Sensor applications

Hybrid Quantum-Classical Neural Network for Cloud-Supported In-Vehicle Cyberattack Detection

Mhafuzul Islam, Mashrur Chowdhury*, Zadid Khan, and Sakib Mahmud Khan*

Glenn Department of Civil Engineering, Clemson University, Clemson, SC 29634 USA
**Senior Member, IEEE*

Manuscript received February 9, 2022; accepted February 18, 2022. Date of publication February 25, 2022; date of current version April 7, 2022.

Abstract—A classical computer works with ones and zeros, whereas a quantum computer uses ones, zeros, and superpositions of ones and zeros, which enables quantum computers to perform a vast number of calculations simultaneously compared to classical computers. In a cloud-supported cyber−physical system environment, running a machine learning application in quantum computers is often difficult, due to the existing limitations of the current quantum devices. However, with the combination of quantum-classical neural networks (NN), complex and high-dimensional features can be extracted by the classical NN to a reduced but more informative feature space to be processed by the existing quantum computers. In this study, we developed a hybrid quantum-classical NN to detect an amplitude shift cyberattack on an in-vehicle controller area network dataset. We showed that by using the hybrid quantum-classical NN, it is possible to achieve an attack detection accuracy of 94%, which is higher than a long short-term memory NN (88%) or quantum NN alone (62%).

Index Terms—Sensor applications, clouds, cyberattack, sensor applications, quantum computing, quantum neural network (NN).

I. INTRODUCTION

The decoherence and mechanical errors in quantum computers can make it harder for the existing quantum computers to learn the underlying data pattern, affecting the performance [1]. With the recent advancement of near-term quantum processors, it is possible to use a combination of classical and quantum computers to reduce errors. In a hybrid quantum-classical setup, some computations are performed in quantum computers, and some are performed in classical computers. Such a setup can be used in a cloud-based cyber−physical systems (CPS) environment, where a controller area network (CAN) bus is connected to the cloud using a CAN logger attached to the OBD-II port of a vehicle. The CAN logger provides CAN bus data to the cloud to run multiple CPS applications in the cloud while meeting the delay requirements (e.g., data upload and download delay) of the vehicle's operation (Fig. 1) [2], [3]. In this letter, the hybrid quantum-classical cyberattack detection application is considered as executed in the cloud to detect a cyberattack on the in-vehicle CAN bus. We considered an amplitude shift cyberattack, where an attacker can compromise an electronic control unit (ECU) locally or remotely and can perform an amplitude shift attack on the in-vehicle CAN bus. The complex nature of the amplitude shift attack that randomly changes the data field of a CAN frame makes it difficult to detect the attack. Studies showed that CAN buses used in existing vehicles do not have sufficient security features [4], [5], and the security can be improved using machine learning (ML) models. The study by Song *et al.* [5] showed an accuracy of 99% in detecting denial of service attacks using ML models. The recent study by Khan *et al.* [4] showed a detection accuracy of 87.9% on detecting amplitude shift attacks using a deep neural network. To improve the attack detection accuracy, we combined a quantum ML method, more specifically a quantum neural network (NN), with a classical NN. By leveraging the advantages of the near-term quantum computers, the study by Farhi and Neven [6], [7] presented a general quantum NN architecture that was able to classify a handwritten digit dataset [i.e., Modified National Institute of Standards and Technology (MNIST)]. However, using such a quantum-only approach yields a lower classification accuracy. A recent study showed that the use of a hybrid quantum-classical NN approach can achieve a higher classification accuracy [8]. However, this approach has not been applied in a cloud-based in-vehicle cyberattack detection system. Using a cloud-based hybrid quantum-classical NN, we can overcome the existing limitations of quantum computers and develop quantum computing applications for in-vehicle cyberattack detection.

The objective of this letter is to evaluate the performance of a cloud-supported hybrid quantum-classical NN to detect an amplitude shift attack compared to an existing long short-term memory (LSTM) NN or a quantum NN alone. Sections II and III discuss the amplitude shift attack model and the dataset used in this experiment, respectively. Different types of NN models for cyberattack detection are discussed in Section IV. The experimental setup is discussed in Section V and findings from our experiment are discussed in Section VI. Finally Section VII concludes this letter.

II. ATTACK MODEL

We created the amplitude shift cyberattack following the study by Khan *et al.* [4]. In the amplitude shift attack, the amplitude of a feature of the in-vehicle network data is shifted (up or down) by a random constant value within a time interval. This random value can be both positive and negative, which is added to the original values of a feature. This simulates the scenario when an ECU is compromised by malware injection, which alters the course of ECU execution by changing the amplitude of the output signal from an ECU. Although the amplitude of a feature changes in this attack, due to the addition of a constant value to the amplitude of a signal, the trend of variations over time remains unchanged. This complex cyberattack is difficult to detect by the traditional cyberattack detection system, as the trend remains the same for the compromised feature.

Corresponding author: Mhafuzul Islam (mdmhafi@clemson.edu).
Associate Editor: S. Kia.
Digital Object Identifier 10.1109/LSENS.2022.3153931

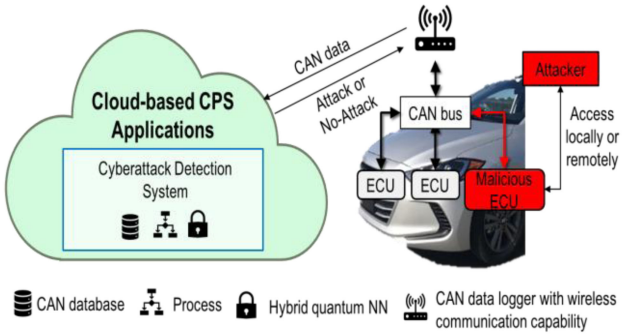

Fig. 1. Cloud-supported framework for in-vehicle cyberattack detection.

TABLE I. Features in HCRL Dataset

Feature no.	Feature name	Value range
1	TQI_COR_STAT	0.00-3.00
2	TQI_ACOR	0.00-99.61
3	N	0.00-16383.75
4	TQI-EMS11	0.00-99.61
5	TQFR	0.00-99.61
6	VS	0.00-254.00
7	BRAKE_ACT	0.00-3.00
8	TPS	0.00-104.69
9	PV_AV_CAN	0.00-99.61
10	TQI_MIN	0.00-99.61
11	TQI-EMS16	0.00-99.61
12	TQI_TARGET	0.00-99.61
13	TQI_MAX	0.00-99.61

III. DATASET

To create an attack dataset, we first required an in-vehicle dataset which is attack-free. We used the dataset created by the Hacking and Countermeasure Research Lab (HCRL) containing CAN data logged from a KIA soul vehicle [9]. The HCRL raw dataset contains different fields such as CAN ID, DATA, and Timestamp. A generic Database CAN (DBC) file is used for decoding the raw CAN bus data. The DBC file was collected from the OpenDBC repository [10]. The decoding information in the DBC file was used to convert data bits into feature values. The features contain data from different in-vehicle sensors, as shown in Table I. The HCRL dataset contains 13 features with 95200 timesteps (each timestep is 0.1 s). We used amplitude shift attack in our earlier study [4], where the amplitude of a feature in the CAN dataset was shifted (up or down) by a random value (i.e., value within the range observed during regular operations, as shown in Table I) within a time interval. Although the amplitude of a feature changed in this attack, the trend of variations over time remained unchanged. Using this attack, we created an attack dataset from the attack-free HCRL dataset. All features were compromised for one interval in the training dataset and another interval in the testing dataset. As a result, each feature was attacked once in training and once in testing. The data from other features remain unchanged during attack on one feature. More details about the attack dataset can be found in Table II.

IV. CYBERATTACK DETECTION

In our hybrid quantum-classical NN (Fig. 2), first, we preprocessed the in-vehicle CAN bus dataset and constructed a CAN image dataset [5]. We performed feature extraction using classical convolution neural network (CNN) from a 13 × 13 input and a 4 × 4 output

TABLE II. Details of Attack Dataset

	Total no. of timesteps	Timesteps of attack on each feature	No. of total attack labels	No. of total non-attack labels
Train	60000	2000	26000	34000
Test	35200	1000	13000	22200

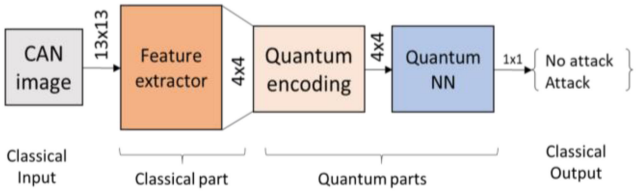

Fig. 2. Hybrid quantum-classical neural network.

from the CNN were converted into 16 qubit quantum data with 1 additional qubit as a control qubit. Then, we used the quantum data into a quantum NN to detect an in-vehicle cyberattack. A measurement was performed from the quantum NN to obtain a classical output of 1 (i.e., attack) or 0 (i.e., no attack).

A. Data Preprocessing

We took the CAN attack dataset created in [4] and constructed a 13 × 13 CAN image from 13 consecutive CAN frames, where each row represented a single CAN frame and each column represented a data feature. We considered a 13 × 13 CAN image as we had 13 data features in our dataset following the similar method presented in [4]. The CAN image dataset can be represented by $D = \{(X_m, y_m)\}_{m=1}^{M}$. Here, X_m is a 13 × 13 CAN image, with a label $y_m \in \{0, 1\}$ representing no attack and attack, and M is the number of total samples in D. Table II shows a total of 95200 CAN frames (60000 training, and validation, and 35200 test data) were available. As we took 13 CAN frames stacked up vertically to create a 13 × 13 CAN image, the size of the final dataset was reduced by a factor of 13 times. Therefore, the final dataset contained 7000 images (as 95200/13 ≈ 7000). We divided the total images of $M = 7000$ into 80% training dataset (i.e., 5600 images) and 20% testing dataset (i.e., 1400 images). Ten percent of the training dataset (i.e., 560 images) was held out for the validation, and 90% of the training data (i.e., 5040 images) was used for NN model fitting.

B. Feature Extraction Using Classical Neural Network

As presented in [5], we used a CNN for extracting the features from a 13 × 13 CAN image and producing a 4 × 4 reduced image. Following [5], the feature extraction from CNN can be represented as follows:

$$L_{4\times 4} = L_{d-1} \bigcirc L_{d-2} \bigcirc L_{d-3} \ldots L_1 \bigcirc L_0. \quad (1)$$

$$L_l : x_{l-1} \rightarrow x_l = \varphi\left(W_l x_{l-1} + v_l\right). \quad (2)$$

where $L_{4\times 4}$ is the output of a CNN, d is the total number of layers, L_l is the l^{th} layer of the CNN, x_{l-1} and x_l are the input and output vectors of L_l , W_l is the weight, v_l is a bias vector, and φ is a nonlinear function. Hyperparameters, such as d, W_l, and v_l were optimized during the training phase of the CNN.

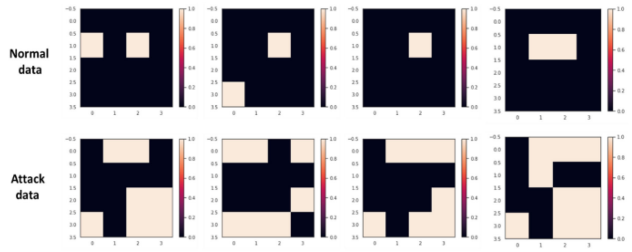

Fig. 3. Example of CAN feature image.

C. Quantum Encoding

To perform quantum operations (e.g., unitary operations, such as rotation and phase flip of qubits), we converted the classical data into quantum data. We performed a quantum basis encoding to represent the classical data into quantum data [7]. In a quantum basis encoding, each encoded quantum state is the bitwise translation of classical binary data to the corresponding qubit of the quantum system. Here, each classical data is an N-bit binary string: $x^m = (b_1, \ldots, b_N)$, with $b_i \in \{0, 1\}$ for $i = 1, \ldots, N$. N features are represented with a binary string (i.e., each bit means if the feature is present or not present). The binary image x^m was produced from $L_{4\times 4}$ using binary thresholding with a value of 0.5. Fig. 3 shows the classical binary image before performing quantum encoding. Each input data x^m was mapped to the quantum state $|x^m\rangle$. Following [7], the entire dataset can be represented in superpositions of computational basis states as

$$|D\rangle = \frac{1}{\sqrt{M}} \sum_{m=1}^{M} |x^m\rangle . \tag{3}$$

Each data sample can be represented as follows:

$$|x^m\rangle = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |b_i\rangle \tag{4}$$

where N is the number of bits (i.e., 16 in our case), b_i is a binary value representing if a feature is present or not, and $|b_i\rangle$ is the corresponding quantum qubit of the classical bit b_i.

D. Classification Using Quantum Neural Network

With quantum encoded data, we trained the parameterized quantum NN [6]. Following [6], the parameterized quantum NN performed unitary operations, such as rotation and phase flip, on qubits and can be represented as follows:

$$Q = Q_{q-1} \bigcirc Q_{q-2} \ldots Q_1 \bigcirc Q_0 \text{ and} \tag{5}$$

$$Q_m : |y\rangle \rightarrow y = U(w)\,|x^m\rangle \tag{6}$$

where Q is a binary output with $\{0, 1\}$ where 0 and 1 represent no attack and attack detected, respectively. Q has q number of layers, $U(w)$ is a unitary operation on $|x^m\rangle$ with a weight w, and y is the output after performing the unitary operation, $U(w)$.

V. EXPERIMENTAL SETUP

We compared the performance of the hybrid quantum-classical NN with the quantum NN alone and LSTM-based NNs. Each NN gives an output to classify attack as 1 or no attack as 0. For the quantum-only NN, we resized the 13 × 13 CAN image to 4 × 4 image, which was used as the input. The LSTM NN was developed following the study in [4] having the same model architecture. All NNs were developed using TensorFlow-quantum library [6]. We executed our hybrid quantum-classical NNs and quantum-only NN models in the Cirq simulator provided by TensorFlow-quantum, and it represented an ideal noise-free quantum computer. We performed the hyperparameter optimization such that we obtained the best cyberattack detection accuracy for each type of NN. The optimized hyperparameters for each type of NN are listed in Table III. The source code is available at GitHub [11].

TABLE III. Optimized Hyperparameters of Different Neural Networks (NNs)

Hyperparameters	Hybrid quantum-classical NN	Quantum-only NN	LSTM NN
Number of qubits	17	17	N/A
Number of epochs	20	50	100
Number of layers	6	7	3
Batch size	1	8	32
Total trainable parameters	96	128	1,090,049

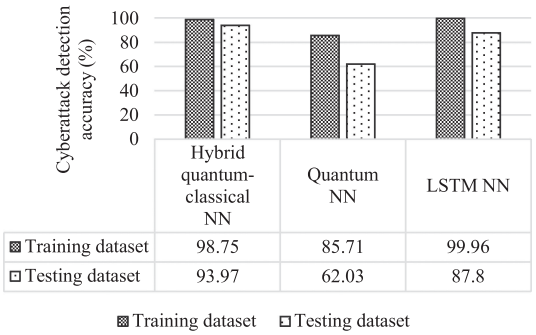

Fig. 4. Comparison of cyberattack detection accuracy for hybrid quantum-classical NN, quantum NN, and LSTM NN.

VI. EXPERIMENTAL RESULTS

As the amplitude shift attack detection falls into a binary classification model (i.e., attack or no-attack), we used the classification accuracy as the performance metric:

$$\text{Accuracy} = \frac{\text{TP} + \text{FN}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}} \tag{7}$$

where TP is number of true positives, TN is number of true negatives, FP is number of false positives, and FN is number of false negatives. Fig. 4 shows the attack detection accuracy on the training dataset and testing dataset. For both the training and testing datasets, the hybrid quantum-classical NN shows 98.7% and 93.9% accuracies, respectively. Here, the classical NN-based feature extractor was able to extract the features and the quantum NN was able to perform more accurate attack detection. The feature map extracted from the classical NNs, CNN in this case, allowed the parameterized quantum NN to explore the neighboring features in an exponentially large linear space, potentially allowing our hybrid quantum-classical NN to capture the patterns in the dataset (i.e., statistical distributions) more efficiently than LSTM NN and quantum NN alone. The quantum-only NN [7] shows 85.7% and 62.0% accuracy on the training dataset and testing dataset, respectively. We found that with the LSTM NN, the attack detection accuracies are 99.9% and 87.8% on the training and testing dataset, respectively.

VII. CONCLUSION

In a cloud-supported CPS environment, a hybrid quantum-classical NN performs better in detecting an in-vehicle cyberattack compared to a quantum NN, and an LSTM NN, as a hybrid quantum-classical NN, can capture the complex pattern of a cyberattack more efficiently. However, this study only demonstrates the use of quantum computers for amplitude shift cyberattack, which can be extended to detect other types of cyberattacks. Also, computational performance of real quantum computers will be conducted in future studies.

ACKNOWLEDGMENT

This work is based upon the work supported by the Center for Connected Multimodal Mobility (C^2M^2) (a U.S. Department of Transportation Tier 1 University Transportation Center) headquartered at Clemson University, Clemson, SC, USA. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of C^2M^2, and the US Government assumes no liability for the contents or use thereof.

REFERENCES

[1] V. Kulkarni, M. Kulkarni, and A. Pant, "Quantum computing methods for supervised learning," 2020, *arXiv:2006.12025*.

[2] L. Nkenyereye and J.-W. Jang, "Integration of big data for querying CAN bus data from connected car," in *Proc 9th Int. Conf. Ubiquitous Future Netw.*, 2017, pp. 946–950.

[3] H.-W. Deng, M. Rahman, M. Chowdhury, M. S. Salek, and M. Shue, "Commercial cloud computing for connected vehicle applications in transportation cyber-physical systems: A case study," *IEEE Intell. Transp. Syst. Mag.*, vol. 13, no. 1, pp. 6–19, Spring 2021.

[4] Z. Khan, M. Chowdhury, M. Islam, C. Huang, and M. Rahman, "Long short-term memory neural network-based attack detection model for in-vehicle network security," *IEEE Sens. Lett.*, vol. 4, no. 6, Jun. 2020, Art. no. 7500904.

[5] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, 2020, Art. no. 100198.

[6] M. Broughton *et al.*, "TensorFlow quantum: A software framework for quantum machine learning," 2020, *arXiv:2003.02989*.

[7] E. Farhi and H. Neven, "Classification with quantum neural networks on near term processors," 2018, *arXiv:2003.02989*.

[8] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, "Transfer learning in hybrid classical-quantum neural networks," *Quantum*, vol. 4, p. 340, 2020.

[9] H. Lee, S. H. Jeong, and H. K. Kim, "OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame," in *Proc. 15th Annu. Conf. Privacy, Secur. Trust (PST)*, 2017, pp. 57–5709.

[10] Commaai, "GitHub - commaai/opendbc: Democratize access to car decoder rings." Accessed: Mar. 3, 2022. [Online]. Available: https://github.com/commaai/opendbc

[11] M. Islam, "Github- Hybrid QNN," Accessed: Mar. 3, 2022. [Online]. Available: https://github.com/mahfuz195/hybrid_qnn