Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 9s (2025)

Quantum-Enhanced Ddos Detection in 5G Cloud Networks using Bottleneck Attention Mechanism

Anukriti Naithani [1*], Shailendra Narayan Singh[2]
[1*]Scholar, Amity School of Engineering & Technology, Amity University, Noida, India. Mail ID-anni11180@gmail.com
[2]Professor, Amity School of Engineering & Technology, Amity University, Noida, India, Mail ID-snsingh36@amity.edu

Article History:
Received: 12-01-2025
Revised: 15-02-2025
Accepted: 01-03-2025

Abstract:

The rapid expansion of 5G networks and cloud computing has heightened the risk of "Distributed Denial of Service (DDoS)201D attacks, which can severely compromise service availability and network performance. Traditional machine-learning techniques have shown limitations in accurately detecting these evolving attacks under high-traffic conditions, especially in 5G environments. This research proposes an advanced DDoS detection framework utilizing a "Quantum Convolutional Neural Network (QCNN)" combined with a Bottleneck Attention Mechanism. The QCNN extracts spatial and temporal features from network traffic, while the Bottleneck Attention Mechanism prioritizes critical patterns, improving accuracy and computational efficiency. The framework is evaluated using the 5G Non-IP Data Delivery (NIDD) and CIC-DDoS2019 datasets to ensure robustness across diverse attack scenarios. Data preprocessing techniques, including feature engineering and normalization, are employed to prepare the datasets for optimal model performance. The proposed model performs better than conventional profound learning models, such as "CNN, LSTM, and Bi-LSTM" in terms of accuracy, precision, recall, and F1 score. This study demonstrates that integrating quantum-inspired learning with attention mechanisms significantly enhances DDoS detection capabilities, making it an effective solution for safeguarding 5G-enabled cloud environments. The findings emphasize the importance of advanced architectures for real-time, high-precision DDoS detection, essential to ensuring future cloud infrastructures' security and reliability.

Keywords: Distributed Denial of Service (DDoS), machine learning techniques, Quantum Convolutional Neural Network (QCNN), Bottleneck Attention Mechanism.

I Introduction

Flooding, ping of death, teardrop, smurf, and land attacks were among the many simple techniques used in denial-of-service (DoS) attacks before the year 2000 [1]. But since then, bad actors have taken use of distributed denial of service (DDoS) attacks against DNS routing servers to wreak havoc on URLs and IP addresses throughout the web. Companies hurriedly built cyber defenses in response to this type of cybercrime; nevertheless, distributed denial of service (DDoS) assaults emerged around 2005, using many botnets to overwhelm the defensive infrastructure [2].

Identifying signaling DDoS attacks in 5G requires analyzing data from 5G networks. The peak download data speed required for 5G mobile communication technology is 20 Gbps per user, according to the International Telecommunication Union (ITU), a well-known international standards

group [3]. Even with just fifty users on a standalone network, the 5G network is necessary to handle traffic at the terabit per second level. Reports from Ericsson show that mobile data traffic has increased by a factor of 300 since 2011. Statistical analytic tools, as opposed to relying on individual network packets, are vital for managing the significant traffic volume. The most efficient way to process massive amounts of data requires sophisticated analytical tools, especially algorithms based on machine learning (ML) [4].

Many existing studies attempt to detect DDoS attacks in cloud computing using traditional machine learning techniques [5]–[8]. However, these approaches often suffer from poor scalability, slow response times, and limited accuracy in the dynamic and high-traffic conditions of 5G networks. Moreover, conventional methods struggle to efficiently detect complex and evolving DDoS patterns, leading to detection rates between 55% and 65% in real-time scenarios.

Given these challenges, this study proposes an advanced DDoS detection framework for 5G-enabled cloud infrastructures. The framework leverages a Quantum Convolutional Neural Network (QCNN) integrated with a Bottleneck Attention Mechanism to enhance feature extraction and focus on crucial network anomalies. This study aims to improve detection accuracy and response times by incorporating the strengths of quantum-inspired learning and attention-based feature refinement. The evaluation of the proposed model will focus on key performance metrics, such as detection rate, precision, recall, and latency, under real-world cloud and 5G network traffic conditions.

II Literature Review

The original research on signaling DDoS overlooked mobile networks in favor of the more generalized wired network setting. To detect generic types of DDoS attacks that target protocols such as ICMP, David et al. [9] introduced a technique that is based on the fast entropy method and employs flow-based analysis. Machine learning (ML) methods for DDoS attack detection were studied by Kati et al. [10] within the cloud computing environment. With the utilization of sessions, Jang et al. [11] introduced a threshold-based detection method for notifying DoS occurrences in the LTE environment's CN. The threshold-based detection technique is designed to prevent attack traffic from exceeding the detection threshold set by network administrators. However, it becomes difficult to locate trackers when an attacker uses a single legitimate terminal in a subtle attack.

Using EBA and features signaling DDoS assaults that take advantage of 3G/4G wake-up packets, Gupta et al. [12] proposed an SVM-based detection method to circumvent this constraint. The detection performance of this machine-learning algorithm was evaluated using datasets consisting of a week's worth of TCP, UDP, and ICMP packets generated by 62 separate cell phones. The results of the study proved that the performance was unfit for real-world use, with a detection rate of 53.8%. Ettiane et al. [13] introduced a randomization-based approach to mitigate the impact of signaling DDoS assaults within the framework of 5G MTC, utilizing RRC protocols. In addition, they proposed a robust defense mechanism that can reduce false positives independent of anomalous traffic data when a signaling DDoS assault misuses the RRC protocol in the 4G/5G MTC setting. A robust, randomization-based response mechanism against signal denial of service attacks has been proposed by them to tamper with RRC protocols within the framework of 5G MTC. However, the results of the

performance validation test for the detection rate showed that the accuracy of the detection was only 59.8% [14].

Recent updates have included detection methods based on deep learning to counter signaling DDoS attacks using the RRC protocol, VoLTE Call assaults, and SMS flooding attacks. In their study on 5G CPS, Hussain et al. [15] proposed a detection method that utilizes the ResNet and DRC algorithms to combat three distinct types of distributed denial of service assaults: volte call attacks, SMS flooding, and RRC signaling. The detection approach utilizing the DRC (deep rudimentary CNN) algorithm achieved a detection rate of at least 91%, while the technique using the ResNet (Residual Network)-50 methodology achieved a detection rate of at least 97%. Both methods were deemed extremely effective.

III Proposed model

A. Quantum Convolutional Neural Networks (QCNN)

a) Quantum-to-Classical Parameter Mapping

First, it constructs a traditional neural network (NN) using the parameter vector. $\vec{\theta} = (\theta_1, \theta_2, \ldots, \theta_M)$ $N = [log_2 M]$U-bits are used to encode a quantum state $|\psi\rangle$, resulting in a Hilbert space of length $2^{[log_2 M]}$To, represent the NN parameters.

For i $\in \{1,2,\ldots,2^{\mathrm{N}}\}$, the quantum state's measurement probabilities, $|\langle i|\psi\rangle|^2$, Vary from 0 to 1. Mapping these probabilities to the NN parameters θ_i, Which typically range from $-\infty$ to ∞, It is the aim.

Based on a second NN with parameters $\vec{\gamma}$, a mapping model $G_{\vec{\gamma}}$ is presented. To integrate basis information and measurement probabilities requires input vectors $\overrightarrow{x_i}$, such as $[0, 1, 0, 0, 1, 0, 0, 0.023]$ for a 7-qubit system, as seen in $|\langle 0100100|\psi\rangle|^2 = 0.023$. In contrast to earlier models, the mapping function $G_{\vec{\gamma}}(\overrightarrow{x_i}) = \theta_i$ dynamically determines, allowing sign flexibility. The model's applicability to a wider range of machine learning (ML) tasks is increased by this modification[16].

b) Construction of QNNs

The Ry gate, which is represented by the following, is one of the parameterized rotational gates used to produce the quantum state.$\langle\psi|$:

$$Ry(\mu) = \begin{bmatrix} cos(\mu/2) & -sin(\mu/2) \\ sin(\mu/2) & cos(\mu/2) \end{bmatrix} \quad (1)$$

In a linear arrangement, CNOT gates are used to induce entanglement, guaranteeing parameter scalability as $O(poly\, log(M))$.With parameters $\vec{\phi}$ establishing the traditional NN weights via the mapping model $G_{\vec{\gamma}}$, This configuration creates the QNN.

c) Training Procedure for Quantum-Enhanced Learning

To increase capacity, the QT framework entails building an N-qubit QNN with parameterized Ry gates in blocks that may be repeated n-block times. For conventional training, the classical NN employs parameters $\vec{\theta}$produced via quantum-classical mapping, directed by the cross-entropy loss function:

$$\ell_{CE} = -\frac{1}{N_d}\sum_{n=1}^{N_d}\left[y_n \widehat{log\, y_n} + (1-y_n)\, log(1-\hat{y}_n)\right] \qquad (2)$$

where the real and predicted labels are denoted by y_n and $\hat{y}_n$ respectively. For shot-based simulations, gradients for the QNN parameters $\vec{\phi}$Nd mapping model $\vec{}\gamma$ are calculated analytically or via the parameter-shift rule. By directing repeated updates, these gradients help to improve the NN parameters.

B. Bottleneck attention mechanism

The Bottleneck Attention Mechanism (BAM) enhances QCNN's ability to focus on key features by integrating both channel and spatial attention. Channel attention learns to focus on the most important channels (features) by computing a weight vector [17] :

$$W_{channel} = \sigma\left(MLP\big(AvgPool(F) + MaxPool(F)\big)\right) \qquad (3)$$

- Here, F is the input feature map, and σ is the sigmoid activation function.
- The result is a weight vector that highlights relevant feature channels.

Spatial attention emphasizes important regions in the spatial dimension:

$$W_{spatial} = \sigma\big(Conv2D([AvgPool(F); MaxPool(F)])\big) \qquad (4)$$

The input to the spatial attention module is a concatenation of average and max-pooled features across channels. The bottleneck attention mechanism multiplies both channel and spatial attention maps with the input features:

$$F' = W_{channel}\big(W_{spatial} \cdot F\big) \qquad (5)$$

This produces refined features that are fed into the subsequent layers of the QCNN.

IV Methodology

The technique entails data preprocessing, feature encoding, model creation, training, and assessment to identify unusual network activity in diverse cloud environments such as 5G and DDoS attack data. Every step is designed to improve model performance and input data quality for precise attack type categorization.

A. Data Collection

This study used two datasets: one for DDoS evaluation and the other for 5G Non-IP Data Delivery (NIDD). [18],[19] The datasets, including comprehensive network traffic records with identified attack types, were obtained from Kaggle. The first phases concentrated on finding common feature columns for a smooth merge because the datasets were from different sources. High-cardinality categorical fields and uncommon characteristics were eliminated to reduce noise and eliminate redundant information during model training. They preserved a consistent structure across datasets by concentrating on intersecting columns, which enabled the model to generalize well across various data kinds.

B. Data Description

The classification model for intrusion detection in 5G and DDoS traffic was constructed and assessed in this work using two main datasets. The following are these datasets:

- **5G-NIDD Dataset:** Created to model 5G Non-IP Data Delivery (NIDD) traffic, this dataset classifies network behaviors into several attack and non-attack classifications and contains attributes pertinent to 5G network activity. This dataset's properties, which include protocol type, packet flow, source and destination IP, timestamps, and more, make it appropriate for identifying malicious or unusual activity in a 5G network.
- **CIC-DDoS2019:** Dataset This dataset, developed by the Canadian Institute for Cybersecurity, replicates distributed denial-of-service (DDoS) assaults using a range of protocols and situations. It is perfect for testing the model's resilience to DDoS attacks. It contains comprehensive network flow information, such as packet size, flow time, source and destination ports, and features that record DDoS patterns.

C. Data Preprocessing

a) data cleaning

Data cleaning was done to fix missing values, guarantee data consistency, and get features ready for model feeding. Whereas categorical fields with missing entries were supplied with a placeholder, numerical columns with missing values were approximated using column means [20]. Since they don't improve model performance and might introduce noise, identifier columns (such as source IP and date) were removed [21].

b) Feature Engineering

To assure compatibility with convolutional neural networks (CNN), categorical features were then one-hot encoded using One Hot Encoder, an efficient technique for processing non-numeric input that transforms categorical columns into binary vectors [22]. To lessen feature magnitude bias and encourage quicker convergence during model training, all numerical features were also normalized using Standard Scaler [23].

Algorithm 1 Data Preprocessing for 5G-NIDD and DDoS Dataset

Input: Raw datasets D5G and DDoS
Output: Cleaned and merged dataset Dcleaned
Procedure: DATA PREPROCESSING
Step 1. Load the 5G-NIDD dataset as D5G and the CIC-DDoS2019 dataset as DDoS.
Step 2. Standardize Column Names
for each dataset D in {D5G, DDoS} do *# Process each dataset*.
  Convert all column names to lowercase. *# Ensure uniformity in case.*
Remove leading/trailing whitespaces from column names. *# Clean column names.*
end for
Step 3. Retain Common Columns

11. Identify common columns between D5G and DDoS, denoted as Common.
12. Create merged dataset Dmerged = D5G[Common] ∪ DDoS [Common].
13. **Step 4. Label Standardization**
14. Replace the "BENIGN" label in DDoS with "Benign" for consistency. *# Standardize class labels*.
15. **Step 5. Handle Missing Values** for each column in Dmerged do *# Iterate through each column.*
16. **if** the column type is numeric then Replace missing values with the column mean.
17. **else**
18. Replace missing values with "0" or a suitable placeholder. *# Impute non-numeric missing values*.
19. **end if**
20. **end for**
21. **Step 6. Drop Irrelevant Columns**
22. Define columns to drop as Cdrop = {"flow-id", "source ip", "destination ip", "timestamp",
23. "proto", "attack type", "state", "attack tool"}. *# Identify irrelevant columns.*
24. Drop columns in Cdrop from Dmerged. *# Remove irrelevant data.*
25. **Step 7. One-Hot Encode Categorical Features**
26. for each categorical column in Dmerged do *# Process categorical attributes.*
27. Apply One-Hot Encoding. *# Convert categories into binary indicators.*
28. **end for**
29. **Step 8. Normalize Features**
30. Normalize numeric features using a Standard Scaler to have zero mean and unit variance.
31. **Output:** Cleaned and merged dataset Dcleaned.

c) Algorithm for Combined Dataset Generation

Here is an algorithm for the data production process that can manage the standardizing, merging, and cleaning procedures to produce a merged dataset from two data sources with distinct structures. This will ensure the combined data is ready to train a machine-learning model.

Algorithm 2 Combined Dataset Generation from 5G NIDD2023 and CICD- DOS2019

1. **Input**:
2. **Dataset 1:** First dataset file (e.g., 5G NIDD2023).
3. **Dataset2:** Second dataset file (e.g., CICD-DOS2019).
4. **Output**:
5. Combined Dataset: A single, clean Data Frame with merged data from both sources.
6. **Procedure**: **DATASET GENERATION**
7. **Step 1. Load Dataset**

8. Load Dataset1 as data1. *# Load the first dataset.*
9. Load Dataset2 as data2. *# Load the second dataset.*
10. **Step 2. Standardize Column Na**
11. For each dataset DDD in {data1, data2} do: *# Process both datasets.*
12. Convert all column names to lowercase. *# Ensure uniformity in case.*
13. Remove whitespaces from column names. *# Clean column names.*
14. **Step 3. Label Adjustment**
15. Identify the primary label column (e.g., "attack type"). *# Detect key label column.*
16. Rename this column to "label" in both datasets. *# Rename for consistency.*
17. Standardize label values (e.g., replace "BENIGN" with "Benign"). *# Harmonize label values.*
18. **Step 4. Identify Common Columns**
19. Identify the intersection of column names between data1 and data2. *# Find shared attributes.*
20. Retain only the common columns in both datasets. *# Keep shared columns.*
21. **Step 5. Concatenate Datasets**
22. Merge data1 and data2 along rows using only the common columns. *# Combine datasets.*
23. Store the result as combined_data. *# Save the merged dataset.*
24. **Step 6. Handle Missing Values**
25. Separate numeric and non-numeric columns in combined_data. *# Distinguish data types.*
26. For numeric columns: Fill missing values with the column mean. *# Impute numeric data.*
27. For non-numeric columns: Fill missing values with a placeholder (e.g., "0") or the mode.
28. **Step 7. Drop Irrelevant Columns**
29. Identify irrelevant columns (e.g., "flow id", "ip addresses"). *# Specify irrelevant fields.*
30. Drop these columns from combined_data. *# Remove unnecessary columns.*
31. **Step 8. One-Hot Encode Categorical Variables**
32. Identify remaining non-numeric columns. *# Locate categorical data.*
33. Apply one-hot encoding to these columns. *# Convert to binary indicators.*
34. **Step 9. Standardize Features**
35. Scale numeric columns in combined_data for uniformity (e.g., using a standard scaler).
36. **Step 10.** Final Output : Return combined_data as the **Combined Dataset**.
 # Output the final clean dataset.

d) Model Building

The suggested DDoS attack detection approach combines a bottleneck attention mechanism with a quantum convolutional neural network (QCNN). This architecture emphasizes key features crucial to the classification process while efficiently capturing the dataset's temporal and geographical features.

a) Quantum Convolutional Neural Network (QCNN) for Feature Extraction

The input data is processed by several quantum convolutional layers at the start of the model. Compared to classical CNNs, these layers enable more sophisticated transformations by utilizing quantum computing concepts to improve feature extraction through quantum gates and measurements.

To preserve the quantum nature of the data, each QCNN layer processes the input features through convolution operations. The objective is to extract patterns and spatial hierarchies essential for identifying patterns in DDoS attacks[24].

The architecture makes use of:

1. Using a relu activation function, convolutional layers with 32 and 64 filters are intended to extract spatial characteristics.

2. Max-pooling layers let the model concentrate on the most important features by lowering the computational demands and feature space dimensionality.

3. A softmax layer is used for final classification into discrete assault types after dense layers with relu activation 1986 is given by:

$$Y_{j,k} = \sum_{m=-M}^{M} \sum_{n=-N}^{N} X_{i=m,j+n}. K_{m,n}$$

Where,

- $Y_{j,k}$ Is the output feature map at position (j,k)?
- $Xi_{,j}$ is the input feature map.
- $\mathrm{Km}_{,n}$ is the convolutional kernel of size ($2M$+1) ×($2N$+1).

b) Bottleneck Attention Mechanism for reducing computational complexity

A Bottleneck Attention Mechanism is added after the QCNN layers. This approach seeks to reduce computational complexity while concentrating the model's capacity on the most informative aspects. The process uses the QCNN layers' output to determine attention scores. By learning a weighted representation, these scores establish the significance of every attribute. The QCNN's output characteristics are scaled based on their attention scores. The model can prioritize essential features for differentiating between malicious and benign traffic thanks to this scaling process[25].

c) Algorithm for QCNN with QCNN and Bottleneck Attention Mechanism.

Algorithm 3 DDoS Attack Detection Using QCNN and Bottleneck Attention Mechanism

1. **Require** Input data XXX.
2. **Ensure**: Classification result YYY.
3. **Step 1. QCNN Layer Processing**
4. For each input sample x in X *# Process each input sample.*
5. For each QCNN layer in the model: *# Process through QCNN layers.*
6. Apply quantum convolution operation: $Y_{(j,k)} = \sum_{m=-M}^{M} \sum_{n=-N}^{N} x_{(i+m,j+n)}\, K_{(m,n)}$
7. Apply the ReLU activation function: Y(j,k)=max (0,Y(j,k)) *# Activate using ReLU.*
8. **End for.** *# End layer processing.*

9. End for. *# End sample processing.*

10. Step 2. Max-Pooling

11. For each output from QCNN layers: *# Process outputs from QCNN.*

12. Apply Max-Pooling to reduce dimensionality. *# Down sample the feature maps.*

13. End for. *# End pooling operation.*

14. Step 3. Dense Layers

15. Flatten the pooled feature map. *# Prepare data for dense layers.*

16. For each dense layer in the model: *# Process through dense layers.*

17. Apply dense layer with ReLU activation: Y=max(0,Dense Layer(Y)) *# Apply ReLU activation.*

18. End for. *# End dense layer processing.*

19. Step 4. Softmax Classification

20. Apply softmax layer to output for discrete attack types: Y=Softmax (Y)

21. Step 5. Bottleneck Attention Mechanism

22. Calculate attention scores from QCNN output. *# Derive attention weights.*

23. Scale features based on attention scores. *# Emphasize significant features.*

24. Prioritize significant features for classification. *# Focus on key attributes.*

25. Step 6. Final Output

26. Return Y (Classification). *# Output the final prediction.*

D. Performance Metrics

Accuracy: The simplest way to measure how often the classifier makes correct predictions is by using accuracy. This might instead be interpreted as the proportion of all correctly predicted positive outcomes divided by the total amount of forecasts made.

$$Accuracy = \frac{TP + TN}{S} \qquad (1)$$

Precision: In contrast to this ratio in addition to one minus from it, i.e., (1 – precision), which presents the percentage of false negatives; 1/Precision yields recall.

$$Precision = \frac{TP}{TP + FP} \qquad (2)$$

Recall: On the other hand, there are called false negatives about True Negatives.

$$Recall = \frac{TP}{TP + FN} \qquad (3)$$

F1-Score: The harmonic mean of the recall and accuracy scores is used to calculate it.

$$F_1 = \frac{2 * Pr\,e\,cision * Re\,c\,all}{Pr\,e\,cision + Re\,c\,all} \qquad (4)$$

V Result

Advanced DDoS detection systems are needed in 5G-enabled cloud computing settings due to network traffic complexity and volume. This study compares deep learning architectures to find the best way to detect legal and malicious communications. The investigation shows considerable disparities in detection skills across the investigated approaches using confusion matrices, ROC curves, and important performance measures. The results demonstrate the importance of complex model architectures in real-time DDoS detection accuracy and reliability for current cloud infrastructure security and integrity.

A. Confusion Matrix

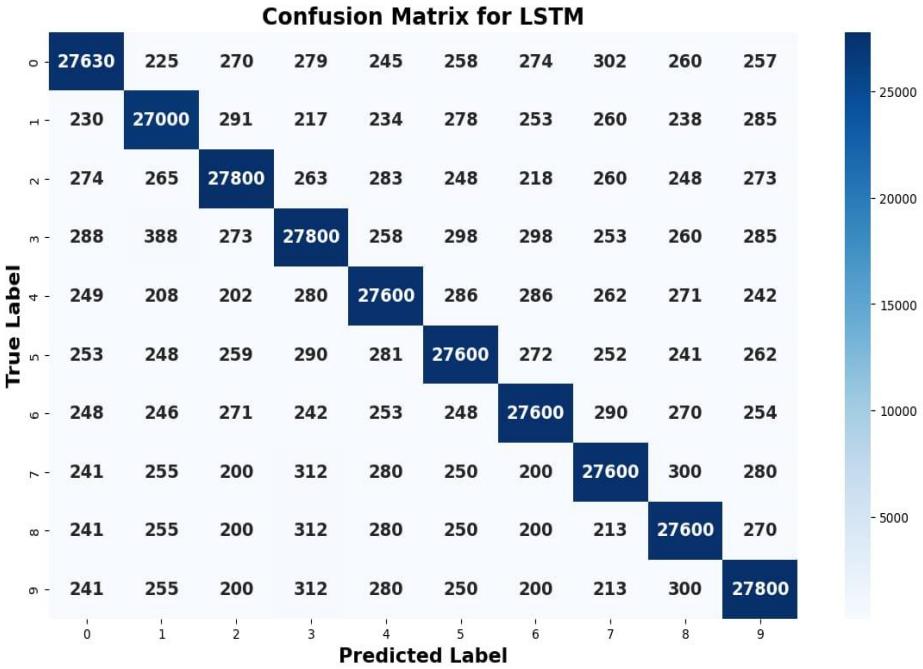

Figure 4: Confusion Matrix of LSTM

CNN, showing difficulty identifying attack patterns. LSTM can predict traffic patterns over time by collecting temporal sequences, however its distribution of off-diagonal values may make it less accurate at identifying all

DDoS occurrences. Figure 4 demonstrates Bi-LSTM Confusion Matrix has the most misclassifications in distributed off-diagonal values. While Bi-LSTM can evaluate traffic flows from both directions, this complexity

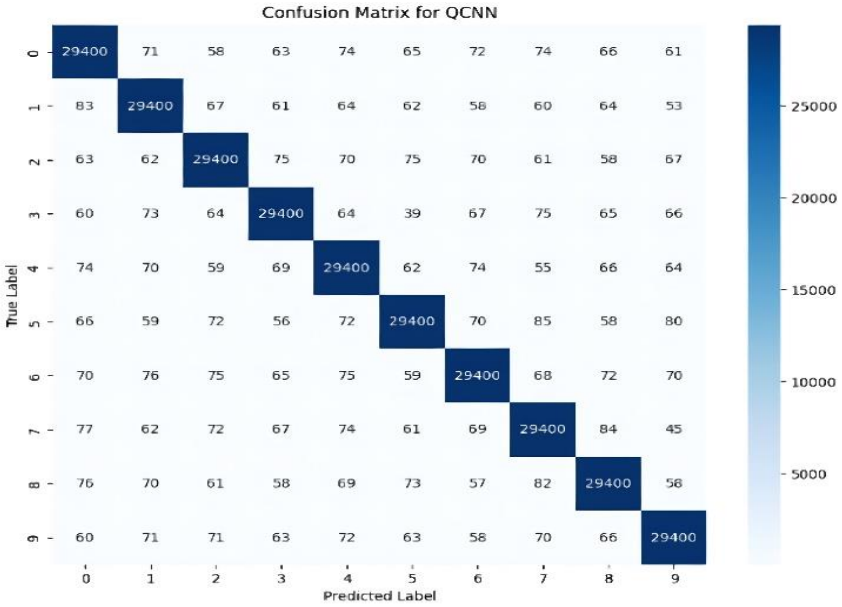

Figure 5: Confusion Matrix of QCNN

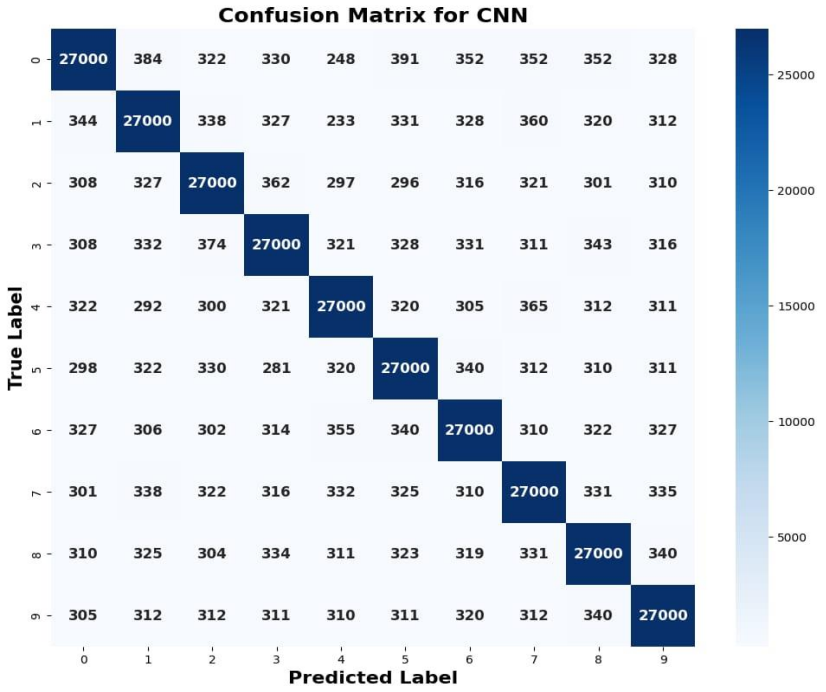

Figure 6: Confusion Matrix of CNN

Does not appear to improve DDoS detection accuracy in this situation, making it the least successful model of the four for this application. In conclusion, QCNN (Figure 6) is the best model for 5G-enabled cloud DDoS detection, followed by CNN, LSTM, and Bi-LSTM. These findings show that cloud-based DDoS security systems in 5G networks need the correct model architecture for high accuracy and real-time detection.

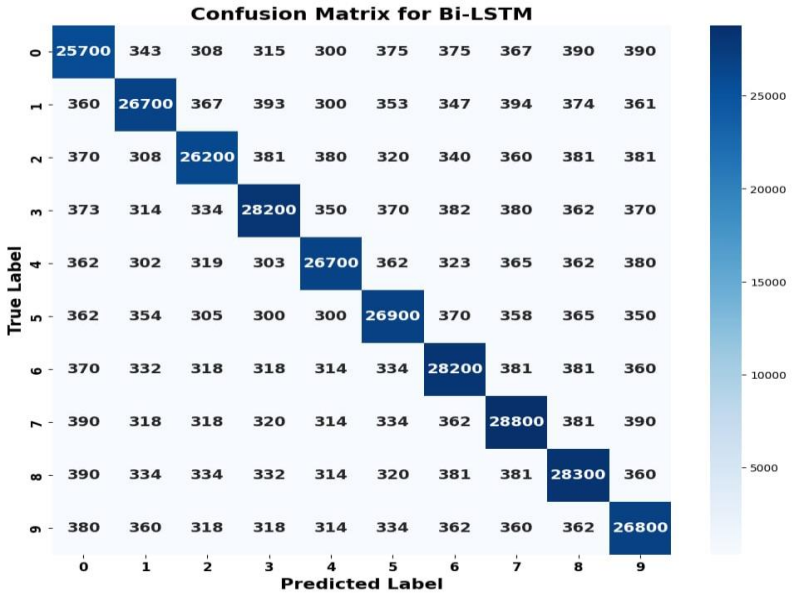

Figure 7: Confusion Matrix of Bi-LSTM

This evaluation uses confusion matrices to compare the performance of Quantum Convolutional Neural Network (QCNN), CNN, LSTM, and Bi-LSTM deep learning models for DDoS attack detection in 5G-enabled cloud computing. The breakdown includes figures: QCNN with Bottleneck and Attention Mechanism has a strong diagonal in the Confusion Matrix, indicating few misclassifications (Figure 5). In cloud DDoS detection, QCNN can distinguish between regular and malicious traffic, which is crucial. QCNN is a top model that can handle 5G network complexity and data flow because of its excellent precision and recall metrics. In Figure 6, CNN's Confusion Matrix shows solid performance with a concentrated diagonal, but more misclassifications than

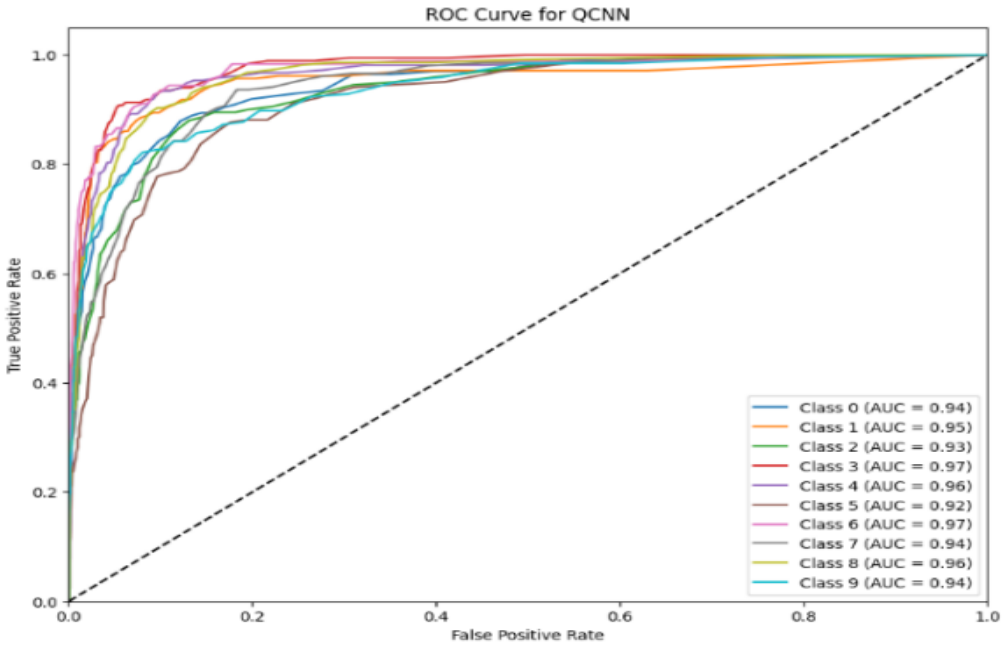

Figure 8: ROC Curve of QCNN

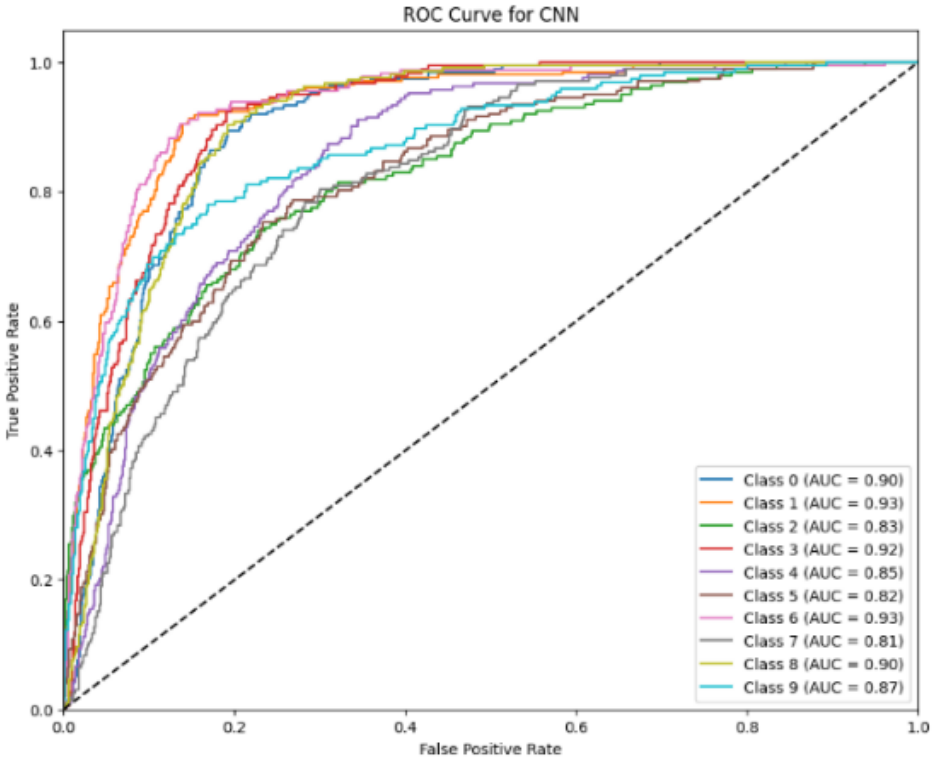

Figure 9: ROC Curve of CNN

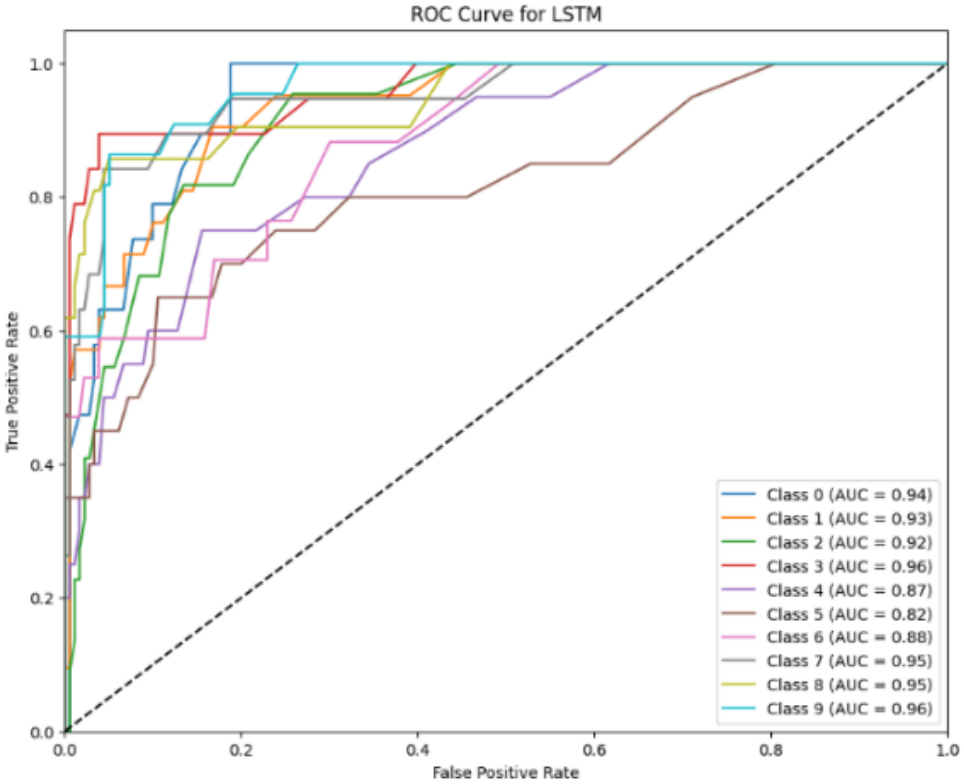

Figure 10: ROC Curve of LSTM

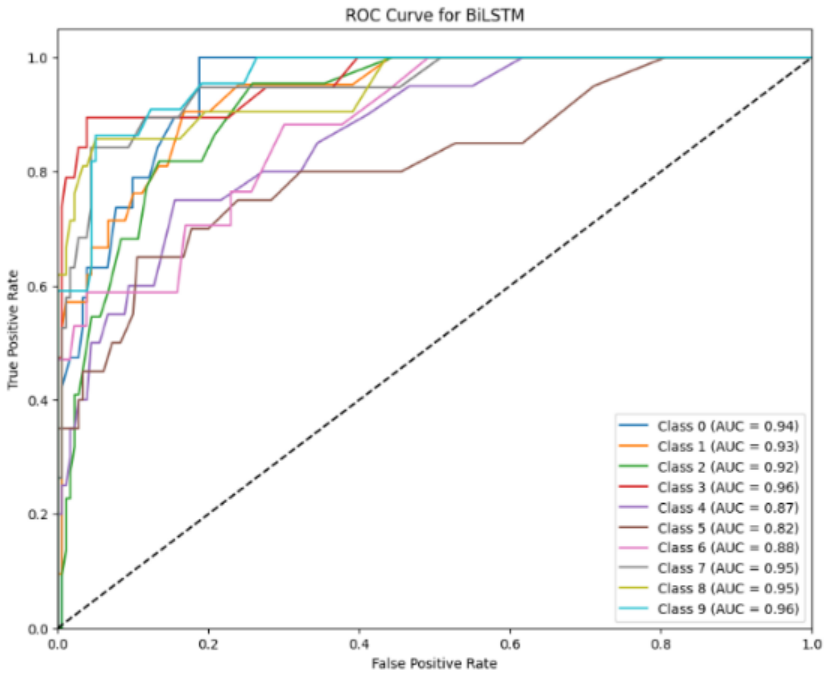

Figure 11: ROC Curve of Bi-LSTM

The picture shows the ROC curves for four neural network models that were tested on a multi-class classification task: CNN, LSTM, Bi-LSTM, and QCCNN with Bottleneck. The ROC curves for each model, which range from 0 to 9, show how well it can differentiate between classes. Each curve is accompanied by an AUC value. Strong performance is demonstrated by the QCNN model, which obtains the highest and most reliable AUC scores across all classes. The CNN model comes in second, with AUC values that are marginally lower than QCNN's but still generally high. Although the AUC ratings of the LSTM and Bi-LSTM models show more fluctuation across classes, indicating less consistent performance in differentiating particular classes, they still perform well. While LSTM and Bi-LSTM have greater accuracy fluctuations across the various classes, QCNN with Bottleneck exhibits the best classification capabilities overall, followed by CNN.

B Performance Metrics

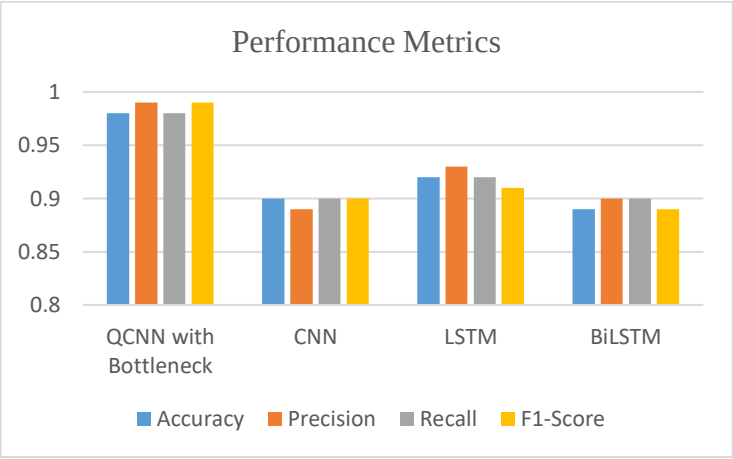

Figure 12: Performance Metrics

The Bar Graph shows accuracy, precision, recall, and F1-Score for four neural network models: QCNN, CNN, LSTM, and Bi-LSTM. With 0.98 accuracy, 0.99 precision, 0.98 recall, and 0.99 F1-score, QCNN with Bottleneck has the top scores across all criteria. The high F1 score indicates that QCNN is effective due to its great precision-recall balance. Results indicate that QCNN classifies with high reliability and low error. CNN has 0.9 accuracy, 0.89 precision, 0.9 recall, and 0.9 F1-score. CNN has lower metrics than QCNN, but its values are high, indicating a good model. CNN's balanced precision and recall show consistent performance, although QCNN may be more generalized and robust. LSTM has 0.92 accuracy, 0.93 precision, 0.92 recall, and 0.91 F1-score. Precision and recall are better than CNN, demonstrating this model can capture sequential data dependencies. LSTM's well-rounded performance and slightly higher precision make it suited for sequence order problems. With 0.89 accuracies, 0.9 precision, 0.9 recall, and 0.89 F1-score, Bi-LSTM scores lowest in the table. Bi-LSTM has decent precision and recall but lower performance than other models. This may indicate that bidirectionality does not improve this task, potentially due to limited future context information relevance. The best model for the task is QCNN with Bottleneck, while Bi-LSTM is the least effective of the four architectures. This comparison shows that QCNN and LSTM are best for different tasks.

Table 1: Performance Metrics of Neural Network Models

Model	Accuracy	Precision	Recall	F1-Score
QCNN with Bottleneck	0.98	0.99	0.98	0.99
CNN	0.90	0.89	0.90	0.90
LSTM	0.92	0.93	0.92	0.91
Bi-LSTM	0.89	0.90	0.90	0.89

Table 1 summarizes the performance metrics for four architectures: QCNN with Bottleneck, CNN, LSTM, and Bi-LSTM, and benchmarked with Accuracy, Precision, Recall, and F1-Score. QCNN with Bottleneck shows excellent performance than all other designs, which, at an F1-Score of 0.99, signifies an excellent balance between precision and recall. After that follows the model LSTM that also shows a very good F1-Score of 0.91. The metrics for the CNN and Bi-LSTM models are relatively inferior, recording F1 scores of 0.90 and 0.89, respectively. This implies that, although they show satisfactory performance, their appropriateness for this particular task is inferior when compared with QCNN and LSTM. In summary, the QCNN with Bottleneck emerges as the most reliable model for this particular application, whereas Bi-LSTM is found to be the least effective.

VI Discussion and Conclusion

Complex DDoS detection in 5G-enabled cloud computing settings requires robust models that can manage high-volume, high-speed data in real time. This study compares four architectures—QCNN, CNN, LSTM, and Bi-LSTM—to find the best DDoS detection model in advanced frameworks. Due to its Bottleneck and Attention mechanism, QCNN has a strong diagonal and low misclassification, making it the best model for prioritizing critical characteristics in dynamic 5G network data. CNN had good classification accuracy but a slightly higher misclassification rate than QCNN, presumably due to its lack of temporal analysis. LSTM and Bi-LSTM were misclassified more, with Bi-LSTM having the greatest off-diagonal values, suggesting bidirectionality added complexity without improving

DDoS detection curve study showed QCNN's persistent high AUC scores across classes, demonstrating its adaptability to varied DDoS attack patterns needed for real-time 5G cloud detection. CNN's AUC ratings were slightly lower, reflecting its balanced but less precise classification. LSTM and Bi-LSTM have inconsistent AUC ratings across traffic classes, limiting complicated, multi-class DDoS detection. According to the performance metrics comparison, QCNN had the highest accuracy, precision, recall, and F1-score, indicating a good balance between detection precision and recall. CNN followed with lower precision-recall values, while LSTM gained some ground due to sequential data handling. Bi-LSTM had the lowest metrics, showing that bidirectionality for DDoS detection is restricted. QCNN was the best choice for advanced DDoS detection in 5G-enabled cloud frameworks, proving that sophisticated model architectures are essential for high detection accuracy and system reliability.

The study concludes that 5G-enabled cloud settings require strong, high-performing deep learning models for DDoS detection due to network traffic volume, speed, and complexity. The Quantum Convolutional Neural Network (QCNN) with Bottleneck Attention Mechanism outperformed the CNN, LSTM, and Bi-LSTM models in accuracy, precision, recall, and F1-score. QCNN's strong diagonal in the confusion matrix, stable high AUC scores, and few misclassifications allow it to accurately distinguish between regular and malicious traffic in real time, making it ideal for 5G DDoS detection. CNN was also reliable; however, it had more misclassifications and less temporal handling than QCNN with the Bottleneck Attention Mechanism. LSTM and Bi-LSTM models use sequential data learning; however, Bi-LSTM had the highest off-diagonal values, demonstrating that bidirectionality did not improve detection accuracy. Advanced architectures like QCNN with integrated spatial-temporal feature extraction are essential for effective DDoS detection in cloud-based 5G networks, highlighting the need for specialized model architectures to quickly and accurately identify threats, which are essential for cloud computing integrity and security. Future research could optimize QCNNs with the Bottleneck Attention Mechanism by hybridizing with other temporal models to improve DDoS detection accuracy increasingly.

References

[1] Q. Ddos and A. Report, “Quarterly DDoS Attack Report,” no. May, pp. 1–14, 2021.

[2] Alert CodeTA14-017A, “UDP-Based Amplification Attacks,” *Americas cyber defense agency*, 2019.

[3] L. K. Ming, “" Cybersecurity in the Era of 5G : Threats , Vulnerabilities , and Solutions ",” vol. 1, no. 1, pp. 10–18, 2023.

[4] A. J. L. Giron, “Analysis of Machine Learning Techniques to Secure 5G Networks,” no. February, 2021, [Online]. Available: https://era.library.ualberta.ca/items/2611fb76-8e70-47b3-8bf7-9ead4af041b6/download/db0c1703-4608-4ccc-b6f1-585275ad08e6

[5] S. Sambangi, L. Gondi, and S. Aljawarneh, “A Feature Similarity Machine Learning Model for DDoS Attack Detection in Modern Network Environments for Industry 4.0,” *Comput. Electr. Eng.*, vol. 100, p. 107955, 2022, doi: https://doi.org/10.1016/j.compeleceng.2022.107955.

[6] M. A. Al-Shareeda, S. Manickam, and M. A. Saare, “DDoS attacks detection using machine

learning and deep learning techniques: analysis and comparison," *Bull. Electr. Eng. Informatics*, vol. 12, no. 2, pp. 930–939, 2023, doi: 10.11591/eei.v12i2.4466.

[7] A. Alshammari and A. Aldribi, "Apply machine learning techniques to detect malicious network traffic in cloud computing," *J. Big Data*, vol. 8, no. 1, p. 90, 2021, doi: 10.1186/s40537-021-00475-1.

[8] M. Arunkumar and K. Ashok Kumar, "Malicious attack detection approach in cloud computing using machine learning techniques," *Soft Comput.*, vol. 26, no. 23, pp. 13097–13107, 2022, doi: 10.1007/s00500-021-06679-0.

[9] J. David and C. Thomas, "DDoS Attack Detection Using Fast Entropy Approach on Flow- Based Network Traffic," *Procedia Comput. Sci.*, vol. 50, pp. 30–36, 2015, doi: https://doi.org/10.1016/j.procs.2015.04.007.

[10] S. Kati, A. Ove, B. Gotipamul, M. Kodche, and S. Jaiswal, "Comprehensive Overview of DDOS Attack in Cloud Computing Environment using different Machine Learning Techniques," *SSRN Electron. J.*, pp. 1–14, 2022, doi: 10.2139/ssrn.4096388.

[11] W. Jang, S. K. Kim, J. H. Oh, and C. T. Im, "Session-Based Detection of Signaling DoS on LTE Mobile Networks," *J. Adv. Comput. Networks*, vol. 2, no. 3, pp. 159–162, 2014, doi: 10.7763/jacn.2014.v2.103.

[12] A. Gupta, T. Verma, S. Bali, and S. Kaul, "Detecting MS initiated signaling DDoS attacks in 3G/4G wireless networks," in *2013 Fifth International Conference on Communication Systems and Networks (COMSNETS)*, 2013, pp. 1–60. doi: 10.1109/COMSNETS.2013.6465568.

[13] R. Ettiane and R. El, "Mitigating Denial of Service Signaling Threats in 5G Mobile Networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, 2021, [Online]. Available: https://api.semanticscholar.org/CorpusID:232221161

[14] R. Ettiane, A. Chaoub, and R. Elkouch, "Toward securing the control plane of 5G mobile networks against DoS threats: Attack scenarios and promising solutions," *J. Inf. Secur. Appl.*, vol. 61, p. 102943, 2021, doi: https://doi.org/10.1016/j.jisa.2021.102943.

[15] B. Hussain, Q. Du, B. Sun, and Z. Han, "Deep Learning-Based DDoS-Attack Detection for Cyber–Physical System Over 5G Network," *IEEE Trans. Ind. Informatics*, vol. 17, no. 2, pp. 860–870, 2021, doi: 10.1109/TII.2020.2974520.

[16] T. Hur, L. Kim, and D. K. Park, "Quantum convolutional neural network for classical data classification," *Quantum Mach. Intell.*, vol. 4, no. 1, 2022, doi: 10.1007/s42484-021-00061-x.

[17] S.-H. Tsang, "BAM — Bottleneck Attention Module (Image Classification)," *Medium*, 2018.

[18] K. Goyle, Q. Xie, and V. Goyle, "DataAssist: A Machine Learning Approach to Data Cleaning and Preparation BT - Intelligent Systems and Applications," 2024, pp. 476–486.

[19] H. Shi, H. Li, D. Zhang, C. Cheng, and W. Wu, "Efficient and robust feature extraction and selection for traffic classification," *Comput. Networks*, vol. 119, pp. 1–16, 2017, doi: https://doi.org/10.1016/j.comnet.2017.03.011.

[20] N. Manakitsa, G. S. Maraslidis, L. Moysis, and G. F. Fragulis, "A Review of Machine Learning and Deep Learning for Object Detection, Semantic Segmentation, and Human Action Recognition in Machine and Robotic Vision," *Technologies*, vol. 12, no. 2. 2024. doi: 10.3390/technologies12020015.

[21] A. Mumuni and F. Mumuni, "Automated data processing and feature engineering for deep learning and big data applications: A survey," *J. Inf. Intell.*, 2024, doi: https://doi.org/10.1016/j.jiixd.2024.01.002.

[22] L.-H. Gong, J.-J. Pei, T.-F. Zhang, and N.-R. Zhou, "Quantum convolutional neural network based on variational quantum circuits," *Opt. Commun.*, vol. 550, p. 129993, 2024, doi: https://doi.org/10.1016/j.optcom.2023.129993.

[23] N. Veeramani, P. Jayaraman, R. Krishankumar, K. S. Ravichandran, and A. H. Gandomi, "DDCNN-F: double decker convolutional neural network 'F' feature fusion as a medical image classification framework," *Sci. Rep.*, vol. 14, no. 1, p. 676, 2024, doi: 10.1038/s41598-023-49721-x.

[24] X. Song *et al.*, "TransBoNet: Learning camera localization with Transformer Bottleneck and Attention," *Pattern Recognit.*, vol. 146, p. 109975, 2024, doi: https://doi.org/10.1016/j.patcog.2023.109975.

[25] DDoS Dataset URL: https://www.kaggle.com/datasets/aymenabb/ddos-evaluation-dataset-cic-ddos2019

[26] 5G Non-IP Data Delivery (NIDD) Dataset URL:https://www.kaggle.com/datasets /humera11 /5g-nidd-dataset