\otimes q}$
6:         Apply parameterized circuit:
7:         $|\psi(\mathbf{X}_i, \boldsymbol{\theta}^*)\rangle \leftarrow V(\boldsymbol{\theta}^*)|\Phi(\mathbf{X}_i)\rangle$
8:         Measure observable:
9:         $\hat{\mathbf{Y}}_i \leftarrow \langle\psi(\mathbf{X}_i, \boldsymbol{\theta}^*)|\hat{O}|\psi(\mathbf{X}_i, \boldsymbol{\theta}^*)\rangle$
10:        Compute loss:
11:        Loss $\leftarrow (\hat{\mathbf{Y}}_i - \mathbf{Y}_i)^2$
12:        Compute gradients $\mathbf{g}$ = Eq. 6
13:        Update parameters: $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^* - \eta\mathbf{g}$
14:     **end for**
15: **end for**
16: **return** $\boldsymbol{\theta}^*$

In order to minimize the loss, the gradients (Eq.1) of the loss function with respect to the model parameters $\boldsymbol{\theta}$ are calculated. After the gradient estimation, the model parameters can be updated by optimization such as stochastic gradient descent update to complete one round of training. In VQNN, the computation of gradients is different and requires the measurement of the qubits' states as detailed below.

Alg. 1 summarizes the training of a VQNN model. As a first step, it uses *quantum feature map* $\Phi(\mathbf{x})$ to encode the classical input data $\mathbf{x}$ into a quantum state [14], [21].

$$|\Phi(\mathbf{x})\rangle = U_\Phi(\mathbf{x})|0\rangle^{\otimes q}, \qquad (3)$$

where $U_\Phi(\mathbf{x})$ is a unitary operator parameterized by $\mathbf{x}$, and q is the number of qubits. The feature map $\Phi(\mathbf{x})$ is a circuit of quantum gates that transforms the initial state $|0\rangle^{\otimes q}$ with a unitary operation U_Φ that is embedded with the information of $\mathbf{x}$. Typically, in angle encoding feature maps, the number of qubits is chosen to be equal to the number of dimensions in $\mathbf{x}$, so $J = q$, where $\mathbf{x} = \{x_1, x_2, \cdots, x_J\}$ and j the dimension index $j = \{1, 2, \cdots, J\}$. A feature map used in the models here is a rotation gate around the X-axis (R_X) [14], [21], with $R_X(x_j) = \exp\left(-i\frac{x_j}{2}\sigma_x\right)$ and σ_x is the Pauli-X matrix. The *ansatz* $V(\boldsymbol{\theta})$ is a parameterized quantum circuit that consists of sequence of quantum gates with trainable parameters $\boldsymbol{\theta}$ to create a highly expressive model [74]:

$$|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle = V(\boldsymbol{\theta})|\Phi(\mathbf{x})\rangle \qquad (4)$$

the ansatz includes rotations around the Y and Z axes and controlled-NOT (CNOT) gates to entangle qubits. To obtain a prediction, a *measurement* on the quantum state $|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle$ is performed. A pre-defined Hermitian operator is used as an *observable* $\hat{O}$ and the expected value is computed [21]:

$$\hat{y} = \langle\psi(\mathbf{x}, \boldsymbol{\theta})|\hat{O}|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle. \qquad (5)$$

A common choice of the observable is the Pauli-Z (Z) operator σ_z, but any other operator can be chosen based on the problem. In classification tasks we encode probabilities with the Pauli-Z operator to measure the state of each qubit. From Eq. 5, for regression tasks the $\hat{y}$ is a continuous value derived from the expectation value of the observable. To train a VQNN, the parameters are measured and gradients are estimated with the *parameter-shift rule* [15], [75]–[77]:

$$g_m = \frac{1}{2}\left[\hat{\mathbf{y}}\left(\boldsymbol{\theta} + \frac{\pi}{2}\hat{\mathbf{e}}_m\right) - \hat{\mathbf{y}}\left(\boldsymbol{\theta} - \frac{\pi}{2}\hat{\mathbf{e}}_m\right)\right] \qquad (6)$$

where $\hat{\mathbf{e}}_m$ is the unit vector of size M, the m-th entry equals one $[\hat{\mathbf{e}}_m]_m = 1$, and all other entries $[\hat{\mathbf{e}}_m]_{i \neq m} = 0$. After evaluation of the gradients, the model parameters can be optimized with similar methods to classical NN.

IV. METHODS

A. Gradient Inversion Attacks

The core idea behind gradient inversion attacks is to exploit the parameters shared during or after a distributed training. An adversary tries to infer the original input data by analyzing these gradients. For example, by using the 'Deep Leakage Gradient' (DLG) attack on classical NN [43]. In classical NNs, the optimization of $\mathbf{x}'$ to recover $\mathbf{x}$ relies simply on back-propagation of a differentiable model to calculate some cost function between $\mathbf{g}'$ and $\mathbf{g}$, denoted by $\mathcal{L}_{\mathbf{g}',\mathbf{g}}$, and differentiate it with respect to x'_j, $\frac{\mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x'_j}$. Then, to use an optimizer to find $\mathbf{x}'$ that minimizes the cost function [47], [78], or even invert it directly for some NN architectures [45]. In a VQNN, the gradients are obtained with a measurement, hence we adapt the finite-difference method to estimate numerically $\frac{\partial \mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x_j}$ for every j. However, the main challenge, is to optimize $\mathcal{L}_{\mathbf{g}',\mathbf{g}}$ to the global minimum. This work proposes an adaptive low-pass filter in the finite difference estimation of gradients. The cost function used here, is the euclidean distance between the gradients $\mathbf{g}'$ and $\mathbf{g}$, where $\mathbf{g}'$ is the gradient of Eq. 1 with proxy model input using $\mathbf{x}'$:

$$\mathcal{L}_{\mathbf{g}',\mathbf{g}} = \frac{1}{M}\sum_{m=1}^{M}\left(g'_m - g_m\right)^2 \qquad (7)$$

Other cost functions and regularization can also be considered [48], but are beyond the scope of the study. The inversion attack scheme has the following steps:

1. Numerical Gradient Approximation: For each dimension j, compute the numerical gradient of the cost function with respect to x_j (the j similar to Eq. 3):

$$\frac{\partial \mathcal{L}^n_{\mathbf{g}',\mathbf{g}}}{\partial x'_j} = \frac{\mathcal{L}_{\mathbf{g}',\mathbf{g}}(x'_j + nhe_j) - \mathcal{L}_{\mathbf{g}',\mathbf{g}}(x'_j - nhe_j)}{2nh} + O(nh) \qquad (8)$$

See Alg. 2, and e_j is the unit vector in the j-th direction. n the moving average window size used in the algorithm $n = [1, N]$. h the step size of the algorithm.

2. Low-pass filter: We can use a simple moving average window with size N, to remove oscillations with frequencies, $1/(Nh)$:

$$\frac{\partial \mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x'_j} = \frac{1}{N}\sum_{n=1}^{N}\frac{\partial \mathcal{L}^n_{\mathbf{g}',\mathbf{g}}}{\partial x'_j} + O(nh) \qquad (9)$$

The window size for filtering, Nh depends on the model complexity and how close $\mathcal{L}_{\mathbf{g}',\mathbf{g}}$ is to zero, the global minima. Nh is initially chosen to be sufficiently large and is dynamically adapted according to convergent criteria. Optimally, close to the global minima, reducing Nh as $\mathcal{L}_{\mathbf{g}',\mathbf{g}}|_{Nh \to 0} \to 0$.

3. Gradient Descent Update: It is sufficient to use a stochastic gradient descent where at each round, k, each dimension of $\mathbf{x}'$ is updated:

$$x'^{\,k}_j \leftarrow x'^{\,k-1}_j - lr\frac{\partial \mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x'_j} \qquad (10)$$

where lr is the learning rate, obtaining the updated $\mathbf{x}'^k$ from the previous $\mathbf{x}'^{k-1}$.

4. Iterative Refinement: Repeat the gradient approximation and update steps iteratively for a fixed number of iterations (K) or until reaching convergence. Implementation of the steps are presented in Alg. 2.

B. Kalman Filter Attack

To enhance the attack's efficiency, we employ a Kalman filter, which iteratively refines the estimate of the input values. The Kalman filter operates in two parts: prediction and measurement. The prediction part uses transition $\mathbf{F}$ matrix to predict new state from previous iteration $\mathbf{x}'^{k-1}$ and initialized to zero in $k = 0$. The measurement here is calculated from the new $\mathbf{x}'^k$ in Eq. 10 to obtain the update:

$$\mathbf{x}'^k = \mathbf{F}\mathbf{x}'^{k-1} + \mathbf{D}^k(\mathbf{x}'^k - \mathbf{H}\mathbf{F}\mathbf{x}'^{k-1}) \qquad (11)$$

$\mathbf{H}$ is the measurement matrix, and $\mathbf{D}^k$ is the Kalman gain matrix for the correction of the prediction. We explain the details of the update stages for obtaining $\mathbf{D}^k$, with a standard application of Kalman filter [79] in Appendix, Eq. 12 and 13.

V. EXPERIMENTS

A. Data Generation and Quantum Model Design

We conduct model training and attack experiments on various benchmark datasets, with different VQNN architectures and qubit/input numbers, for both regression and binary classification tasks. The datasets used are MNIST [58], data generated from a commonly used cosine function with $\mathbf{y} = 0.7\cos(\mathbf{x})$, and Fraud detection data [59]. Model architectures are shown in Fig. 6, including $ZZFeatureMap$ [32], [33] and $EfficientSU2$ [31] (referred here as **ZZEfficient**). Experiments involve systems of 2 and 6 qubits and inputs. Table I displays VQNN training performance comparable to classical NNs. It includes 5-fold Cross Validation averages, with datasets split into 80/20. We apply the proposed attack in this study to shared gradients in a round of training. The

TABLE I
VQNN MODEL TRAINING PERFORMANCE AND ARCHITECTURES. THE $ZZEfficient$ MODEL IN THE EXPERIMENTS IS THE $ZZFeatureMap$ [32], [33] AND $EfficientSU2$ [31] MODELS. ALL MODELS USE A LOSS FUNCTION OF THE MSE. THE COSINE IS A REGRESSION MODEL AND THE FRAUD AND MNIST ARE BINARY CLASSIFICATION MODELS. THE TRAINING/TEST RESULTS ARE THE AVERAGE OF 5-FOLD CROSS VALIDATION.

Dataset	Model	Qubits/Inputs	Model Diagram	Metric	Train	Test	Attack Plot
Cosine	Simple	2	Fig. 6a	R^2	99%	96%	Appx. Fig. 12
	Simple	6	Fig. 6a	R^2	98%	94%	Appx. Fig. 12
	Complex	2	Fig. 6c	R^2	99%	98%	Appx. Fig. 12
	Complex	6	Fig. 6c	R^2	99%	95%	Fig. 7
MNIST	Simple	2	Fig. 6a	Balanced Acc.	89%	80%	Appx. Fig. 13
	Simple	6	Fig. 6a	Balanced Acc.	91%	81%	Appx. Fig. 13
	Complex	2	Fig. 6c	Balanced Acc.	84%	81%	Appx. Fig. 13
	Complex	6	Fig. 6c	Balanced Acc.	83%	80%	Fig. 7
Credit Card Fraud	Simple (Entangled)	2	Fig. 6b	Balanced Acc.	88%	86%	Appx. Fig. 14
	Simple (Entangled)	6	Fig. 6b	Balanced Acc.	85%	83%	Appx. Fig. 14
	Complex	2	$ZZEfficient$ (Fig. 6d)	Balanced Acc.	84%	81%	Appx. Fig. 14
	Complex	6	$ZZEfficient$ (Fig. 6d)	Balanced Acc.	80%	78%	Fig. 7

Algorithm 2 Evaluate Gradient Inversion Susceptibility in VQNNs with Kalman Filter

Require: $\mathbf{x}'$ (initial random data), N (moving average window size), h (step size), lr (learning rate), J (number of dimensions $\mathbf{x}'$), $\mathbf{g}$ (gradient of original data $\mathbf{x}$), threshold (gradient change), K the number of iterations

Ensure: Optimized $\mathbf{g}'$ approaching $\mathbf{g}$

1: $\mathbf{g} \leftarrow$ VQNN gradients of original data $\mathbf{x}$ from clients C
2: Initialize ($k = 0$) Kalman filter parameters as all-ones elements ($\mathbf{F}, \mathbf{H}$) and zero initial state $\mathbf{x}'^{k-1}$, see Appendix, Eq. 12 and 13.
3: **for** $k = 1$ to K **do**
4:     **for** $j = 1$ to J **do**
5:         Initialize $loss_+ \leftarrow 0$ and $loss_- \leftarrow 0$
6:         **for** $n = 1$ to N **do**
7:             $\mathbf{x}'_+ \leftarrow \mathbf{x}'$
8:             $x'_{+,j} \leftarrow x'_{+,j} + (n+1) \cdot h$
9:             $\mathbf{g}'_+ \leftarrow$ VQNN parameter-shift on $\mathbf{x}'_+$ (Eq. 6)
10:            $loss_+ \leftarrow loss_+ + \mathcal{L}_{\mathbf{g}'_+,\mathbf{g}}$ (Eq. 7)
11:            $\mathbf{x}'_- \leftarrow \mathbf{x}'$
12:            $x'_{-,j} \leftarrow x'_{-,j} - (n+1) \cdot h$
13:            $\mathbf{g}'_- \leftarrow$ VQNN parameter-shift on x'_- (Eq. 6)
14:            $loss_- \leftarrow loss_- + \mathcal{L}_{\mathbf{g}'_-,\mathbf{g}}$ (Eq. 7)
15:            Numerical gradient approximation (Eq. 8):
16:            $\frac{\partial \mathcal{L}^n_{\mathbf{g}',\mathbf{g}}}{\partial x'_j} \leftarrow (loss_+ - loss_-)/(2 \cdot n \cdot h)$
17:         **end for**
18:         Numerical gradient approximation (Eq. 9):
19:         $\frac{\partial \mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x'_j}$
20:         Compute average $\bar{\mathcal{L}}^k_{\mathbf{g}',\mathbf{g}}$:
21:         $\bar{\mathcal{L}}^k_{\mathbf{g}',\mathbf{g}} \leftarrow \frac{1}{N} \sum^{2N} (loss_+ + loss_-)$
22:         $x'^k_j \leftarrow x'^{k-1}_j - lr \cdot \frac{\partial \mathcal{L}_{\mathbf{g}',\mathbf{g}}}{\partial x'_j}$
23:         $x'^k_j \leftarrow$ Perform Kalman update (Eq. 11)
24:     **end for**
25:     **if** $(\bar{\mathcal{L}}^k_{\mathbf{g}',\mathbf{g}}/\bar{\mathcal{L}}^{k-1}_{\mathbf{g}',\mathbf{g}}) >$ threshold **then**
26:         $N \leftarrow \max(1, N//2)$
27:     **end if**
28: **end for**
29: **return** $\mathbf{x}'$
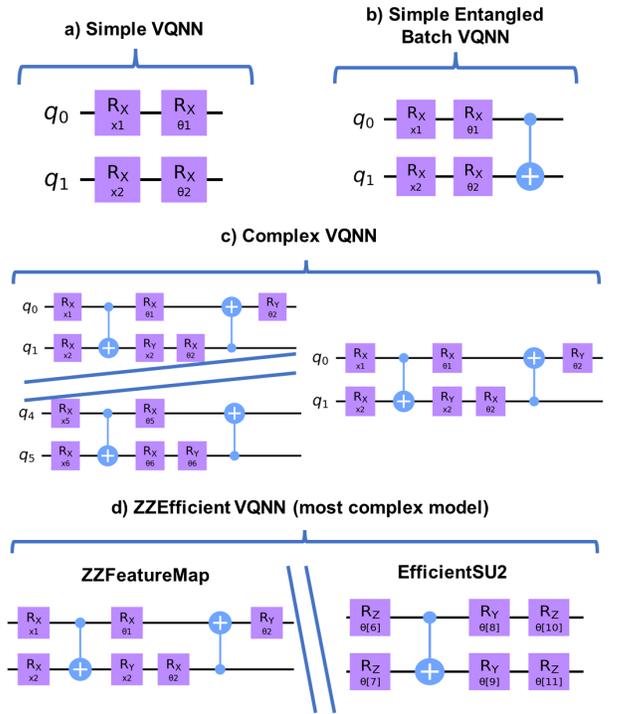
Fig. 6. The model architectures which are used in the experiments. A Simple VQNN system (a) and a Complex designed system (c) for both regression and classification for the generated Cosine and the MNIST datasets. This schematic shows a 2 and a 6-dimensional qubit/input model. System (b) is designed for batch data. For the Credit Card Fraud experiment, the trainable architecture $ZZFeatureMap$ [32], [33] and $EfficientSU2$ [31] (d) are used, named as $ZZEfficient$ in the paper.

experiments are carried out on untrained models, but complementary results for trained model are provided in Appendix Fig. 17. Qiskit [80] and Pennylane [81] libraries are both used through out the study with combination of Torch library to evaluate VQNNs model gradients using the parameter shift rule, and for training the models.

B. Inversion attack experiment

Table II provides an overview of the attack experiments. The attack algorithm in Section IV-A is used here to optimize

TABLE II
Attacks on a variety of VQNN models, with single row data and batch-size data. The results are the sample average $MSE(\mathbf{x}', \mathbf{x})$ of 10 attacks with different starting points $\mathbf{x}'$. The ($\pm SE$) is the standard error of the results. Successful attacks are considered to reach a threshold of ≤ 0.005 MSE. The model architecture is based on Fig. 6. Maximum iterations of the models are ≤ 250.

Model	One Shot Success %	Adaptive Moving Average MSE ($\pm SE$)	One Shot Success %	Kalman MSE ($\pm SE$)
VQNN 2-qubits Cosine Complex	100%	0.0002 ($\pm 7e{-}4$)	100%	0.0002 ($\pm 5e{-}4$)
VQNN 6-qubits Cosine Complex	40%	0.0005 ($\pm 4e{-}4$)	70%	0.0003 ($\pm 3e{-}4$)
VQNN 6-dims Fraud ZZEfficient	50%	0.0007($\pm 2e{-}4$)	70%	0.0003($\pm 1e{-}4$)
VQNN 6-dims MNIST Complex	40%	0.0005($\pm 1e{-}4$)	80%	0.0002($\pm 1e{-}4$)
VQNN Batch size 2	100%	0.00005 ($\pm 1e{-}5$)	100%	0.00002 ($\pm 4e{-}5$)
VQNN Batch size 3	100%	0.002 (± 0.005)	100%	0.003 (± 0.004)

random initial points $\mathbf{x}'$ with adaptive moving average and also with Kalman filter. Experiments are carried out on $J = 2$ and $J = 6$ dimensional datasets, for various different circuit architectures (Fig. 6).

C. Batch Inversion Attack

In an additional experiment, we explore a batch attack on a VQNN (Table II). In this experiment, gradients are summed over a batch of vectors, before being shared. Overall, the larger the batch is, the harder is to invert the gradients to the input vectors. In VQNN, a batch size of 2 and 3 are used. Larger batch sizes causes significant degradation of model performance due to their small model sizes. In this experiment, the model used is a Simple Entangled VQNN depicted in Fig. 6. The CNOT gate introduces entanglement between qubits, which maximizes the dependencies between the input elements in the model. This ensures an 'Over-parameterized' model.

D. Differential Privacy and NN Experiment

Experiments for benchmark NN models carried out to compare both training, and privacy against inversion attack, under differential privacy schemes involving adding noise. A Gaussian noise with standard deviation σ, is added to the shared original gradients $\mathbf{g}$. The added noise can reduce the inversion attack accuracy but at the same time degrades the model training performance [82], [83]. We evaluate the attack and model performances under various differential privacy budgets (noise magnitudes). A typical benchmark NN attack, Deep Leakage Gradients (DLG), is also implemented for this comparison from [43].

E. Assumptions

Assumptions on the setup and the success criteria of the attack are provided below:

1) Unknown input data: In an attack scenario, there is no access to the original underlying data $\mathbf{x}$ neither to the target values $\mathbf{y}$. The initialized weights for the VQNN model gradients are all unit vectors. For robustness, attacks are performed also successfully with random initialized weights (see also Appendix, Fig.16).

2) Inversion evaluation: Inversion success is measured by using Eq. 7, the Euclidean distance between the guessed $\mathbf{x}'$ and the real $\mathbf{x}$, $MSE(\mathbf{x}', \mathbf{x})$. There are other metrics to evaluate attacks [84], but they are concerned with different

types of datasets and embeddings. To evaluate if the algorithm successfully recovers the data, it is sufficient to use a finite and small number of iterations (up to 200) are used in a single-shot trials, and a sample size of 10, to obtain the success rate of the trials.

3) Convergent criteria and a successful attack: The attack is considered successful if it meets the threshold $MSE(\mathbf{x}', \mathbf{x}) \leq 0.005$. To support that this threshold is sufficient to recover the vector $\mathbf{x}$ (at the global minima) with high precision. A convergence analysis of longer runs is also provided in Appendix. Fig. 15 . This supporting data shows recovery of all sample with error down to $MSE(\mathbf{x}', \mathbf{x}) \sim 1e{-}10$ for initial vectors $\mathbf{x}'$ constrained to ≥ 0.005 random distances.

4) Over-parameterized Models: The inversion is possible as long as the model (the number of equations can be constructed from $\boldsymbol{\theta}$) is comparable or larger than the number of unknowns ($\mathbf{x}$, $\mathbf{y}$) [85]. In NN, this is usually the case, as models generally contain a large number of parameters. VQNN models are small, and using batch size larger than one is sometimes sufficient to obtain an 'under-parameterized' model. In fact, in a batch size larger than one, it requires to use more complex models in order to remain 'over-parameterized'. Note, that increasing batch sizes as a strategy to prevent the attack is not sufficient, as it can cause also to large performance degradation. Specifically, in the model used here (Simple Entangled VQNN), increasing the batch size beyond 3 inputs vectors already affects performance dramatically.

VI. RESULTS AND ANALYSIS

A. Attack Results

Fig. 7 shows successful convergence profiles of the inversion attacks on various datasets for Complex and $ZZEfficient$ models. Averaged error statistics are in Table II. Using the scheme from Alg. 2 (without the Kalman component), the adaptive low-pass filter attack can recover all sensitive input data $\mathbf{x}$ in small number of iterations, often within a single trial (except 6 qubits at 40%). Results in Table II reports that adding Kalman improves further the success and convergence rates (see also Appendix Figs. 12, 13, 14). Increasing the number of iterations would reduce error to $1e{-}10$ (see convergence analysis in Appendix Fig. 15). In Fig. 8, we assess the attack across different circuit depths, showing model complexity's impact. The attack recovers original data in all cases, though convergence takes longer due to increased
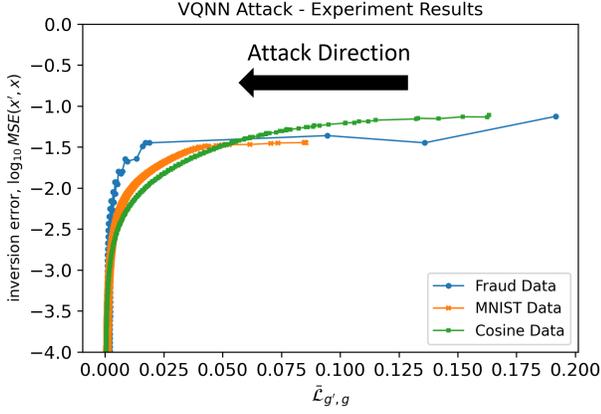
Fig. 7. 6-dimensional Complex VQNN model (Fig. 6 and $ZZEfficient$) on a variety of datasets. Each line provides a visual of the optimization path (attacks) taken to move the random starting point $\mathbf{x}'$ (start of the line on the right) closer to the baseline/true value $\mathbf{x}$. The y-axis is the log MSE distance between the $\mathbf{x}'_k$ and the baseline $\mathbf{x}$, with K the number of iterations for the attack. The x-axis is the loss function between gradient $\mathbf{g}'$ and original gradient $\mathbf{g}$ (Eq. 7).
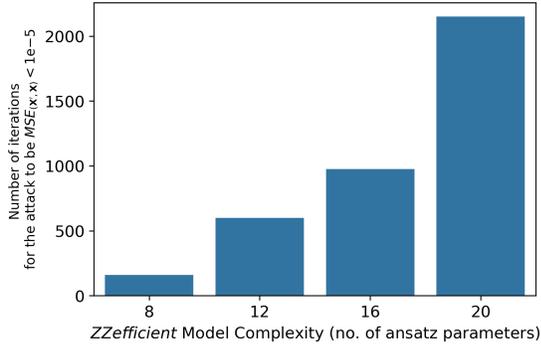
Fig. 8. The plot shows the number of iterations required for the attack to reach error of $1e{-}5$ for a $ZZEfficient$ model of 2 dimensions/qubits of different complexities (8, 12, 16, and 20 parameters ansatz). The results show the average iteration number for Cosine and Fraud datasets together.

ansatz complexity. More gates lead to longer optimization but remain fully recoverable as expected.

In addition, Fig. 9 shows the increase of efficiency when adding Kalman Filter to update the step in the attack. With Kalman filter updates we can successfully recover $\mathbf{x}$ with small starting filter sizes, e.g. $N = 16$, which improves significantly the convergent rate. It shows that for $N = 16$ the non-Kalman attack with adaptive moving average fails to converge at all, but is still successful when using sufficiently large filters, e.g. $N = 64$.

B. Batch Data

Since in many ML and FL pipelines the data that the model is trained on a batch of inputs, and subsequently the gradients are a sum of these batches, the model quickly becomes 'under-parameterized' in comparison to the size of unknown inputs. Table II shows that using a batch size of 2, the attack algorithm approximates $\mathbf{x}$ as close as $MSE(\mathbf{x}', \mathbf{x}) = 2e{-}5$. When batch size increases to 3 though, the accuracy decreases to $MSE(\mathbf{x}', \mathbf{x}) = 0.003$. Due to the small size of VQNN in
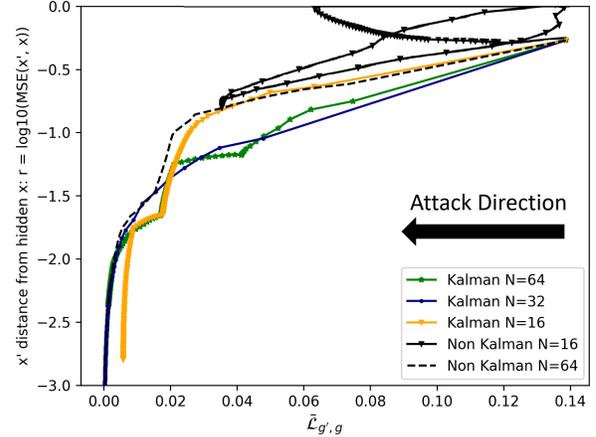
Fig. 9. VQNN complex model $ZZEfficient$ with 2 qubits/inputs $\mathbf{x}$. The graph shows the Kalman based attacks on Fraud data with different number of perturbations ($N = 16, 32, 64$) for the adaptive low-pass filtering (Alg. 2). The figure also includes a failed non-Kalman attack for $N = 16$ and a successful non-Kalman attack for $N = 64$.

comparison to classical NNs, it is enough to use a batch of size above 3 to not be able to recover the gradients in the VQNN model used. However, as long as the model has sufficient complexity it would possible to recover the data as shown in Table II. Naturally, increasing the batch size in such small models would also lead to reduced training performance, hence it is not considered as a sufficient defense against this attack. For example, for a Simple Entangled VQNN model trained on the Cosine dataset and batch size 5, the test accuracy is 93%; with batch size 20 the test accuracy decreases to 88%; with batch size 60 the accuracy is down to 71%; and batch size 100 the accuracy is 66%.

C. Comparison to NN and Differential Privacy

Table III summarizes model performance and attack error for NN and VQNN on Fraud and Cosine datasets. The DLG attack is successful for NN but fails for VQNN, hence the results are compared to our algorithm which can successfully invert the data. We also examine the model performance and the attack performance when noise is added to increase privacy and potentially prevent the attack.

Privacy versus performance results are also provided for Cosine (Fig. 10) and Fraud datasets (Fig. 11). VQNNs excel in fitting cosine functions due to their quantum gate design, outperforming NNs with few linear layers. Fig. 10 shows VQNNs offer better security against inversion with noise, while NNs remain vulnerable even with degraded performance. The inversion error's square root aligns with added noise, as shown in [45]. However for more realistic datasets, such as the Fraud dataset, VQNNs struggle to match simple NNs with 16 weights. Fig. 11 indicates VQNNs may be more secure but with lower performance. Notably, noise with standard deviation $\sigma = 0.1$ yields significant inversion error ($MSE(\mathbf{x}', \mathbf{x}) \sim \pm 0.013$) while maintaining model utility ($R^2 = 0.65$) for VQNN. NNs perform better than VQNN but

TABLE III

TRAIN/TEST MODEL ACCURACY OF A 5-FOLD CROSS VALIDATION SET, WITH THE STANDARD ERRORS ($\pm SE$) OF THE RESULTS. THE ATTACKS ON THE MODELS ARE PERFORMED ONCE. THE VQNN MODELS WITH FRAUD DATA IS THE $ZZEfficient$ AND FOR THE COSINE DATA IS THE 'COMPLEX ENTANGLED' (FIG. 6). A TYPICAL ATTACK FOR NN, DEEP LEAKAGE GRADIENTS (DLG) [43], IS ALSO INCLUDED HERE FOR NN.

Models	Train accuracy ($\pm SE$)	Test accuracy ($\pm SE$)	DLG (mse, iters)	Kalman (mse, iters)
Fraud 6 input 16weights/ NN	91% ($\pm 0.8\%$)	88% ($\pm 1.0\%$)	<0.001, 10	<0.001, 200
Fraud 6 inputs 16weights/ NN + Noise0.05	90% ($\pm 0.8\%$)	88% ($\pm 0.9\%$)	<0.002, 10	<0.004, 200
Fraud 6 inputs 16weights/ NN + Noise0.14	80% ($\pm 1.2\%$)	79% ($\pm 1.5\%$)	<0.06, 12	<0.017, 200
Fraud 6 input/ VQNN	80% ($\pm 2.0\%$)	78% ($\pm 2.3\%$)	-	<0.003, 200
CosinBenchmark6 VQNN	99% ($\pm 0.2\%$)	95% ($\pm 0.3\%$)	-	<0.004, 200
CosinBenchmark2 VQNN + Noise0.08	88% ($\pm 0.3\%$)	86% ($\pm 0.4\%$)	-	<0.007, 200
CosinBenchmark2 VQNN	99% ($\pm 0.2\%$)	98% ($\pm 0.5\%$)	-	<0.001, 200
CosinBenchmark NN16x2	97% ($\pm 0.2\%$)	96% ($\pm 0.4\%$)	<0.002, 10	<0.001, 200
CosinBenchmark NN16x2 + Noise0.08	87% ($\pm 0.3\%$)	85% ($\pm 0.4\%$)	<0.01, 10	<0.003, 200
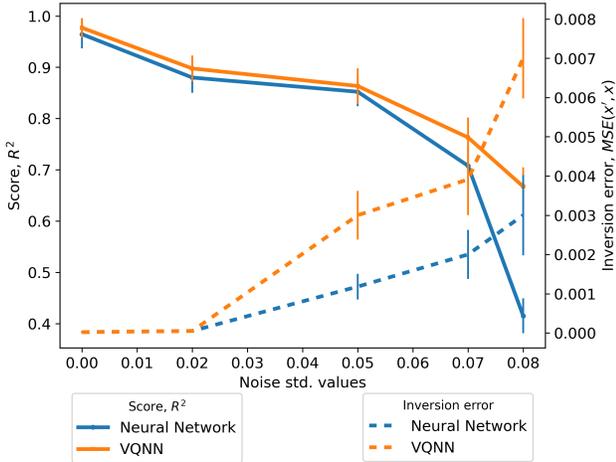
Fig. 10. The plot shows the R^2 regression score (LH y-axis and solid lines) and the inversion error between $\mathbf{x}'$ and $\mathbf{x}$ (RH y-axis and dashed lines) over different noise standard deviation (σ) values ($\sigma = 0.0, 0.02, 0.05, 0.07, 0.08$) added to gradients shared. It follows the Cosine data results from Table III. The models are the Cosine Benchmark VQNN 2 qubits and the Cosine Benchmark NN 2 dimensions.
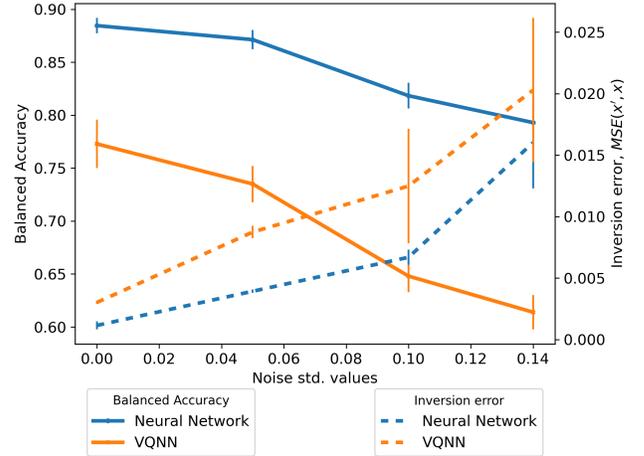
Fig. 11. The plot shows the balanced accuracy (LH y-axis and solid lines) and the inversion error between $\mathbf{x}'$ and $\mathbf{x}$ (RH y-axis and dashed lines) over the different noise levels ($\sigma = 0.0, 0.05, 0.10, 0.14$) added to the gradients shared. It follows the Fraud data results from Table III. The model is the 6 dimensional $ZZEfficient$.

with smaller noise inversion error. With that, other considerations may included in practice. For example, it is worthwhile to note that noise is inherent in quantum hardware, often reduced in VQNN gradients [86], but can be leveraged here for privacy. This suggests that advanced low-error quantum computers may not be essential, as noisy systems could benefit privacy.

VII. CONCLUSION

This paper introduces a numerical gradient attack scheme that effectively navigates the frequency modulated landscape of VQNN to obtain global minima via adaptive low-pass filtering. The algorithm proposed can be further optimized with Kalman filter updates to obtain efficient convergence. We show empirically, that as long as the model is trainable (numerically), the proposed attack can invert successfully gradients to data within the precision of the numerical scheme, error down to $\sim 1e-10$. Analysis of various models and datasets show that the algorithm can be used as a benchmark scheme to evaluate the tradeoff between privacy and performance in distributed learning setup, and the practicality of VQNN in comparison to classic NN with differential privacy. Finally,

while this work explores optimization for the inversion of gradients, this optimization approach with low-pass filtering could be insightful for future studies in improving training and the usability of VQNN models.

DISCLAIMER

References

[1] S. Bravyi, O. Dial, J. M. Gambetta, D. Gil, and Z. Nazario, "The future of quantum computing with superconducting qubits," *Journal of Applied Physics*, vol. 132, no. 16, 10 2022. [Online]. Available: http://dx.doi.org/10.1063/5.0082975

[2] J. W. Z. Lau, K. H. Lim, H. Shrotriya, and L. C. Kwek, "Nisq computing: where are we and where do we go?" *AAPPS Bulletin*, vol. 32, no. 1, 9 2022. [Online]. Available: http://dx.doi.org/10.1007/s43673-022-00058-z

[3] G. Kumar, S. Yadav, A. Mukherjee, V. Hassija, and M. Guizani, "Recent advances in quantum computing for drug discovery and development," *IEEE Access*, vol. 12, pp. 64 491–64 509, 2024.

[4] S. M. L. Pfaendler, K. Konson, and F. Greinert, "Advancements in quantum computing—viewpoint: Building adoption and competency in industry," *Datenbank-Spektrum*, vol. 24, no. 1, p. 5–20, 3 2024. [Online]. Available: http://dx.doi.org/10.1007/s13222-024-00467-4

[5] M. Aghaee, A. Alcaraz Ramirez, Z. Alam, R. Ali, M. Andrzejczuk, A. Antipov, M. Astafev, A. Barzegar, B. Bauer, J. Becker, U. K. Bhaskar, A. Bocharov, S. Boddapati, D. Bohn, J. Bommer, L. Bourdet, A. Bousquet, S. Boutin, L. Casparis, B. J. Chapman, S. Chatoor, A. W. Christensen, C. Chua, P. Codd, W. Cole, P. Cooper, F. Corsetti, A. Cui, P. Dalpasso, J. P. Dehollain, G. de Lange, M. de Moor, A. Ekefjärd, T. El Dandachi, J. C. Estrada Saldaña, S. Fallahi, L. Galletti, G. Gardner, D. Govender, F. Griggio, R. Grigoryan, S. Grijalva, S. Gronin, J. Gukelberger, M. Hamdast, F. Hamze, E. B. Hansen, S. Heedt, Z. Heidarnia, J. Herranz Zamorano, S. Ho, L. Holgaard, J. Hornibrook, J. Indrapiromkul, H. Ingerslev, L. Ivancevic, T. Jensen, J. Jhoja, J. Jones, K. V. Kalashnikov, R. Kallaher, R. Kalra, F. Karimi, T. Karzig, E. King, M. E. Kloster, C. Knapp, D. Kocon, J. V. Koski, P. Kostamo, M. Kumar, T. Laeven, T. Larsen, J. Lee, K. Lee, G. Leum, K. Li, T. Lindemann, M. Looij, J. Love, M. Lucas, R. Lutchyn, M. H. Madsen, N. Madulid, A. Malmros, M. Manfra, D. Mantri, S. B. Markussen, E. Martinez, M. Mattila, R. McNeil, A. B. Mei, R. V. Mishmash, G. Mohandas, C. Mollgaard, T. Morgan, G. Moussa, C. Nayak, J. H. Nielsen, J. M. Nielsen, W. H. P. Nielsen, B. Nijholt, M. Nystrom, E. O'Farrell, T. Ohki, K. Otani, B. Paquelet Wütz, S. Pauka, K. Petersson, L. Petit, D. Pikulin, G. Prawiroatmodjo, F. Preiss, E. Puchol Morejon, M. Rajpalke, C. Ranta, K. Rasmussen, D. Razmadze, O. Reentila, D. J. Reilly, Y. Ren, K. Reneris, R. Rouse, I. Sadovskyy, L. Sainiemi, I. Sanlorenzo, E. Schmidgall, C. Sfiligoj, M. B. Shah, K. Simoes, S. Singh, S. Sinha, T. Soerensen, P. Sohr, T. Stankevic, L. Stek, E. Stuppard, H. Suominen, J. Suter, S. Teicher, N. Thiyagarajah, R. Tholapi, M. Thomas, E. Toomey, J. Tracy, M. Turley, S. Upadhyay, I. Urban, K. Van Hoogdalem, D. J. Van Woerkom, D. V. Viazmitinov, D. Vogel, J. Watson, A. Webster, J. Weston, G. W. Winkler, D. Xu, C. K. Yang, E. Yucelen, R. Zeisel, G. Zheng, and J. Zilke, "Interferometric single-shot parity measurement in inas–al hybrid devices," *Nature*, vol. 638, no. 8051, p. 651–655, 2 2025. [Online]. Available: http://dx.doi.org/10.1038/s41586-024-08445-2

[6] M. Liu, R. Shaydulin, P. Niroula, M. DeCross, S.-H. Hung, W. Y. Kon, E. Cervero-Martín, K. Chakraborty, O. Amer, S. Aaronson, A. Acharya, Y. Alexeev, K. J. Berg, S. Chakrabarti, F. J. Curchod, J. M. Dreiling, N. Erickson, C. Foltz, M. Foss-Feig, D. Hayes, T. S. Humble, N. Kumar, J. Larson, D. Lykov, M. Mills, S. A. Moses, B. Neyenhuis, S. Eloul, P. Siegfried, J. Walker, C. Lim, and M. Pistoia, "Certified randomness using a trapped-ion quantum processor," *Nature*, 3 2025. [Online]. Available: http://dx.doi.org/10.1038/s41586-025-08737-1

[7] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Phys. Rev. Lett.*, vol. 113, p. 130503, 9 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.113.130503

[8] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, "Training deep quantum neural networks," *Nature Communications*, vol. 11, no. 1, 2 2020. [Online]. Available: http://dx.doi.org/10.1038/s41467-020-14454-2

[9] V. Saggio, B. E. Asenbeck, A. Hamann, T. Strömberg, P. Schiansky, V. Dunjko, N. Friis, N. C. Harris, M. Hochberg, D. Englund, S. Wölk, H. J. Briegel, and P. Walther, "Experimental quantum speed-up in reinforcement learning agents," *Nature*, vol. 591, no. 7849, p. 229–233, 3 2021. [Online]. Available: http://dx.doi.org/10.1038/s41586-021-03242-7

[10] K. M. Barnes and M. B. Gale, "Meta-genetic programming for static quantum circuits," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '19. ACM, 7 2019, p. 2016–2019. [Online]. Available: http://dx.doi.org/10.1145/3319619.3326907

[11] F. Chen, Q. Zhao, L. Feng, C. Chen, Y. Lin, and J. Lin, "Quantum mixed-state self-attention network," 2024. [Online]. Available: https://arxiv.org/abs/2403.02871

[12] M. Schuld, "Supervised quantum machine learning models are kernel methods," 2021. [Online]. Available: https://arxiv.org/abs/2101.11020

[13] Y. Du, T. Huang, S. You, M.-H. Hsieh, and D. Tao, "Quantum circuit architecture search for variational quantum algorithms," *npj Quantum Information*, vol. 8, no. 1, 5 2022. [Online]. Available: http://dx.doi.org/10.1038/s41534-022-00570-y

[14] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, p. 625–644, 8 2021. [Online]. Available: http://dx.doi.org/10.1038/s42254-021-00348-9

[15] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," *Phys. Rev. A (Coll. Park.)*, vol. 99, no. 3, 3 2019.

[16] L. Bischof, S. Teodoropol, R. M. Fuchslin, and K. Stockinger, "Hybrid quantum neural networks show strongly reduced need for free parameters in entity matching," *Sci Rep*, vol. 15, no. 4318, 2025. [Online]. Available: https://doi.org/10.1038/s41598-025-88177-z

[17] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, "Challenges and opportunities in quantum machine learning," *Nature Computational Science*, vol. 2, no. 9, p. 567–576, 9 2022. [Online]. Available: http://dx.doi.org/10.1038/s43588-022-00311-3

[18] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, and J.-R. Vlimant, "Quantum machine learning in high energy physics," *Machine Learning: Science and Technology*, vol. 2, no. 1, p. 011003, 3 2021. [Online]. Available: https://dx.doi.org/10.1088/2632-2153/abc17d

[19] H. Hirai, "Practical application of quantum neural network to materials informatics," *Scientific Reports*, vol. 14, no. 1, 4 2024. [Online]. Available: http://dx.doi.org/10.1038/s41598-024-59276-0

[20] M. Sajjan, J. Li, R. Selvarajan, S. H. Sureshbabu, S. S. Kale, R. Gupta, V. Singh, and S. Kais, "Quantum machine learning for chemistry and physics," *Chem. Soc. Rev.*, vol. 51, pp. 6475–6573, 2022. [Online]. Available: http://dx.doi.org/10.1039/D2CS00203E

[21] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, no. 1, 11 2018. [Online]. Available: http://dx.doi.org/10.1038/s41467-018-07090-4

[22] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, and M. Cerezo, "Barren plateaus in variational quantum computing," *Nature Reviews Physics*, 3 2025. [Online]. Available: http://dx.doi.org/10.1038/s42254-025-00813-9

[23] D. Wierichs, C. Gogolin, and M. Kastoryano, "Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer," *Phys. Rev. Res.*, vol. 2, p. 043246, 11 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.2.043246

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. [Online]. Available: https://arxiv.org/abs/1412.6980

[25] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Math. Program.*, vol. 45, no. 1–3, p. 503–528, 8 1989.

[26] S.-i. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, 1998.

[27] N. Kumar, J. Heredge, C. Li, S. Eloul, S. H. Sureshbabu, and M. Pistoia, "Expressive variational quantum circuits provide inherent privacy in federated learning," 2023. [Online]. Available: https://arxiv.org/abs/2309.13002

[28] W. Gong, D. Yuan, W. Li, and D.-L. Deng, "Enhancing quantum adversarial robustness by randomized encodings," *Phys. Rev. Res.*, vol. 6, p. 023020, 4 2024. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.6.023020

[29] Y. Zhao, H. Zhong, X. Zhang, Y. Li, C. Zhang, and M. Pan, "Bridging quantum computing and differential privacy: Insights into quantum computing privacy," in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 01, 2024, pp. 13–24.

[30] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 4 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[31] C.-Y. Liu, E.-J. Kuo, C.-H. A. Lin, S. Chen, J. G. Young, Y.-J. Chang, and M.-H. Hsieh, "Training classical neural networks by quantum machine learning," 2024. [Online]. Available: https://arxiv.org/abs/2402.16465

[32] B.-S. Chen and J.-L. Chern, "Generating quantum feature maps for svm classifier," 2022. [Online]. Available: https://arxiv.org/abs/2207.11449

[33] D. Alvarez-Estevez, "Benchmarking quantum machine learning kernel training for classification tasks," 2024. [Online]. Available: https://arxiv.org/abs/2408.10274

[34] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, "Opportunities of federated learning in connected, cooperative, and automated industrial systems," *IEEE Communications Magazine*, vol. 59, no. 2, pp. 16–21, 2021.

[35] H. Zhang, J. Bosch, and H. H. Olsson, "Real-time end-to-end federated learning: An automotive case study," in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2021, pp. 459–468.

[36] P. Sun, L. Wu, Z. Wang, J. Liu, J. Luo, and W. Jin, "A profit-maximizing data marketplace with differentially private federated learning under price competition," *Proc. ACM Manag. Data*, vol. 2, no. 4, 9 2024. [Online]. Available: https://doi.org/10.1145/3677127

[37] E. Callaway, "Alphafold is running out of data - so drug firms are building their own version," 3 2025. [Online]. Available: https://www.nature.com/articles/d41586-025-00868-9?secureweb=Teams

[38] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3347–3366, 2023.

[39] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, "Model aggregation techniques in federated learning: A comprehensive survey," *Future Generation Computer Systems*, vol. 150, pp. 272–293, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X23003333

[40] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020. [Online]. Available: https://arxiv.org/abs/2003.02133

[41] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. Vincent Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[42] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20329848

[43] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," 2019. [Online]. Available: https://arxiv.org/abs/1906.08935

[44] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" *ArXiv*, vol. abs/2003.14053, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:214728347

[45] S. Eloul, F. Silavong, S. Kamthe, A. Georgiadis, and S. J. Moran, "Mixing gradients in neural networks as a strategy to enhance privacy in federated learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1 2024, pp. 3956–3965.

[46] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[47] H. Yin, A. Mallya, A. Vahdat, J. M. Álvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16 332–16 341, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:233241017

[48] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song, "The secret revealer: Generative model-inversion attacks against deep neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2020.

[49] N. Subbanna, M. Wilms, A. Tuladhar, and N. D. Forkert, "An analysis of the vulnerability of two common deep learning-based medical image segmentation techniques to model inversion attacks," *Sensors*, vol. 21, no. 11, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/11/3874

[50] Z. Zhang, Q. Liu, Z. Huang, H. Wang, C.-K. Lee, and E. Chen, "Model inversion attacks against graph neural networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 9, pp. 8729–8741, 2023.

[51] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 1322–1333. [Online]. Available: https://doi.org/10.1145/2810103.2813677

[52] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: Efficient homomorphic encryption for Cross-Silo federated learning," in *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, 7 2020, pp. 493–506. [Online]. Available: https://www.usenix.org/conference/atc20/presentation/zhang-chengliang

[53] L. Zhang, J. Xu, P. Vijayakumar, P. K. Sharma, and U. Ghosh, "Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 5, pp. 2864–2880, 2023.

[54] J. Ma, S.-A. Naas, S. Sigg, and X. Lyu, "Privacy-preserving federated learning based on multi-key homomorphic encryption," *International Journal of Intelligent Systems*, vol. 37, no. 9, pp. 5880–5901, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/int.22818

[55] C. Li, N. Kumar, Z. Song, S. Chakrabarti, and M. Pistoia, "Privacy-preserving quantum federated learning via gradient hiding," *Quantum Science and Technology*, vol. 9, no. 3, p. 035028, 5 2024. [Online]. Available: https://dx.doi.org/10.1088/2058-9565/ad40cc

[56] R. Huang, X. Tan, and Q. Xu, "Quantum federated learning with decentralized data," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 28, no. 4: Mach. Learn. in Photon. Commun. and Meas. Syst., pp. 1–10, 2022.

[57] S. Y.-C. Chen and S. Yoo, "Federated quantum machine learning," *Entropy*, vol. 23, no. 4, 2021. [Online]. Available: https://www.mdpi.com/1099-4300/23/4/460

[58] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[59] M. L. G. ULB, "Credit card fraud detection," 3 2018. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[60] W. Li, S. Lu, and D.-L. Deng, "Quantum federated learning through blind quantum computing," *Science China Physics, Mechanics & Astronomy*, vol. 64, no. 10, 9 2021. [Online]. Available: http://dx.doi.org/10.1007/s11433-021-1753-3

[61] J. Heredge, N. Kumar, D. Herman, S. Chakrabarti, R. Yalovetzky, S. H. Sureshbabu, C. Li, and M. Pistoia, "Prospects of privacy advantage in quantum machine learning," 2024. [Online]. Available: https://arxiv.org/abs/2405.08801

[62] C.-H. H. Yang, J. Qi, S. Y.-C. Chen, P.-Y. Chen, S. M. Siniscalchi, X. Ma, and C.-H. Lee, "Decentralizing feature extraction with quantum convolutional neural network for automatic speech recognition," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6 2021. [Online]. Available: http://dx.doi.org/10.1109/ICASSP39728.2021.9413453

[63] W. M. Watkins, S. Y.-C. Chen, and S. Yoo, "Quantum machine learning with differential privacy," *Scientific Reports*, vol. 13, no. 1, 2 2023. [Online]. Available: http://dx.doi.org/10.1038/s41598-022-24082-z

[64] C. Hirche, C. Rouzé, and D. S. França, "Quantum differential privacy: An information theory perspective," *IEEE Transactions on Information Theory*, vol. 69, no. 9, pp. 5771–5787, 2023.

[65] R. Rofougaran, S. Yoo, H.-H. Tseng, and S. Y.-C. Chen, "Federated quantum machine learning with differential privacy," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4 2024. [Online]. Available: http://dx.doi.org/10.1109/ICASSP48485.2024.10447155

[66] L. Sünkel, P. Altmann, M. Kölle, and T. Gabor, "Quantum federated learning for image classification," in *Proceedings of the 16th International Conference on Agents and Artificial Intelligence - Volume 3: ICAART*, INSTICC. SciTePress, 2024, pp. 936–942.

[67] N. Franco, A. Sakhnenko, L. Stolpmann, D. Thuerck, F. Petsch, A. Rüll, and J. M. Lorenz, "Predominant aspects on security for quantum machine learning: Literature review," 2024. [Online]. Available: https://arxiv.org/abs/2401.07774

[68] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, *Evasion Attacks against Machine Learning at Test Time*. Springer Berlin Heidelberg, 2013, p. 387–402. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40994-3_25

[69] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," 2016. [Online]. Available: https://arxiv.org/abs/1608.04644

[70] H. Liao, I. Convy, W. J. Huggins, and K. B. Whaley, "Robust in practice: Adversarial attacks on quantum machine learning,"

Physical Review A, vol. 103, no. 4, 4 2021. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.103.042427

[71] M. Weber, N. Liu, B. Li, C. Zhang, and Z. Zhao, "Optimal provable robustness of quantum classification via quantum hypothesis testing," *npj Quantum Information*, vol. 7, no. 1, 5 2021. [Online]. Available: http://dx.doi.org/10.1038/s41534-021-00410-5

[72] L. Zhou and M. Ying, "Differential privacy in quantum computation," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, 8 2017. [Online]. Available: http://dx.doi.org/10.1109/CSF.2017.23

[73] Y. Du, M.-H. Hsieh, T. Liu, D. Tao, and N. Liu, "Quantum noise protects quantum classifiers against adversaries," *Phys. Rev. Res.*, vol. 3, p. 023153, 5 2021. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.3.023153

[74] M. Das and T. Bolisetti, "Variational quantum neural networks (vqnns) in image classification," 2023. [Online]. Available: https://arxiv.org/abs/2303.05860

[75] J. Li, X. Yang, X. Peng, and C.-P. Sun, "Hybrid quantum-classical approach to quantum optimal control," *Phys. Rev. Lett.*, vol. 118, no. 15, 4 2017.

[76] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," *Phys. Rev. A (Coll. Park.)*, vol. 98, no. 3, 9 2018.

[77] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, "General parameter-shift rules for quantum gradients," *Quantum*, vol. 6, no. 677, p. 677, 3 2022.

[78] Y. Huang, Z. Song, K. Li, and S. Arora, "Instahide: Instance-hiding schemes for private distributed learning," *ArXiv*, vol. abs/2010.02772, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:221081368

[79] G. Welch and G. Bishop, "An introduction to the kalman filter," University of North Carolina at Chapel Hill, USA, Tech. Rep., 1995.

[80] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, "Quantum computing with Qiskit," 2024.

[81] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi, J. M. Arrazola, U. Azad, S. Banning, C. Blank, T. R. Bromley, B. A. Cordier, J. Ceroni, A. Delgado, O. D. Matteo, A. Dusko, T. Garg, D. Guala, A. Hayes, R. Hill, A. Ijaz, T. Isacsson, D. Ittah, S. Jahangiri, P. Jain, E. Jiang, A. Khandelwal, K. Kottmann, R. A. Lang, C. Lee, T. Loke, A. Lowe, K. McKiernan, J. J. Meyer, J. A. Montañez-Barrera, R. Moyard, Z. Niu, L. J. O'Riordan, S. Oud, A. Panigrahi, C.-Y. Park, D. Polatajko, N. Quesada, C. Roberts, N. Sá, I. Schoch, B. Shi, S. Shu, S. Sim, A. Singh, I. Strandberg, J. Soni, A. Száva, S. Thabet, R. A. Vargas-Hernández, T. Vincent, N. Vitucci, M. Weber, D. Wierichs, R. Wiersema, M. Willmann, V. Wong, S. Zhang, and N. Killoran, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," 2022. [Online]. Available: https://arxiv.org/abs/1811.04968

[82] E. Bagdasaryan, O. Poursaeed, and V. Shmatikov, *Differential privacy has disparate impact on model accuracy*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[83] A. Ziller, D. Usynin, R. Braren, M. Makowski, D. Rueckert, and G. Kaissis, "Medical imaging deep learning with differential privacy," *Scientific Reports*, vol. 11, no. 1, 6 2021. [Online]. Available: http://dx.doi.org/10.1038/s41598-021-93030-0

[84] G. Papadopoulos, Y. Satsangi, S. Eloul, and M. Pistoia, "Absolute variation distance: An inversion attack evaluation metric for federated learning," in *Advances in Information Retrieval*, N. Goharian, N. Tonellotto, Y. He, A. Lipani, G. McDonald, C. Macdonald, and I. Ounis, Eds. Cham: Springer Nature Switzerland, 2024, pp. 243–256.

[85] J. Qian, H. Nassar, and L. K. Hansen, "Minimal model structure analysis for input reconstruction in federated learning," 2021. [Online]. Available: https://arxiv.org/abs/2010.15718

[86] D. A. Kreplin and M. Roth, "Reduction of finite sampling noise in quantum neural networks," *Quantum*, vol. 8, p. 1385, 6 2024. [Online]. Available: https://doi.org/10.22331/q-2024-06-25-1385

[87] M. B. Rhudy, R. A. Salguero, and K. Holappa, "A kalman filtering tutorial for undergraduate students," *International Journal of Computer Science & Engineering Survey*, vol. 8, no. 1, pp. 1–9, 2017.

APPENDIX A
KALMAN FILTER

The Kalman model's matrices and vectors, as specified in the Kalman filter methods [79], [87], are initialized ($k = 0$)
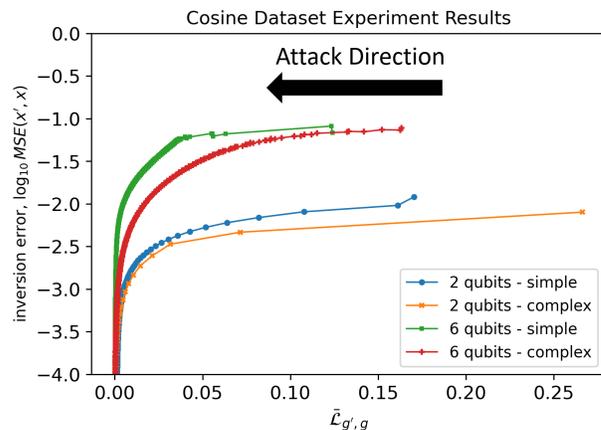
Fig. 12. These plots feature 2 and 6-dimensional Cosine datasets $\mathbf{x}$. Fig. 6 shows the Simple and Complex models respectively. Each line provides a visual of the optimization path taken to move the random point $\mathbf{x}'$ (start of the line on the right) closer to the original $\mathbf{x}$ (at coordinates x-axis$= 0$, y-axis$= 0$). The y-axis is the $log10$ MSE distance between the $\mathbf{x}'_k$ and the $\mathbf{x}$, with $k = \{1, 2, \cdots, K\}$ the attack steps and K the total steps. The x-axis is the distance between the gradient $\mathbf{g}'_k$ of the $\mathbf{x}'_k$ and the gradient $\mathbf{g}$

as all-ones elements ($\mathbf{F}, \mathbf{Q}, \mathbf{P}, \mathbf{H}, \mathbf{R}$) and zero initial state. Therefore, we do not assume a prior knowledge of the state or the transitions $\mathbf{F}$. The Kalman filter at each iteration step is divided into two parts, the prediction and the measurement update. The process starts by predicting the state from previous iteration $\mathbf{x}'^{k-1}$ (Eq.10) using the transition matrix $\mathbf{F}$:

$$
\begin{aligned}
\hat{\mathbf{x}}'^{k-1} &= \mathbf{F}\mathbf{x}'^{k-1}, \\
\hat{\mathbf{P}}^{k-1} &= \mathbf{F}\mathbf{P}^{k-1}\mathbf{F}^T + \mathbf{Q},
\end{aligned}
\tag{12}
$$

where $\mathbf{P}^{k-1}$ is the prior covariance matrix of the state and $\mathbf{Q}$ is the process noise covariance. The update step refines the state estimate using the current measurement $\mathbf{x}'^k$ and the predicted previous state $\hat{\mathbf{x}}'^{k-1}$:

$$
\begin{aligned}
\tilde{\mathbf{x}}'^k &= \mathbf{x}'^k - \mathbf{H}\hat{\mathbf{x}}'^{k-1}, \\
\mathbf{S}^k &= \mathbf{H}\hat{\mathbf{P}}^{k-1}\mathbf{H}^T + \mathbf{R}, \\
\mathbf{D}^k &= \hat{\mathbf{P}}^{k-1}\mathbf{H}^T\mathbf{S}^{-1,k}, \\
\mathbf{x}'^k &= \hat{\mathbf{x}}'^{k-1} + \mathbf{D}^k\tilde{\mathbf{x}}'^k, \\
\mathbf{P}^k &= (\mathbf{I} - \mathbf{D}^k\mathbf{H})\hat{\mathbf{P}}^{k-1},
\end{aligned}
\tag{13}
$$

where $\mathbf{H}$ is the measurement matrix, $\mathbf{R}$ is the measurement noise covariance, $\mathbf{S}^k$ the innovation covariance, and $\mathbf{D}^k$ is the Kalman gain.

APPENDIX B
SUPPORTING ANALYSIS

Fig.12 shows the attacks on cosine generated dataset, for 2 and 6 dimensions. The models are the Simple and the Complex from Fig. 6. The attack gets as close as $MSE(\mathbf{x}', \mathbf{x})$ is $\leq$ $1e-10$.
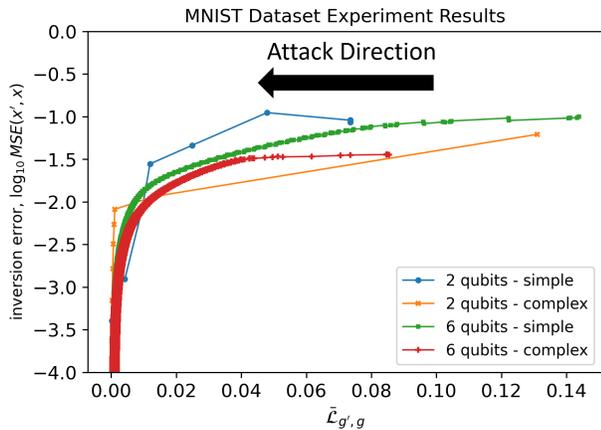
Fig. 13. These plots feature 2 and 6-dimensional MNIST data. Fig. 6 shows the Simple and Complex models respectively. Each line provides a visual of the optimization path taken to move the random point $\mathbf{x}'$ (start of the line on the right) closer to the original $\mathbf{x}$ (at coordinates x-axis$= 0$, y-axis$= 0$). The y-axis is the $log10$ MSE distance between the $\mathbf{x}'_k$ and the $\mathbf{x}$, with $k = \{1, 2, \cdots, K\}$ the attack steps and K the total steps. The x-axis is the distance between the gradient $\mathbf{g}'_k$ of the $\mathbf{x}'_k$ and the gradient $\mathbf{g}$
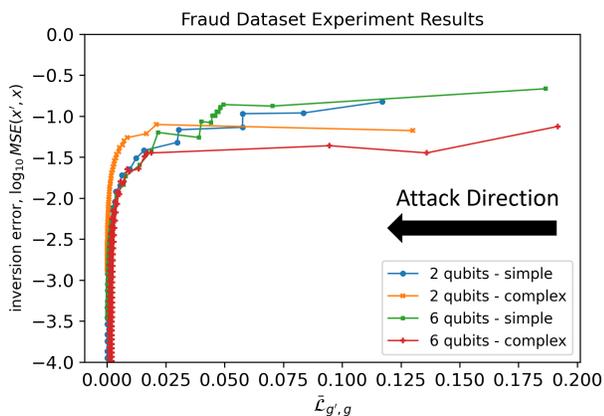
Fig. 14. These plots feature 2 and 6-dimensional credit Fraud classification data [59]. **For the complex model we relied on the trainable architecture $ZZFeatureMap$ [32], [33] and $EfficientSU2$ [31].** Each line provides a visual of the optimization path taken to move the random point $\mathbf{x}'$ (start of the line on the right) closer to the original $\mathbf{x}$ (at coordinates x-axis$= 0$, y-axis$= 0$). The y-axis is the $log10$ MSE distance between the $\mathbf{x}'_k$ and the $\mathbf{x}$, with $k = \{1, 2, \cdots, K\}$ the attack steps and K the total steps. The x-axis is the distance between the gradient $\mathbf{g}'_k$ of the $\mathbf{x}'_k$ and the gradient $\mathbf{g}$

Fig. 15. Attack on Fraud dataset of the 6 dimensional VQNN $ZZEfficient$ model. We show that when we attack the 6-VQNN model for more than $k \geq 200$ the results will converge to as low as $1e{-}10$.
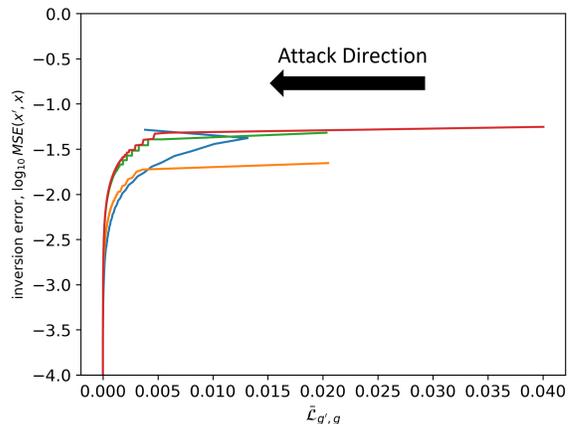
Fig. 16. We perform 4 attacks by using different initial model weights each time. We utilize a 2-qubit/inputs Simple model (6) on Cosine data. The results converge to as low as $1e{-}10$.

even when we target gradients coming from a trained model $\mathbf{g}$. This shows that in real-life FL settings the attack algorithm can approximate the base data $\mathbf{x}$ under most scenarios we have put forward.

Fig.13 pictures the attack on the MNIST dataset [58]. The $MSE(\mathbf{x}', \mathbf{x})$ error of the attacks is $\leq 1e{-}10$. The models are the Simple and the Complex from Fig. 6.

Fig. 14 illustrates the attacks on the Fraud dataset [59]. The VQNN models are the Simple and for complex the $ZZEfficient$ (Fig. 6). The $MSE(\mathbf{x}', \mathbf{x})$ is $\leq 1e{-}10$.

Fig. 15 shows that for runs with more that $k > 200$ iterations the algorithm eventually converges to as low as $MSE(\mathbf{x}', \mathbf{x}) = 1e{-}10$.

Fig. 16 depicts attacks on the Cosine data by using different initialized weights for each attack attempt. It proves the robustness of the attack algorithm under different scenarios (instead of starting the weights with 1s).

In Fig. 17, we observe that the attack algorithm is successful

Fig. 17.   We perform 3 attacks using as a base gradient **g** the gradient after the successful training of a VQNN classification model (Fraud data, $ZZEfficient$, 6 qubits/dimensions). The results converge to as low as $1e-5$.

Appendix C
List of Symbols

The following list contains the symbols used in the text:

- $\boldsymbol{\theta}$: Vector of ansatz parameters/weights of the Quantum Machine Learning model.
- M: The total number of weights of a QVNN model.
- C: The total number of classes in the target values $\mathbf{y}$.
- Loss: The loss function of the Quantum Machine Learning model.
- $\mathbf{g}$: Vector of gradient $\partial\text{Loss}/\partial\boldsymbol{\theta}$.
- $\mathcal{L}_{\mathbf{g}',\mathbf{g}}$: The loss function of the Quantum Machine Learning model gradient $\mathbf{g}$ and an approximate gradient $\mathbf{g}$.
- ∇: The symbol of gradient.
- $\mathbf{x}$: A vector of the input data that feed into the Quantum Machine Learning model.
- $\mathbf{y}$: The target values/classes that are used during the training process.
- $\hat{\mathbf{y}}$: Predicted values.
- I: The total number of training steps/epochs. The total number of the data points.
- i: The training steps (iterations) during the training processes $i \in \{1, 2, \ldots, I\}$.
- B: The batch data size.
- b: The number of the batch segment.
- γ: The learning rate parameter during model training/optimization.
- C_l: The total number of clients in a Federated Learning framework.
- $(')$: We use the apostrophe/quote symbol to designate proxy elements, ($\mathbf{x}'$ proxy to original data $\mathbf{x}$ or f' proxy to original model f).
- J: The number of features/dimensions.
- f: A function that takes inputs $\mathbf{x}$ and parameters $\boldsymbol{\theta}$ and generates predicted values $\hat{\mathbf{y}}$.
- $\hat{\mathbf{e}}$: A unit vector.
- δ: A parameter for the finite difference method that approximates gradients.
- η: The learning rate of the gradient descent algorithm.
- q: The number of qubits of a VQNN model. For this paper, all of our cases are $J = q$. Number of qubits equals the number of inputs/dimensions of $\mathbf{x}$.
- Φ: The feature map of a VQNN model .
- U_{Φ}: The unitary operator of a VQNN model.
- ψ: The VQNN function that gives us the measurement using the parameterized circuit.
- V: The ansatz of a VQNN model.
- $\hat{O}$: A Hermitian operator used as an observable in the VQNN.
- $\otimes$: Tensor product.