ADVANCED QUANTUM TECHNOLOGIES
www.advquantumtech.com

Quantum-Neural Network Model for Platform Independent Ddos Attack Classification in Cyber Security

Muhammed Yusuf Küçükkara, Furkan Atban, and Cüneyt Bayılmış*

Quantum Machine Learning (QML) leverages the transformative power of quantum computing to explore a broad range of applications, including optimization, data analysis, and complex problem-solving. Central to this study is the using of an innovative intrusion detection system leveraging QML models, with a preference for Quantum Neural Network (QNN) architectures for classification tasks. The inherent advantages of QNNs, notably their parallel processing capabilities facilitated by quantum computers and the exploitation of quantum superposition and parallelism, are elucidated. These attributes empower QNNs to execute certain classification tasks expediently and with heightened efficiency. Empirical validation is conducted through the deployment and testing of a QNN-based intrusion detection system, employing a subset of the CIC-DDoS 2019 dataset. Notably, despite employing a reduced feature set, the QNN-based system exhibits remarkable classification accuracy, achieving a commendable rate of 92.63%. Moreover, the study advocates for the utilization of quantum computing libraries such as Qiskit, facilitating QNN training on local machines or quantum simulators. The findings underscore the efficacy of a QNN-based intrusion detection system in attaining superior classification accuracy when confronted with large-scale training datasets. However, it is imperative to acknowledge the constraints imposed by the limited number of qubits available on local machines and simulators.

1. Introduction

Cloud computing systems provide data storage, processing, and application services via internet-based mechanisms. This constitutes fundamental infrastructure components of contemporary information technology environments, facilitating user accessibility. The growing utilization of these systems is accompanied by the emergence of novel security challenges. Specifically, threats such as insufficient data encryption, phishing, data leakage, and malware infiltration, prominently feature among the array of potential security risks confronting cloud computing systems. In this context, safeguarding the integrity of cloud systems has assumed paramount importance for both individual stakeholders and corporations. Recent technological advancements have become intricately intertwined with the mitigation of cyber security challenges. Consequently elevating the significance of computer-aided classification methods in contemporary discourse.

Deep learning as a machine learning methodology is widely preferred in the field of cyber security due to the robust classification accuracy demonstrated by custom-designed and pre-trained networks.[1,2] Traditional machine learning methodologies used for security breach detection often face significant barriers when for with the volume and complexity inherent in big data environments.[3,4]

Traditional approaches, typically based on predefined rules or patterns, struggle to effectively process and analyze the vast amounts of information generated in such large datasets. The application of large-scale data input-based ML classification

M. Y. Küçükkara, F. Atban
Sakarya University of Applied Sciences
Faculty of Technology
Department of Computer Engineering
Sakarya 54050, Turkey
E-mail: muhammedkucukkara@subu.edu.tr

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/qute.202400084

DOI: 10.1002/qute.202400084

M. Y. Küçükkara, F. Atban
Sakarya University
Institute of Natural Sciences
Department of Computer and Information Engineering
Sakarya 54050, Turkey

C. Bayılmış
Sakarya University
Faculty of Computer and Information Sciences
Department of Computer Engineering
Sakarya 54050, Turkey

methodologies significantly undermines the efficacy of Intrusion Detection Systems (IDS) across both their training and testing phases. The notable decline in performance results from the intricate processing of extensive datasets. The computational load imposed by ML algorithms hampers the IDS's accuracy in identifying and classifying potential security threats.

Within the domain of machine learning, there has been a burgeoning interest in leveraging quantum-based frameworks to enhance both speed and classification accuracy. These frameworks are designed to overcome the limitations of conventional computer architectures. They leverage parallel processing capabilities and the principles of quantum superposition to achieve faster and more efficient classification outcomes. QML stands as a paradigm shift in quantum computing going forward, bringing together traditional machine learning methodologies and quantum processing in a unified framework. In this framework, This convergence has the potential to facilitate the execution of computationally complex problems on quantum devices.

This study delves deeply into the effectiveness of a QNN approach customized for high-performance intrusion classification. It is strategically crafted to overcome the inherent limitations of handling big data and the computational constraints of conventional hardware architectures. The consequences of cyber-attacks can be financially and operationally burdensome. Achieving precise and rapid classification accuracy stands as a critical parameter in mitigating these consequences. In response to these challenges, research endeavors centered on quantum-based methodologies have assumed significance. By integrating the architectural principles of traditional neural networks with quantum computing principles, the QNN approach accelerates the processing and analysis of large datasets and improves their performance.[5] In this context, the advantages of QNNs over traditional methods are their parallel processing capabilities and fast and efficient classification results obtained by quantum superposition principles. QNN can identify patterns in large datasets faster and more accurately by utilizing elements such as the quantum data density matrix and quantum entanglement principles. This significantly increases the effectiveness of cyber attack detection systems. Many traditional machine learning and neural network approaches have shown commendable performance in cyber attack detection and classification tasks. Their ability to adapt to evolving attack typologies under varying operational conditions necessitates meticulous examination. QNN promise to improve the effectiveness and reliability of IDS by providing faster and more accurate classification capabilities in this dynamic threat environment. However, to the best of our knowledge, the exploration of QNNs remains relatively scarce in existing literature. Wei et al. developed a hybrid model for Distributed Denial of Service (DDoS) attack detection that combines an Autoencoder with a Multilayer Perceptron to identify key features and achieved an F1 score of over 98%.[6] Xu et al. improved intrusion detection accuracy to 99.7% using SMOTE and mutual information for dataset refinement.[7] Akgun et al. evaluated various deep learning models for DDoS detection and found that convolutional neural network (CNN) is highly effective with 99.99% binary accuracy.[1] Aldhyani et al. achieved 100% accuracy by combining Long-Short Term Memory (LSTM) and CNN-LSTM architectures.[8] Alkhudaydi et al. achieved 98.50% accuracy by integrating CatBoost and XGBoost classifiers with SMOTE.[9]

Kasongo et al. achieved 87.07% training accuracy on UNSW-NB15 using RNNs with XGBoost feature selection.[10] Nabi et al. showed that the Random Projection approach improved the accuracy of NSL-KDD dataset to 82%.[11] Hossain et al. presented an ensemble learning method using Random Forest and achieved over 99% accuracy.[12] Aldhyani et al. presented a Rock Hyrax Swarm Optimization model for Android malware detection and achieved 99.05% accuracy.[13] Diaba et al. achieved 97.8% accuracy in binary classification using a meta-heuristic algorithm with RBM.[14] Alsarhan et al. optimized support vector machine (SVM) using Genetic Algorithm and achieved higher accuracy.[15] Salvakkam et al. proposed an ensemble model for cloud computing intrusion detection with a recall rate of 92.14%.[16]

Studies[17–21] on the combination of IoT technology and cybersecurity, including general strategies have become an important research topic with the increasing potential of AI-based approaches.

Several methodological approaches to intrusion detection can be found in the literature, including hybrid methodologies,[6,8] optimization via metaheuristics,[13–15] data preprocessing techniques,[7,11] the classification aptitude of neural networks consequent to feature selection,[10] and the utilization of ensemble learning frameworks.[12,22,23] However, these methods face limitations in terms of speed and efficiency, especially for large datasets and complex data structures.

Quantum computing holds immense promise in offering solutions that transcend the limitations encountered by conventional methodologies in addressing these challenges.

Said et al. used a Quantum-SVM model for DDoS attack detection and achieved over 99% accuracy.[24] Kalinin et al. investigated QML methods for high performance intrusion detection and found that ML-based intrusion detection is effective with 98% accuracy.[25] Shara emphasized the importance of QML in cybersecurity.[26]

An overview of the literature shows that QML approaches have only been used to a limited extent in cybersecurity. However, QML methods have shown applications in different fields such as medicine, finance, and chemistry. Apart from the security domain, it is clear that QML applications have been successful in healthcare, such as drug discovery, disease prediction and healthcare system improvements,[27–30] as well as in financial forecasting[31-33] and chemistry.[34] This literature review demonstrates the potential of QML approaches for new and different applications in various disciplines. A general evaluation of the literature is shown in **Table 1**.

Quantum computing and the use of emerging mid-scale quantum technology offer the potential to improve the performance of classical machine learning algorithms. QML algorithms currently have low fault tolerance. However, it appears that research on quantum hardware could lead to the development of a new generation of error-tolerant devices. Therefore, discoveries in this area are of great importance.

1.1. Motivation and Contribution

The main motivation underpinning the methodology expounded in this study is the amalgamation of quantum machine learning with the pronounced performance benefits afforded by quantum

Table 1. An overview of the literature.

References	Application	Models	Technique	Dataset	Performance
Wei et al. 2021[6]	DDos Attack Classification	Autoenconder-Multi Layer Perceptron	Two deep learning - based hybrid approach	CIC-DDos 2019	F1 Score = +98%
Xu et al. 2023[7]	Intrusion and Anomaly Detection	Automated Machine learning (KNN, SVM, NN, Tree, Ensemble Model)"	For quality of dataset - Synthetic Minority Oversampling Technique For classification - ML	KDDcup99	Accuracy = 97%
Akgun et al. 2022[1]	DDos Attack Detection	Deep Neural Network (DNN), CNN, LSTM and Custom CNN	Design of custom CNN	CIC-DDos2019	Accuracy= 99.9%
Aldhyani et al. 2023[8]	DDos Attack Detection	CNN-LSTM	Combined with long short-term memory (CNN–LSTM)	CIC-DDos 2019	Precision= 100%
Kasongo et al. 2023[10]	Intrusion Detection	Different Recurrent Neural Network	XGBoost-based feature selection algorithm and Simple Recurrent Neural Network (RNN)	NSL-KDD and UNSW-NB15	Accuracy = 87.07%
Nabi et al. 2024[11]	Intrusion Detection	Decision Tree	J48 Tree - Random Projection	NSL- KDD	Accuracy = 79.01%
Hossain et al. 2023	Intrusion Detection	Random Forest, Gradient Boosting, Adaboost, Gradient XGBoost, Bagging, and Simple Stacking)"	Ensemble-based ML technique	SIMARGL21	Accuracy = +99%
Albakri et al. 2023[13]	Malware Detection and Classification	Rock Hyrax Swarm Optimization with deep learning-based Android malware detection	RHSO based feature subset selection (RHSO-FS) technique	Andro-AutoPsy	Accuracy = 99.05%
Diaba et al. 2023[14]	Intrusion Detection	RF-RBM	Meta Heuristic Algorithm	Power System Attack Detection Dataset	Accuracy = 97.8% for binary classification
Alsarhan et al. 2023[15]	Intrusion Detection	SVM	Optimizing SVM using Meta Heuristic Algorithms	NSL - KDD	Accuracy = 99%
Said et al. 2023[23]	Intrusion Detection	SVM	Qantum - SVM	CIC-DDos 2019	Accuracy= +99%
Kalinin et al. 2023[24]	Intrusion Detection	SVM-CNN	QSVM-Quantum-CNN	Created DataSet	Accuracy=98%
Our System	Intrusion Detection	Neural Networks	QNN	CIC-DDos 2019	Accuracy = 92.63%

parallelization. This approach delineates a crucial realm of inquiry aimed at enhancing the capacity of machine learning algorithms to discern intricate distributions within high-dimensional datasets. In our study, we hypothesize that the intelligent integration of quantum computing into the field of link prediction has the potential to improve prediction accuracy less data and features, taking advantage of the performance improvements identified in previous scientific research. We believe that this integration can yield promising results.

The primary contributions of our study are delineated as follows:

a) We leverage quantum advancements to project feature sets into quantum space, capitalizing on the advantages afforded by quantum parallelization.
b) We elucidate the efficacy of QNNs employing various learning approaches in the realm of intrusion detection, thereby illuminating their potential applicability across diverse domains.
c) We execute these operations on a feature subset within the dataset, enabling the evaluation of accuracy, speed, and balance utilizing a qubit-centric methodology. Compared to other studies, we report high performance with a much smaller number of features.
d) Our model facilitates a comparative analysis between predictions generated on a local computer and those produced by different simulators. The findings also showed the potential for future training on a real quantum computer.
e) Our study's findings unveil a precise decision mechanism, mitigating the pronounced error tendencies inherent in traditional machine learning methods through the integration of Quantum Computing. These results are poised to constitute a significant contribution to the existing literature in the field.

The rest of the paper is as follows: Section 2 presents in detail the materials and methods used in the study. Section 3 presents

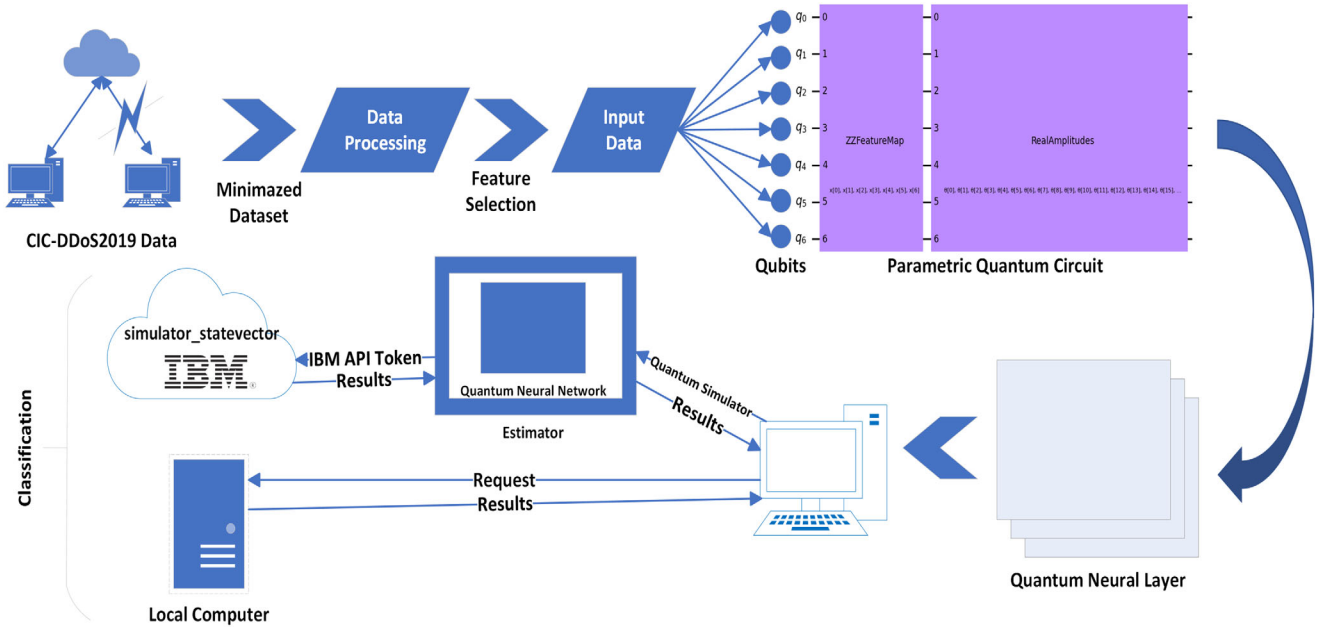


Figure 1. The flowchart of the QNN classification approach.

the experimental setup and comprehensive analysis results for the developed system. The obtained experimental results are evaluated in various test environments. Finally, Section 4 summarizes the conclusions of the study and points out potential future research areas.

2. Experimental Section

In this study, the complete spectrum of quantum advancements in neural networks was leveraged. Through the utilization of dynamic network properties, this data was encoded onto a quantum circuit designed for a neural network-based predictive model. This method utilizes quantum computing not only for feature projection but also for conducting training and optimization tasks within the quantum domain, facilitated by quantum circuits. The general flowchart of the study is given in **Figure 1**.

2.1. Dataset Information

DDos attacks represented a formidable challenge within cyber security, aiming to incapacitate targeted network infrastructures through inundating traffic.[35] Despite the development of numerous statistical methodologies to identify such incursions, the creation of an efficient detection framework that was both computationally economical and capable of real-time operation continues to be a significant area of academic interest.

The CIC-DDos 2019 dataset had been developed to closely mimic the characteristics of actual network traffic data (PCAPs), encompassing both benign and prevalent DDos attack vectors. The granularity of the network traffic analysis was achieved through the utilization of the CICFlowMeter-V3 tool, with the resultant data being organized into a CSV format.[36] This format included labelled flows with associated timestamps, source, and destination IP addresses, source and destination ports, utilized protocols, and identified attack classifications. During the dataset's preparation phase, considerable emphasis was placed on the simulation of realistic background traffic. To this end, the B-Profile methodology, as introduced by Sharafaldin et al., was employed to facilitate the accurate behavioural modelling of human interactions, thereby ensuring the generation of authentic, non-malicious background traffic within the testing environment.[37] In this study, 16000 samples were randomly selected and 7 features from the dataset.

Using balanced data allowed the model to learn both classes with equal weight, increasing the model's generalization ability and allowing it to detect both attacks and harmless situations with high accuracy. When unbalanced data was used, the model may overfit the dominant class, which might reduce the detection accuracy of the other class. The research by Wei et al. emphasizes the importance of using balanced training data and specifically states that achieving a 50% balance between different classes leads to higher accuracy measures.[38] This finding clearly demonstrated the importance of balanced data in improving classifier performance. Furthermore, the study by Kawakubo et al. emphasized the need to correct for bias due to class balance changes by weighted training based on the class proportion of the test data. This approach ensured that the classifier was robust to changes in class distributions between training and test datasets, further reinforcing the importance of balanced data in classifier training.[39] Real-world datasets were often imbalanced, but for intrusion detection systems, which were critical in the cybersecurity domain, it was known that models trained with imbalanced data may perform poorly. Therefore, it was a common approach to deal with data imbalance in order to improve the reliability and generalization ability of the model. In this study, the model was trained using balanced datasets. The balanced dataset allowed the model to learn both attack and harmless examples,

achieving high accuracy in both classes. This increases the generalization ability of the model and avoided the class bias that imbalanced datasets can introduce.[40] Increased generalizability increases the model's capacity to adapt to different data distributions in real-world scenarios. In this study, the data balance was manually adjusted, giving full control over which samples to include or exclude. This was particularly important for data quality and representativeness.

2.2. Feature Selection

One of the primary challenges encountered when employing quantum-based algorithms in the work was the limitation of available hardware. Both the local computing resources and simulators, such as IBM's statevector simulator, were restricted to a finite number of qubits. Given that each feature in the dataset required a separate qubit, it became imperative to reduce the number of features. This constraint necessitated the implementation of feature selection techniques to ensure the feasibility of the quantum computations while maintaining the integrity and performance of the model. Feature selection, a pre-processing step, aims to eliminate non-essential features from high-dimensional datasets. Filter-based feature selection methods, such as the KBest method, aim to reduce computational cost by selecting a specific number of top features. However, these methods do not directly optimize model performance. Instead, they provide a subset of features based on their overall information value. This means that although KBest had low computational cost, it might have limitations in improving the overall performance of the model. Studies in the literature had shown that feature selection methods based on filtering methods (such as K-best) were indirectly effective in improving the overall performance of the model, despite the low computational cost in applications.[41–43] Furthermore, reducing the number of features with the K-best method also optimized memory usage. This was an important advantage, especially when working with large data sets, and contributed to a more efficient use of computational resources. The KBest method analyzed 80 features in this study, selecting the seven most informative based on their dependency on the target variable through a scoring function. The KBest method calculated an f_{score} for each feature to identify those with the highest values and performance while reducing computational complexity and the risk of overfitting.[44] However, there was also a computational cost of evaluating each feature independently, especially in large data sets or when there were a large number of features. The total computational cost of the KBest algorithm was expressed as $O(m \cdot n + m \log m)$. Here, m represents the number of features and n represents the number of data points. The first term covered the independent evaluation of each feature, while the second term accounted for the sorting of these features.

2.3. Quantum Computing

The conceptual foundation of quantum computing was pioneered by Richard Feynman and Yuri Manin, grounding the computing paradigm in quantum mechanics principles.[45,46] This model of computing exhibits the capacity to solve specific problems with efficiency that classical computers find challenging.[47] Unlike classical computing, which relied on binary bits, quantum computing uses qubits, capable of representing both 0 and 1 simultaneously through superposition and entanglement, enhancing computational power.[48] This advancement enabled tackling problems beyond the scope of classical computing's.[49] Quantum gates, similar to classical logic gates but operating on qubits, were fundamental in crafting quantum algorithms and circuits, utilizing superposition, and entanglement to perform multidimensional operations.[50] This positions quantum computing as superior in fields like cryptography, optimization, and simulations.[51,52]

2.3.1. Entanglement and Superposition

Entanglement and superposition, cornerstone principles of quantum mechanics, collectively enable advancements in quantum computing and technologies. Entanglement facilitated an instantaneous connection between particles, regardless of distance, serving as a foundation for enhanced quantum communication, computation, and precision measurements, significantly boosting the functionality and security of quantum technologies. It aids in creating scalable quantum processors, neural networks, and sophisticated encryption methods, which were essential for surpassing classical systems in computational and communication tasks.[53,54] Concurrently, the superposition principle allowed quantum systems to coexist in multiple states until observed, a pivotal concept that underpins quantum mechanics and catalyzes a plethora of quantum effects. This principle extended from particles to macroscopic entities, enabling quantum information processing systems to operate in superpositions of causal structures, crucial for the complex calculations performed by quantum computers.[55–57] Collectively, these principles underscore the distinctive features of quantum coherence, which differentiate quantum physics from classical physics. They form the foundation for the development of quantum-enhanced technologies, including encryption and metrology, representing a substantial advancement in quantum computing and its various applications.[58–60]

2.4. Quantum Machine Learning

QML offered a transformative approach that harnesses the computational power of quantum mechanics to address the limitations of classical machine learning algorithms. By leveraging principles such as superposition and entanglement, QML provided enhanced capabilities for processing and analyzing complex datasets, making it particularly advantageous for tasks requiring high-dimensional data handling and intricate pattern recognition.

In QML, classical data was encoded into quantum states using quantum circuits. This encoding process utilized quantum gates, such as Hadamard gates for creating superposition and CNOT gates for generating entanglement, to map classical data into the quantum domain. The representation of datasets as quantum state vectors enabled QML algorithms to exploit the inherent parallelism of quantum computing, leading to more efficient data processing and higher accuracy in specific problem domains compared to classical methods.

A critical component of QML was the use of quantum circuits to perform learning tasks. These circuits incorporate specialized quantum algorithms designed to solve complex optimization and nonlinear operations that were challenging for classical algorithms. By employing quantum circuits, QML could execute these tasks with greater computational efficiency, opening new possibilities for tackling problems in various fields such as cryptography, optimization, and simulations.

In this study, QNNs was employed, a specific type of QML model, for training and classification tasks. QNNs integrated the quantum properties of traditional neural networks, leveraging quantum circuits to enhance the learning process. The architecture of QNNs included components such as feature maps and ansatz circuits, which were critical for encoding input data and optimizing the quantum state. This approach allowed QNNs to better capture the complexity within the data, providing more accurate and efficient solutions.

In QML approaches, classical data must first be projected into a complex quantum vector space using a feature map circuit. This vector space is expressed as follows:

$$\upsilon_{\phi}(x) = \prod_{d} \bigcup_{\phi} (x)^{H^{\otimes n}} \tag{1}$$

here, n represents the number of qubits utilized in the quantum circuit, indicating the dimensionality of the quantum system under consideration. The depth of the circuit, denoted as d, encompasses the number of sequential applications of essential quantum gates, notably including the Hadamard gate (H) and the entangling block $\upsilon_{\phi}(x)$. Here $\bigcup$ means that a union operation is performed on the phi sets. This represents the combination of different features defined by phi (ϕ). The Hadamard gate induces superposition by transforming basis states into a balanced mixture of 0 and 1 states, while the entangling block facilitates the entanglement of qubits, crucial for generating non-trivial quantum states and enabling complex quantum computations.

Furthermore, Pauli matrices are used to describe fundamental quantum gates and operations:

$$U_{\phi}(x) = \exp\left(i \sum_{x \subseteq \{1,\ldots,n\}} \phi_S(x) \prod_{k \in S} P_i \right) \tag{2}$$

here, $P_i \in I, X, Y, Z$ are the Pauli matrices, and $U_{\phi}(x)$ is the data mapping function. ϕ_S is a function that determines how data is mapped into quantum space. This function defines how the data is transformed over quantum bits. The connection between qubits or data points is defined by $S = \binom{n}{k}$ combinations, $k = 0, 1, \ldots, n$. These matrices are used as gates that represent operations performed on a qubit, providing a mathematical expression of the operations performed in quantum circuits.

In this study, the QNN model was trained using a quantum simulator, which facilitated the emulation of quantum circuits on classical hardware. This simulator-based approach allowed for the assessment and refinement of the QNN model before deploying it on actual quantum hardware. By conducting extensive training and optimization on the simulator, the potential of QNNs was demonstrated to achieve high classification accuracy with fewer features, highlighting the efficiency of quantum-enhanced learning methods.

In summary, QML represented a significant advancement in machine learning, offering unique advantages through the use of quantum computing principles. The integration of QNNs in this study showcased the potential of QML to revolutionize data processing and analysis, particularly in domains requiring complex pattern recognition and high-dimensional data handling.

2.5. Quantum Circuit

Quantum Circuit architecture and deployment were crucial in determining efficacy in Quantum Machine Learning. This research employed the QNNCircuit class from the Qiskit Machine Learning library.[61] The QNNCircuit class offered an integrated framework that combined feature maps and ansatz circuits, facilitating the streamlined integration of these critical components. The adoption of QNNCircuit not only elucidates the quantum mechanical foundations of the proposed model but also ensured an efficacious realization of outcomes on quantum computational platforms. The QNNCircuit created in the study is shown in **Figure 2**.

Figure 2 illustrates a quantum circuit comprising two primary blocks: ZZFeatureMap and RealAmplitudes. The ZZFeatureMap block was responsible for encoding classical data into quantum states by applying a series of quantum gates to the qubits, thus mapping the input data into a higher-dimensional quantum feature space. This process leverages entangling gates and single-qubit rotations to capture the correlations within the input data. Conversely, the RealAmplitudes block implemented a parameterized quantum circuit with real-valued parameters to optimize the quantum state. This block utilized a sequence of parameterized single-qubit rotations and entangling gates designed to effectively explore the quantum state space, thereby enhancing classification or regression tasks in quantum machine learning.

2.6. Quantum Neural Network

Building on the foundation laid by the QNNCircuit class, this study further explores the application of QNNs on quantum computing platforms, utilizing the Estimator and EstimatorQNN classes from the Qiskit Machine Learning library.[61] The Estimator class was a foundational tool for computing the expected values of observables within quantum circuits. Concurrently, EstimatorQNN, a neural network paradigm, leverages this foundation to deduce the expected values of a quantum circuit, which was parameterized by specified inputs and/or weights. The quantum circuit employed herein was a composite structure, incorporating a feature map for input data and an ansatz for the weights, designated as a circuit parameter within EstimatorQNN. This configuration facilitated the computation of the circuit's expectation values, which were pivotal for the model's training. Such methodology marks a significant advancement in quantum machine learning, aiming to enhance model precision while harnessing the quantum computational advantage. The QNN block diagram is shown in **Figure 3**.

Figure 3 illustrates the architecture of a QNN comprising three primary stages: Data Loading, Data Processing, and Mea-

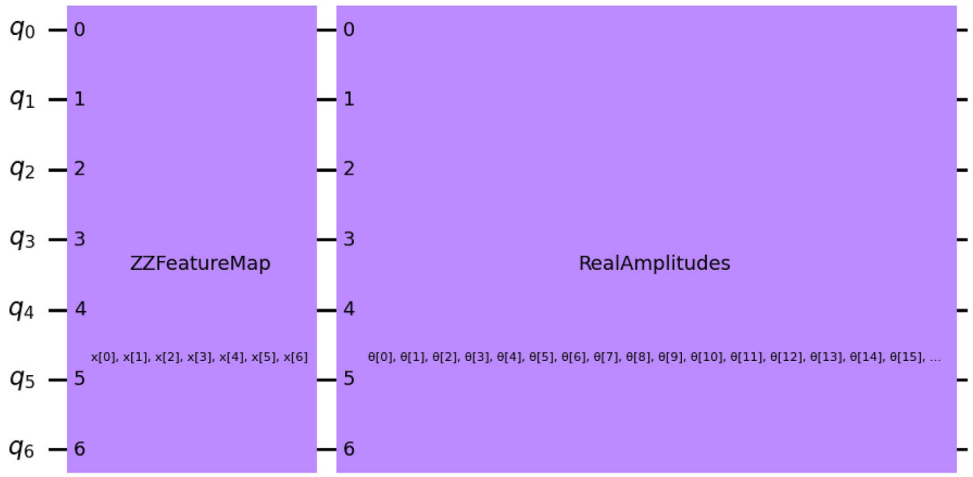


Figure 2. QNNCiruit architecture.

surement. In the Data Loading stage, classical data inputs were encoded into quantum states using a feature map, preparing the qubits (from q0 to q6) for subsequent processing. This encoding process leveraged quantum gates to transform classical data into a higher-dimensional quantum feature space. The encoded data was then passed to the Data Processing stage, represented by the 'Ansatz' block, which applied a series of parameterized quantum gates to the qubits. This stage is critical for exploring the quantum state space and optimizing the circuit for specific machine-learning tasks. Finally, the Measurement stage involved measuring the quantum states to extract meaningful classical information, which was then utilized for classification or regression tasks. This diagram effectively delineated the flow of data from classical input through quantum processing to measurement, providing a comprehensive overview of the QNN structure.

2.6.1. Classifier

The QNN Classifier exemplified the integration of quantum computing with machine learning, designed to undertake classification tasks through neural networks. It could support both binary and multidimensional outputs and allowed for the depiction of results as either direct class identifications or classifications over probability distributions. For multidimensional outputs, the framework permits the interpretation of each output vector either as a unified sample or as separate predictions utilizing one-hot encoding. This classifier enhanced the utility of quantum machine learning by incorporating adjustable loss function selections and a customizable optimization process.

2.7. Quantum Simulators

Quantum simulators, integral to quantum computing, facilitate the emulation of complex quantum systems unmanageable by classical means. IBM's contributions, notably the simulator_statevector, offer platforms for simulating quantum circuits under ideal and noisy conditions, aiding algorithm testing and hardware benchmarking.[62,63] Leveraging Qiskit, IBM's open-source quantum computing library, these simulators enabled diverse applications from quantum chemistry to cryptographic algorithm adaptation.[64] Their use in Hamiltonian simulations exemplifies their broad utility across scientific fields. IBM's quantum simulators were pivotal in advancing quantum computing, fostering algorithm development, and enhancing understanding in various domains.[63]

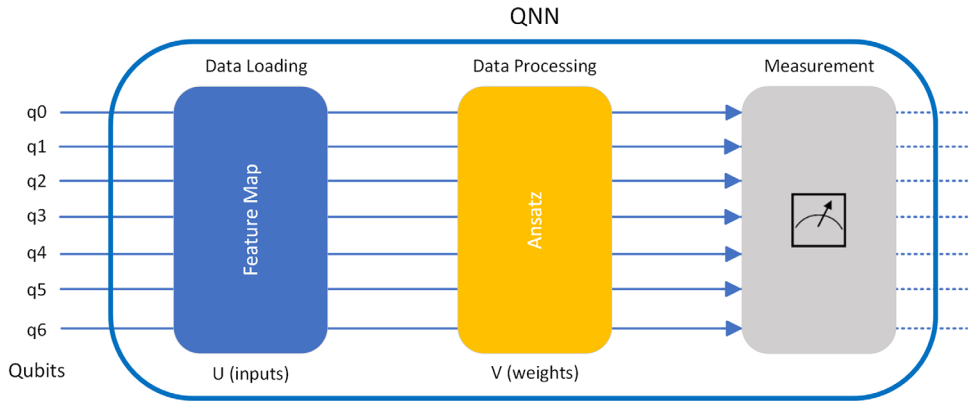


Figure 3. Quantum neural network block diagram.

2.8. Proposed Approach

In the proposed approach (1), data processing was initially performed. In this process, out of 80 features in the analyzed dataset, two of them were separated as label classes. In this study, only one of these label classes was used. This label indicated whether the feature is benign or attack. Out of the remaining 78 features, 66 features were obtained by eliminating 12 features with zero, repeated or null values. Among these features, seven features selected by the KBest feature selection method are as follows: "Fwd Packet Length Min, Fwd Packet Length Mean, Flow Bytes/s, Packet Length Min, Packet Length Max, Avg Packet Size, Avg Fwd Segment Size". After determining these features, the data preprocessing phase begins. The CIC-DDos-2019 dataset contains 431,370 data points, 5,449 of which were removed from the dataset due to zero, null or duplicate values. Of the remaining 425,921 data points, 1,000, 2,000, 4,000, 8,000, 16,000 sub-dataset were created to represent a homogeneous distribution (half labeled as attack, half as harmless). The features within the chosen datasets were normalized employing the Min–Max normalization technique.

The seven features selected in the proposed approach were encoded into qubits, the primary processing units of quantum computers. Each feature was encoded on a qubit. This coding process involved converting classical data into quantum states. Each qubit expressed quantum superpositions and entanglements that represent the relevant property. Seven features were thus converted into seven qubits, providing parallel processing and data analysis capacity using the quantum mechanical properties of the features. In this study, one of the biggest difficulties encountered in the use of quantum-based algorithms was hardware limitations. Both the local computer and simulators such as IBM statevector require the use of a certain number of qubits. In order to overcome these limitations, the number of features was reduced by using the KBest method for feature selection, taking into account the requirement of one qubit for each feature. Although the KBest method was a good feature selection method, it was stated that the main determining factor in classification success was the machine learning or quantum machine learning algorithms selected. For example, Wei et al. obtained high accuracy rates by using machine learning algorithms after feature selection with KBest method.[6] Similarly, Sharma et al. improved the classification accuracy by using various machine learning algorithms after feature selection with KBest method.[65] These studies show that the KBest method provides an effective feature selection, but the final success depends on the algorithms chosen. These findings emphasised that quantum machine learning algorithms were superior to the KBest method in classification success. Following the encoding of seven features into the qubit system, a quantum circuit was architectured to facilitate processing this encoded data. The circuit's design incorporated critical elements such as the ZZFeatureMap and Real Amplitudes. The ZZFeatureMap component is instrumental in entangling the encoded features into pairs among qubits, thereby delineating the intricate relationships between these features within a quantum mechanical framework. This entanglement process was pivotal for modelling the nonlinear interactions among features, enabling an extensive exploration of the feature space through quantum superposition. Conversely, the Real Amplitudes component was tasked with optimizing the circuit's state by adjusting the qubits' amplitudes or the real-valued components of the quantum state. This optimization was essential for enhancing the accuracy of operations executed on the qubits, which correspond to the set of features, thereby augmenting the quantum circuit's operational precision.

After the assembly of the quantum circuit, the next stage in the investigation involved incorporating the QNN model. The QNN architecture synergizes the foundational aspects of classical artificial neural networks with quantum computing's capabilities, offering substantial enhancements in data processing and learning efficacy. Preliminary evaluations of the QNN model were conducted locally, where no estimators were deployed to assess the model's performance. These initial tests aimed to ascertain the model's essential operational integrity and seamless integration with the quantum circuit, confirming that the model's functionality aligns with quantum mechanical principles and the feasibility of applying the theoretical framework in practice. Conversely, in the simulator-based experiments, an estimator was utilized for a more granular analysis of the model's performance. The QNN's output was scrutinized through the estimator, facilitating the computation of classification accuracy, error rates, and other pivotal metrics.

The optimization of the QNN model, incorporating seven qubits, was conducted on subsets of data comprising 1000, 2000, 4000, 8000, and 16000 instances. For the optimization endeavour, the COBYLA (Constrained Optimization BY Linear Approximations) optimizer was selected for a series of 100 iterations. COBYLA was a method for constrained optimization utilizing linear approximations, mainly applied within quantum computing to refine model parameters toward optimal values within predefined constraints. This phase of the study aimed to assess the model's efficacy across datasets of varying magnitudes. The application of the COBYLA optimizer aligned with the objective to enhance the classification accuracy of the QNN model, facilitating an efficient exploration within the high-dimensional parameter space. These experimental procedures, conducted on datasets of diverse sizes, comprehensively analyzed the model's adaptability to varying data volumes and the optimization process's influence on the model's overall performance.

3. Experimental Results

3.1. Experimental Setup

In conducting our study, we utilized two computational libraries: the Qiskit library, which allowed us to implement QNNs, and the IBM runtime library, which granted us access to IBM simulators. Using the Qiskit framework, we built and trained our QNN model with an API key acquired from the IBM Quantum platform. Throughout the training process, we concentrated on the simulator_statevector device. We also conducted model training on a local computational setup.

The model under consideration was executed on a local computing device and on the state_vector simulator provided by IBM Quantum simulators. The latter is a significant instrument for emulating quantum computing operations on classical computing systems. By representing the intricate states of quantum circuits via mathematical vectors, the simulator enables the theoretical evaluation of algorithms' accuracy and performance without

Table 2. Hardware specifications of the local computer.

Specifications	
CPU	AMD Ryzen 9 7900 12-Core
GPU	NVDIA GeForce GTX 3090
RAM	32 GB
Operating System	Ubuntu 20.04.6 LTS

necessitating access to a physical quantum computing apparatus. This simulation tool, developed by IBM and accessible through the Qiskit framework, plays a crucial role in advancing and validating quantum algorithms. The hardware specifications of the local computer used in the study are shown in **Table 2**.

3.2. Performance Metrics

The study utilizes accuracy as its performance metric. This metric is derived by computing the proportion of correctly classified samples to the total number of samples. It offers valuable insights into the model's ability to effectively distinguish between regular network traffic and DDos attacks.

a) True Positives (TP): The number of samples that the model correctly predicts as positive,
b) True Negatives (TN): The number of samples that the model correctly predicts as negative,
c) False Positives (FP): The number of samples for which the model incorrectly predicts as positive,
d) False Negatives (FN): The number of samples that the model incorrectly predicts as negative,
e) Total number of samples: Total number of samples (sum of TP, TN, FP and FN).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

3.3. Performance Evaluations

In this section, we evaluate the performance of the QNN architecture on the local computer and quantum simulator for different data sizes. In Scenario 1, we present the training results on the local computer. In Scenario 2, we analyze the results obtained on the quantum simulator.

Scenario 1: Quantum Computing on Local Computer for QNN Architecture: In this subsection, the QNN architecture is used for classification training in a local computer environment. QNN is an innovative approach to describe complex data patterns by exploiting quantum computing power.

For the training process, the homogeneous data set is divided into different subsets: 1,000, 2,000, 4,000, 8,000 and the largest subset of 16,000. These subsets were used to sample and then train the QNN sequentially.

Looking at the technical details of the QNN architecture, each quantum element (qubit) is used as a data processing unit. These qubits are encoded to represent the input data and then manipulated with quantum gates and measurements to perform data processing and classification.

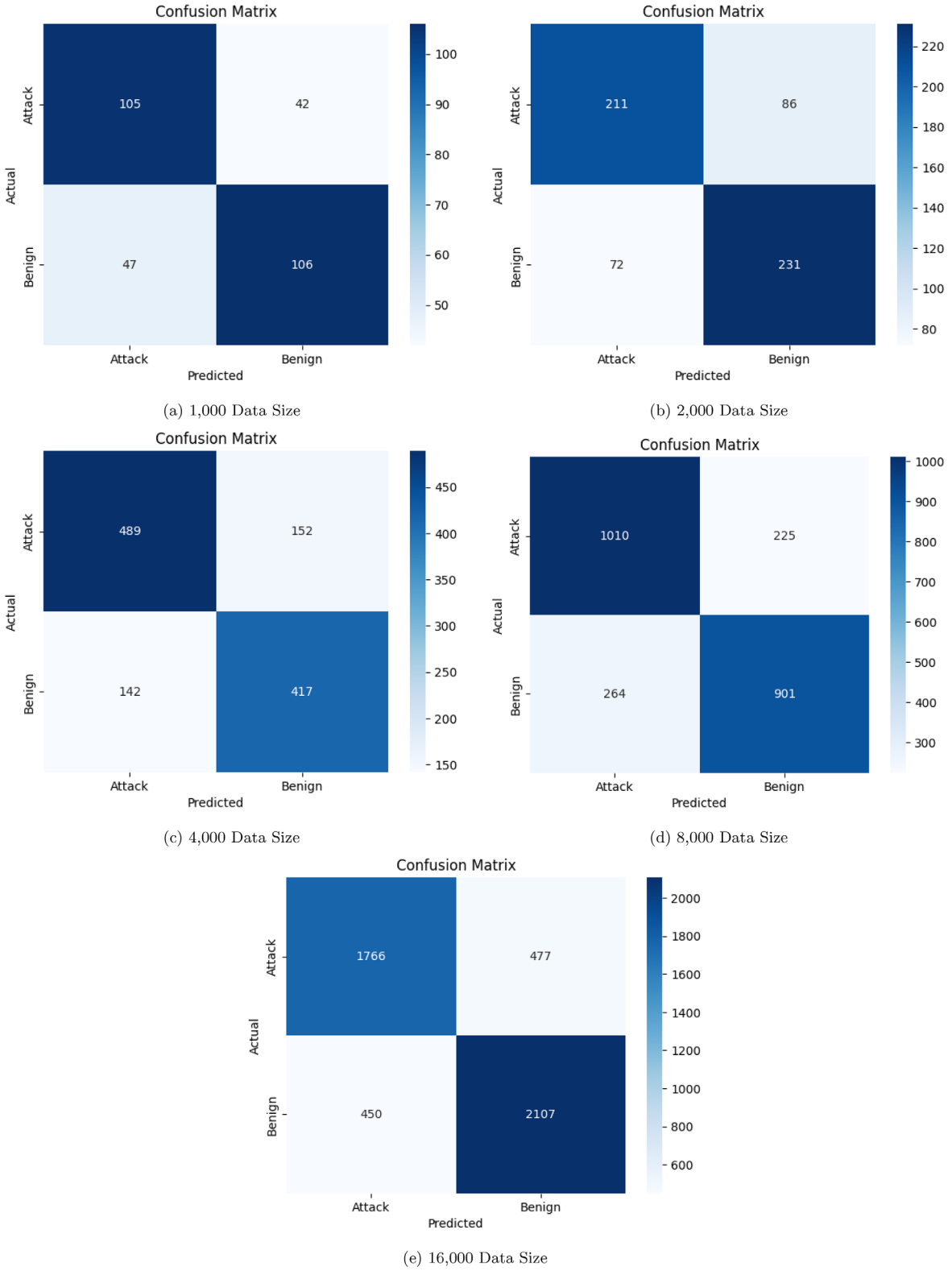
During the training process, the parameters of the QNN were tuned using an optimization method such as COBYLA (Constrained Optimization BY Linear Approximations). This optimization method was used to search for the best values of the parameters to improve the accuracy of the QNN.

The training was performed for 100 iterations. In particular, the training session using the largest data subset of 16,000 data points achieved an accuracy of 80.69%. This shows that the QNN can work effectively on large datasets using a local computer.

Regarding the increase in the amount of data, the training results show that the accuracy rate generally increases as the amount of data increases. It is also observed that this increase decreases after a certain point. The QNN has the capacity to learn more complex relationships when fed with more data. However, this increase in complexity can lead to overfitting of the model. Therefore, the trainings were finalized for 16,000 data. This point plays an important role in finding the balance between the amount of data and the complexity of the QNN. The confusion matrices of the QNN training results (different data sizes) on the local computer and quantum simulator are shown **Figure 4** and **5**.

Scenario 2: Quantum Computing on Quantum Simulator Results for QNN Architecture: This subsection describes the methodology for training classification using a QNN in a simulator environment.Ideal simulations of quantum circuits were performed using the Qiskit library and IBM's classical simulators. The results obtained in this scenario obtained are entirely based on classical simulations. Experiments on real quantum computers have not been performed. For the analysis, the datasets defined in Scenario 1 served as the basis, and the training of the QNN progressed sequentially through the identified data subsets. In the training process, an optimization technique such as COBYLA was chosen. The training of the model was subjected to an extensive series of 100 iterations. During the training of the QNN, a high success rate of 92.63% was achieved, especially on the 16,000 dimensional sub-dataset. This result shows that QNN can work effectively in the simulator environment and can be used to identify complex data patterns. This success of the simulator reveals the computational power of future noise-free quantum computers.

Regarding the increase in the number of data, the training results show that the use of larger data sets generally improves model performance. However, it is also observed that this increase is not linear and decreases after a certain point. At this point, factors such as the computational power of the simulator and the complexity of the QNN need to be taken into account. The state_vector simulator offers a number of advantages for QNN training. First, it is more accessible compared to real quantum computers. Furthermore, the simulator is a useful tool for evaluating QNN design and performance, allowing for de-



(a) 1,000 Data Size

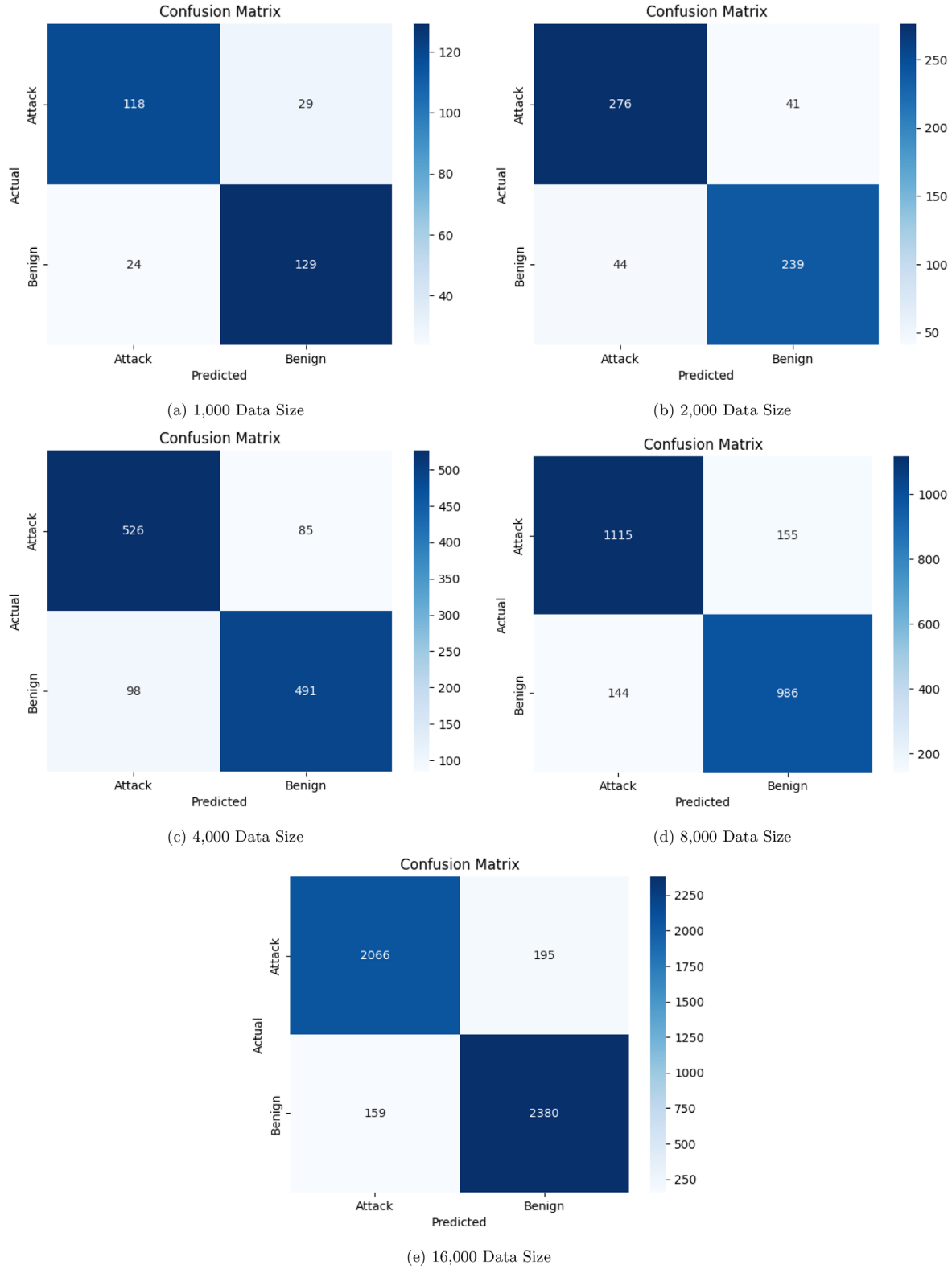
(b) 2,000 Data Size

(c) 4,000 Data Size

(d) 8,000 Data Size

(e) 16,000 Data Size

Figure 4. Confusion matrices from QNN performance on local computer.



(a) 1,000 Data Size

(b) 2,000 Data Size

(c) 4,000 Data Size

(d) 8,000 Data Size

(e) 16,000 Data Size

Figure 5. Confusion matrices for QNN performance on quantum simulator.

Table 3. Confusion matrices from QNN performance on the IBM Statevector quantum simulator.

Data Size	Accuracy		Precision		Recall		F1-Score	
	Local	Simulator	Local	Simulator	Local	Simulator	Local	Simulator
16000	80,69%	92,63%	78,73%	91,38%	79,69%	92,85%	79,21%	92,11%
8000	79,63%	87,54%	81,78%	87,80%	79,28%	88,56%	80,50%	88,18%
4000	75,50%	84,75%	76,29%	86,09%	77,50%	84,29%	76,89%	85,18%
2000	73,67%	85,83%	71,04%	87,07%	74,56%	86,25%	72,76%	86,66%
1000	70,33%	82,33%	71,43%	80,27%	69,08%	83,10%	70,33%	81,66%

sign improvement before conducting real experiments on quantum computer.

These numerical results reveal the effectiveness of QNN on quantum simulators and the impact of increasing the amount of data on QNN performance. All experimental results performed on the local computer and simulation are compared with different performance metrics in **Table 3**. Using classical numerical simulations of quantum circuits, significant differences in accuracy have been observed between simulations performed on a local classical computer and on an IBM classical computer. These differences are based on several factors such as numerical precision and performance optimisations. IBM's classical computers achieve higher accuracy rates in simulations by using higher precision arithmetic and more stable numerical techniques compared to local classical computers. Furthermore, IBM's classical computers are optimized for large-scale quantum simulations and use advanced techniques such as parallel processing and memory management. Detailed analysis of the results obtained reveals that the higher accuracy observed on the IBM platform is due to superior numerical stability and performance optimisations. In local simulations, relatively lower accuracy rates were obtained due to the absence of these advanced features. These findings suggest that IBM's classical computers provide higher accuracy and stability in quantum circuit simulations. The comparison of our study with other studies in the literature using the same data set is shown in **Table 4**.

According to the results of the analysis, the comparison of different intrusion detection methods is summarized in the table. Among these methods, QNN architecture, which is a new and innovative approach than traditional machine learning algorithms, stands out. First of all, one of the reasons why the QNN stands out is that it provides a high accuracy rate despite the higher number of data used compared to other methods. This shows that QNN utilizes data more efficiently. Although only seven features were used, an accuracy of 92.63% was achieved. While traditional methods such as DNN, CNN, SVM used in other studies usually require more features, QNN achieved good performance with fewer features. This shows that QNN can work more effectively with less information from the dataset and creates a cleaner and more authentic model by avoiding unnecessary features.

4. Conclusion and Discussion

This study highlights the potential of quantum computing in machine learning (ML) models to combat cyber attacks. A novel intrusion detection system for DDos attacks based on quantum ML models is presented. In this system, QNN architecture is preferred for the classification task. An important advantage of QNNs is the parallel processing capabilities provided by quantum computers and their capacity to handle specific computational problems more effectively. Another important advantage of QNNs is that, unlike traditional neural networks, they can utilize quantum superposition and quantum parallelism. These properties allow the network to work on multiple possible outcomes simultaneously, which allows certain classification tasks to be performed more quickly and efficiently. However, factors that determine the success of QNNs include the correct parameter settings and the design of appropriate quantum circuits. A properly designed QNN can solve certain classification problems more effectively than traditional methods. In this context, preprocessing steps such as removing duplicate records, improving performance, and applying minimum-maximum normalization operations are also important to obtain a clean and non-repetitive data set.

In this study, unlike other studies in the literature, a subset of the CIC-DDos 2019 dataset with much fewer features was obtained. The analysis of platforms and frameworks for the implementation of quantum computing shows that the most promising option is quantum computing libraries such as Qiskit. Tools like Qiskit can be used for training QNNs that can be run on local computers or quantum simulators. The benchmarks prove the effectiveness of the QNN model in terms of accuracy and consumption of computational resources. The results of the study indicate that a QNN-based intrusion detection system can provide protection with higher classification accuracy when a large-scale training data set is used. However, in such studies, the limited number of qubits in local computers and simulators should be taken into account. In our study, the training results on quantum simulators gave better accuracy than the results obtained from training on a local computer. In our study, the number of qubits

Table 4. Comparison of our study with other studies in the literature using the same data set.

References	Application	Models	Technique	Dataset	Features Number	Data Size	Performance
Wei et al. 2021[6]	DDos Attack Classification	Autoenconder-Multi Layer Perceptron	Two deep learning - based hybrid approach	CIC-DDos 2019	83	N/A	F1 Score = +98%
Akgun et al. 2022[1]	DDos Attack Detection	DNN, CNN, LSTM and Custom CNN	Design of custom CNN	CIC-DDos 2019	40	16500	Accuracy= 99.9%
Said et al. 2023[23]	Intrusion Detection	SVM	Qantum - SVM	CIC-DDos 2019	38	2950	Accuracy= +99%
Our System	Intrusion Detection	Neural Networks	QNN	CIC-DDos 2019	7	16000	Accuracy = 92.63%

on the local computer and on the simulator is limited. It is not possible to increase this number too much in our current systems. However, training with real IBM quantum computers has some differences and limitations. For training using IBM quantum computers, there is usually limited access and users can perform a limited number of operations in a given time frame. The results of the study show that high accuracy and low fault tolerance can be achieved with a much lower number of features and more data. These analyses show that the same experiments on a real quantum computer have great potential in terms of speed and time.

This study demonstrates the potential of the QNN architecture in cybersecurity while taking into account the limitations of quantum computers, such as limited access and computational cost. In the future, the focus will be on optimizing quantum algorithms and parallelization schemes for fast training of QML models. Also, Fault-tolerant quantum devices can provide high accuracy and performance beyond current classical simulations, which can further strengthen the applications of QNN in cyber security.

Acknowledgements

The authors acknowledged the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are openly available in CIC-DDos2019 at https://doi.org/10.1109/CCST.2019.8888419, reference number [36].

Keywords

cyber security, intrusion detection, network, quantum computing, quantum neural

Received: March 6, 2024
Revised: July 4, 2024
Published online: August 1, 2024

[1] D. Akgun, S. Hizal, U. Cavusoglu, *Comput. Security* **2022**, *118*, 102748.
[2] J. Martínez Torres, C. Iglesias Comesaña, P. J. García-Nieto, *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 2823.
[3] I. H. Sarker, *Ann. Data Sci.* **2023**, *10*, 1473.
[4] J. Bharadiya, *Eur. J. Technol.* **2023**, *7*, 1.
[5] D. Ristè, M. P. Da Silva, C. A. Ryan, A. W. Cross, A. D. Córcoles, J. A. Smolin, J. M. Gambetta, J. M. Chow, B. R. Johnson, *npj Quantum Inf.* **2017**, *3*, 16.
[6] Y. Wei, J. Jang-Jaccard, F. Sabrina, A. Singh, W. Xu, S. Camtepe, *IEEE Access* **2021**, *9*, 146810.
[7] H. Xu, Z. Sun, Y. Cao, H. Bilal, *Soft Comput.* **2023**, *27*, 14469.
[8] T. H. Aldhyani, H. Alkahtani, *Mathematics* **2023**, *11*, 233.
[9] O. A. Alkhudaydi, M. Krichen, A. D. Alghamdi, *Information* **2023**, *14*, 550.
[10] S. M. Kasongo, *Comput. Commun.* **2023**, *199*, 113.
[11] F. Nabi, X. Zhou, *Cyber Secur. Appl.* **2024**, *2*, 100033.
[12] M. A. Hossain, M. S. Islam, *Array* **2023**, *19*, 100306.
[13] A. Albakri, F. Alhayan, N. Alturki, S. Ahamed, S. Shamsudheen, *Appl. Sci.* **2023**, *13*, 2172.
[14] S. Y. Diaba, M. Shafie-Khah, M. Elmusrati, *IEEE Access* **2023**, *11*, 18660.
[15] A. Alsarhan, M. Alauthman, E. Alshdaifat, A.-R. Al-Ghuwairi, A. Al-Dubai, *J. Ambient Intell. Humaniz. Comput.* **2023**, *14*, 6113.
[16] D. B. Salvakkam, V. Saravanan, P. K. Jain, R. Pamula, *Cogn. Comput.* **2023**, *15*, 1593.
[17] J. D. D. H. Diego, J. Saldana, J. Fernández-Navajas, J. Ruiz-Mas, *IEEE Access* **2019**, *7*, 29942.
[18] M. G. Samaila, C. Lopes, É. Aires, J. B. Sequeiros, T. Simoes, M. M. Freire, P. R. Inácio, *Comput. Networks* **2021**, *199*, 108496.
[19] J. Wang, *Int. J. Commun. Syst.* **2022**, *35*, e4695.
[20] M. Tao, K. Ota, M. Dong, Z. Qian, *J. Parallel Distrib. Comput.* **2018**, *118*, 107.
[21] J. Srinivas, S. Mukhopadhyay, D. Mishra, *Ad Hoc Networks* **2017**, *54*, 147.
[22] M. N. Alatawi, N. Alsubaie, H. Ullah Khan, T. Sadad, H. S. Alwageed, S. Ali, I. Zada, *Secur. Commun. Netw.* **2023**, *2023*, 8048311.
[23] J. Zhu, X. Liu, *Communist Chin. Sci. Abstr.* **2024**, *115*, 109113.
[24] D. Said, *Energies* **2023**, *16*, 3572.
[25] M. Kalinin, V. Krundyshev, *J. Comput. Virol. Hacking Tech.* **2023**, *19*, 125.
[26] J. Shara, *Quantum* **2023**, *12*, 47.
[27] A. S. Bhatia, M. K. Saggi, S. Kais, *J. Chem. Inf. Model.* **2023**, *63*, 6476.
[28] G. Abdulsalam, S. Meshoul, H. Shaiba, *Intell. Autom. Soft Comput.* **2023**, *36*, 761.
[29] R. Ur Rasool, H. F. Ahmad, W. Rafique, A. Qayyum, J. Qadir, Z. Anwar, *Future Internet* **2023**, *15*, 94.
[30] S. Kavitha, N. Kaulgud, *Soft Comput.* **2023**, *27*, 13255.
[31] K. Miyamoto, K. Kubo, *IEEE Trans. Quantum Eng.* **2021**, *3*, 1.
[32] J. Alcazar, A. Cadarso, A. Katabarwa, M. Mauri, B. Peropadre, G. Wang, Y. Cao, *New J. Phys.* **2022**, *24*, 023036.
[33] S. Wilkens, J. Moorhouse, *Quantum Inf. Process.* **2023**, *22*, 51.
[34] M. Sajjan, J. Li, R. Selvarajan, S. H. Sureshbabu, S. S. Kale, R. Gupta, V. Singh, S. Kais, *Chem. Soc. Rev.* **2022**, *51*, 6475.
[35] J. Mirkovic, P. Reiher, *SIGCOMM Comput. Commun. Rev.* **2004**, *34*, 39.
[36] I. Sharafaldin, A. H. Lashkari, S. Hakak, A. A. Ghorbani, in *2019 International Carnahan Conference on Security Technology (ICCST)*, IEEE, Piscataway, NJ **2019**, pp. 1–8.
[37] A. Gharib, I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, in *2016 International Conference on Information Science and Security (ICISS)*, IEEE, Piscataway, NJ **2016**, pp. 1–6.
[38] Q. Wei, R. L. Dunbrack Jr, *PloS one* **2013**, *8*, e67863.
[39] H. Kawakubo, M. C. Du Plessis, M. Sugiyama, *IEICE Trans. Inf. Syst.* **2016**, *99*, 176.
[40] M. C. Du Plessis, M. Sugiyama, *Neural Networks* **2014**, *50*, 110.
[41] D. Theng, K. K. Bhoyar, *Knowl. Inf. Syst.* **2024**, *66*, 1575.
[42] E. Hemphill, J. Lindsay, C. Lee, I. I. Măndoiu, C. E. Nelson, in *BMC bioinformatics*, vol. *15*, Springer, Berlin, Heidelberg **2014**, pp. 1–14.
[43] E. O. Abiodun, A. Alabdulatif, O. I. Abiodun, M. Alawida, A. Alabdulatif, R. S. Alkhawaldeh, *Neural Comput. Appl.* **2021**, *33*, 15091.
[44] N. Sánchez-Maroño, A. Alonso-Betanzos, M. Tombilla-Sanromán, in *Intelligent Data Engineering and Automated Learning - IDEAL 2007* (Eds: H. Yin, P. Tino, E. Corchado, W. Byrne, X. Yao), Springer, Berlin Heidelberg, Berlin **2007**, pp. 178–187.

[45] R. P. Feynman, *Int. J. Theor. Phys.* **1982**, *21*, 6.
[46] Y. I. Manin, *Computable and Uncomputable*, Sovetskoye Radio, Moscow **1980**.
[47] Z. Wang, *Highl. Sci., Eng. Technol.* **2023**, *38*, 370.
[48] M. Ruggenthaler, D. Sidler, A. Rubio, *Chem. Rev.* **2023**, *123*, 11191.
[49] R. Kuang, M. Perepechaenko, *EPJ Quantum Technol.* **2022**, *9*, 26.
[50] S. M. Pillay, I. Sinayskiy, E. Jembere, F. Petruccione, *Adv. Quantum Technol.* **2024**, *7*, 2300249.
[51] M. Nivelkar, S. G. Bhirud, *J. Phys.: Conf. Ser.* **2022**, *2161*, 012023.
[52] V. Bhagwani, V. Mishra, *Interantional Journal of Scientific Research in Engineering and Management* **2023**, https://doi.org/10.55041/IJSREM17616.
[53] S. Bose, A. Mazumdar, S. Martine, M. Toroš, *Phys. Rev. D* **2022**, *105*, 106028.
[54] A. Vesperini, G. Bel-Hadj-Aissa, R. Franzosi, *Sci. Rep.* **2023**, *13*, 2852.
[55] M. Carlesso, S. Donadi, L. Ferialdi, M. Paternostro, A. Bassi, *Nat. Phys.* **2022**, *18*, 243.
[56] M. Bild, M. Fadel, Y. Yang, U. v. Lüpke, P. Martin, A. Bruno, Y. Chu, *Science* **2023**, *380*, 274.
[57] E. Jorge, A. Delgado, S. P. Walborn, *Quantum* **2023**, *7*, 945.
[58] A. Soiguine, *J. Appl. Math. Phys.* **2023**, *11*, 448.
[59] V. Montenegro, G. S. Jones, S. Bose, A. Bayat, *Phys. Rev. Lett.* **2022**, *129*, 120503.
[60] N. E. Abari, A. A. Rakhubovsky, R. Filip, *New J. Phys.* **2022**, *24*, 113006.
[61] 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, J. M. Gambetta, Quantum computing with Qiskit, **2024**, https://doi.org/10.48550/arXiv.2405.08810.
[62] Ibm quantum, https://quantum.ibm.com/ (accessed: January 2021).
[63] Y.-H. Liu, Z.-D. Qi, Q. Liu, *Sci. Rep.* **2022**, *12*, 7776.
[64] A. Carbone, D. E. Galli, M. Motta, B. Jones, *Symmetry* **2022**, *14*, 624.
[65] A. Sharma, P. K. Mishra, *International Journal of Information Technology* **2022**, *14*, 1949.