Comparative study of QNN with CNN for human detection

Vansh Seth
Instruments Research and Development Establishment (IRDE)
Dehradun, India
sethvansh2004@gmail.com

***Abstract*—Human detection is an important task in computer vision, with a main application in, surveillance capabilities, autonomous systems, and safety monitoring. Traditional methods, specifically Convolutional Neural Networks (CNN), have been shown to be highly accurate for image based detection tasks because of their feature extraction capabilities and hierarchical learning. However, with the recent introduction of quantum computing technology, a Quantum Neural Network (QNN) could also perform human detection tasks while utilizing potentially faster computations of the technology and ability to learn using quantum entanglement and parallelism. This study presented the comparison of Convolutional Neural Networks (CNN) and Quantum Neural Networks (QNN) architectures used for human detection. The main metrics, accuracy, training time, model complexity and scalability were used. A classical convolutional neural network model was trained and evaluated alongside a QNN that models VQC circuits. Results indicated that while CNNs are still currently more robust in detection accuracy and robustness to standard benchmark datasets, QNNs might be effective in reducing development time and computational resource use with fewer training iterations, as quantum hardware improves and introduction becomes standard. The paper elaborates advantages and challenges of QNNs, and future vision opportunities to integrate quantum machine learning into vision-based human detection systems.**

***Index Terms*—Human Detection, Image Processing, Variational Quantum Circuits, Comparative Analysis, Model Performance Evaluation.**

I. Introduction

Over the last several years, machine learning (ML) has transformed many application domains, such as text analysis, speech recognition, image classification, object detection, and by processing real-time video [1,2,3,4,5]. Among all ML approaches, there are general machine learning, deep learning approaches with neural networks (DL), and most especially deep learning with Convolutional Neural Networks (CNNs) particularly for computer vision applications that have been transformative as they can automatically learn the hierarchical feature representation of raw input data [6,7,8]. CNNs have excelled in human detection, object tracking, and visual recognition (significantly outperforming active-handcrafted features) [9,10].

While convolutional neural networks (CNNs) have become the foundation of present-day visual recognition systems, the growing computational costs of deploying deep learning models have encouraged researchers to explore other computing paradigms. Quantum computing, an emergent field of technology, has the possibility to provide significant speed-ups for certain classes of problems leveraging quantum parallelism and entanglement [11,12]. The use of quantum computing in machine learning has resulted in a new area of research, called Quantum Machine Learning (QML), which includes Quantum Neural Networks (QNNs) that are novel architectures that are capable of encoding and processing data with quantum circuits [13,14].

QNNs presented via variational quantum circuits and hybrid quantum-classical algorithms can significantly lower training complexity and allow for higher-dimensional data spaces, while possibly achieving better generalization with fewer parameters than classical approaches [15,16]. Quantum hardware is still considered nascent, however, recent work has introduced QNNs successfully in the context of supervised learning, categorization, and pattern recognition [17,18]. Human detection as an application for QNNs has the potential to provide competitive results with CNN based current solutions, as it is relevant to surveillance, autonomous driving as well as safety monitoring.

Ability to detect humans is a complicated task as models need to identify human shapes, posture, and context within a wide range of backgrounds and lighting conditions [19]. Convolutional neural networks (CNN's) have produced high accuracies by automatically extracting hierarchies of spatial features through their convolutional layers, however, they also require large labeled datasets and more computational power, which can be restrictive in real-time or less-resourced situations [20,21]. Quantum neural networks (QNN's) may provide a quantum advantage, and address some of these issues, especially with continued development of quantum hardware and hybrid algorithms [22,23].

A number of reviews and comparative studies for either CNN developments [24], object detection systems [25], or emerging applications of quantum computing [26,27] have been established. At present there are no thorough studies that enable a distinct comparison between QNNs and CNNs in human detection tasks. Addressing this research gap is important. With this in mind this paper will present a thorough comparative study of QNNs and CNNs in the context of human detection -both QNNs and CNNs will be considered, in terms of accuracy, efficiency in training and resource utilization.

This paper provides the following contributions:

- We describe a comprehensive review of QNNs and

CNNs, outlining their architectures, ways of learning, and computer frameworks.
- We describe our implementation of both QNN and CNN models of human detection, we benchmark and analyze their performance considering various measures in order to compare their capabilities.
- We discuss the limitations and challenges of quantum machine learning in vision-based detection systems and future possibilities.
- We provide an outlook on the scalability and possible implementation of QNN-based models in light of quantum hardware development.

This paper is structured as follows. Section 2 presents related work and background of CNNs and QNNs; Section 3 describes the methodology and includes dataset, model architectures, and training; Section 4 describes the experimental results and comparative analysis; Section 5 discusses the key findings, challenges, and future possibilities; and Section 6 concludes the paper.

II. Related Work and Background

In this section we provide a comprehensive overview of the fundamental principles and recent advances in Convolutional Neural Networks (CNNs) and Quantum Neural Networks (QNNs), as they pertain to human detection. The section's contents will cover the following subsections:

A. Convolutional Neural Networks (CNNs)
B. Human Detection using CNNs
C. Quantum Neural Networks (QNNs)
D. Human Detection using QNNs

A. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a group of deep learning algorithms made to solve visual recognition problems. LeCun et al. [1] first introduced CNNs which have become perhaps the most common architectures used in the computer vision user community. The main benefit of CNNs is their ability to automatically learn representations from raw input data without requiring much manual extraction of features [6], [7].

CNNs have convolutional layers, pooling layers, and fully connected layers, where convolutional layers, or CNN layers, act as automatic feature extractors; extracting representations from raw data that generally contains spatial hierarchies based on learnable filters [2], [4]. There are several popular CNN architectures such as AlexNet [2] and VGGNet, GoogLeNet [4], ResNet [5], and YOLO [3] were able to achieve state-of-the-art in large scale object detection and classification tasks.

As shown in Figure 1, anterior CNN architectures consist of several important components:

- Convolutional Layers: These layers serve as automatic feature extractors and learn the spatial hierarchies from the original data through learnable filters.
- Pooling Layers: These layers minimize the spatial dimension of the feature maps, which is helpful for controlling overfitting and reducing computational complexity.

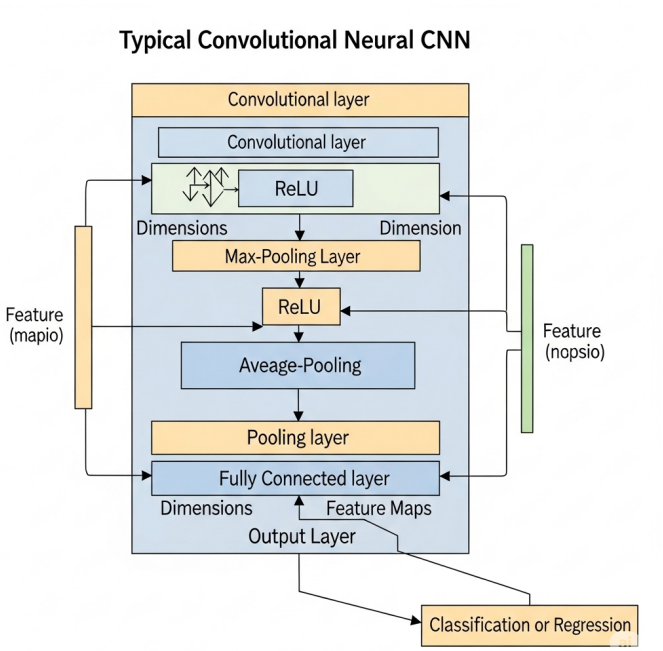

Fig. 1. CNN architecture

- Fully Connected Layers: These layers conduct high-level reasoning for the extracted features and typically lead to the final classification or regression output.

CNNs have revolutionized a number of domains including medical imaging analysis [21], action recognition [24], and graph-based learning in [25]. However, CNNs utilize considerable computational resources, especially in training deep architectures on large datasets.

B. Human Detection using CNNs

Detecting humans is a major problem in computer vision. Humans are typically detected as a component of larger applications ranging from surveillance to autonomous vehicles, and public safety. Prior to using CNNs, handcrafted features were used to detect humans, including Histograms of Oriented Gradients (HOG) [9] and Scale-Invariant Feature Transform (SIFT) [10]. The development of CNN-based approaches displaced these approaches due to significant improvements in accuracy and robustness over variations and environments [6], [19].

Modern object detection systems such as YOLO [3] and SSD accomplish real-time human detection by replacing the regional proposals and classification from a standard detection framework with a single unified detection network. Xu et al. [20] and Hossain et al. [19] provided a broad overview of CNNs based approaches to human detection, particularly important elements such as multi-scale detection, data augmentation, and large labelled datasets.

Many of the issues with detecting humans still exist among CNN based approaches, including high computational re-

quirements, needing extensive training datasets, and lack of generalization to complex environments [7], [20].

C. Quantum Neural Networks (QNNs)

Quantum Neural Networks (QNNs) are a new kind of machine learning model that are based on principles of quantum computing, such as superposition, entanglement, and quantum parallelism to process information [11], [12]. Unlike classical neural networks, QNNs operate using quantum bits (qubits) and quantum circuits, which has the potential to provide exponential improvements for certain learning tasks [15], [16].

QNNs usually employ variational quantum circuits (VQCs)- these are parameterized circuits that are trained with classical optimizers [14],[16] . As illustrated in Figure 2, which depicts a simple VQC, these circuits are central to many quantum machine learning models, including Quantum Support Vector Machines [15] and Quantum Convolutional Neural Networks [26]. These models demonstrate the feasibility of quantum-based learning, especially on current near-term quantum hardware.

The supposed advantages of QNNs are an improvement to computational efficiency relative to classical neural networks, the capacity to deal with more complex data spaces, and smaller models [11], [13], [14]. However, real-world applications are currently limited due to quantum hardware constraints, decoherence, and other noise [16], [22].

As seen in Figure 2, a VQC is composed of a series of quantum gates that are applied to a group of qubits (indicated by horizontal lines). The main components that we want to point out in this image are:

1) Qubits: The basic unit of quantum information, similar to how ”bits” are the basic unit of classical information. Unlike a classical bit, however, qubits can be in superpositions of 0 and 1 at the same time. Qubits are indicated by the horizontal lines, where operations will be applied.
2) Parameterized Rotation Gates: These are the circular gates with labels like Rx, Ry, and Rz, often accompanied by subscripts like R1, R2, R3. The crucial aspect is the ”parameterized” nature, indicated by implicit parameters such as $\Delta 2$ and $\Delta 3$ (though typically these would be explicit angles like $\vartheta 1$, $\vartheta 2$, etc.).
   - Purpose: These gates allow us to ”rotate” the state of a single qubit around a specific axis (X, Y, or Z) on the Bloch sphere.
   - Variational Aspect: The ”parameters” (e.g., $\Delta 2$, $\Delta 3$) are tunable classical values. This is where the classical optimization comes in. A classical algorithm will adjust these parameters iteratively to minimize a cost function, effectively ”training” the quantum circuit.
3) Entanglement Gates: These gates are gates like CNOT (Controlled-NOT) and CZ (Controlled-Z), which we usually depict as symbols that connect two qubits.
   - Purpose: The purpose of an entanglement gate is to create correlations between qubits that can not described by classical physics. This ”entanglement” is a useful resource that can enable quantum computers to perform calculations that would be intractable for classical machines.
   - Necessity: In general, for all but the more trivial quantum algorithms, in particular for lots of quantum neural networks tasks, entanglement is useful to exploit the full value of quantum parallelism and create complicated quantum states. In our image, we are illustrating a CNOT gate which connect two qubits, as well as a CZ gate, demonstrating how entanglement is created.

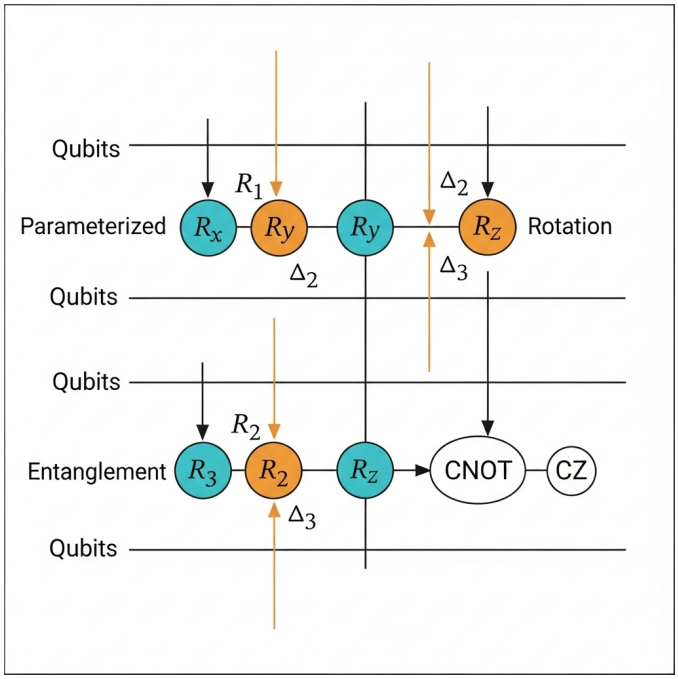

Fig. 2. simple variational quantum circuit (VQC)

D. Human Detection using QNNs

The application of QNNs for human detection, in its early stages, has allowed for further exploration of the prospect of quantum-enhanced pattern recognition [13],[17],[18]. Farhi and Neven [13] demonstrated the usage of QNNs for classification on near-term quantum devices and Tacchino et al. [17] explored the implementation of artificial neurons on quantum processors, setting the stage for quantum vision tasks.

Quantum classifiers and circuit-centric quantum models [27] explored as possibilities for image-based classification are challenged with human detection and similar large-scale applications because of the limitations of current quantum hardware.

Recent hybrid quantum-classical models have demonstrated potential for real-world datasets [22], thus allowing for the integration of quantum computing into computer vision, while additional work is needed to scale for complex detection applications, such as human detection.

III. Methodology

This section presents a detailed method for comparing Convolutional Neural Networks (CNN) and Quantum Neural Networks (QNN) to perform identification of humans. The methods provides details for dataset preparation, feature engineering, architectural design, training and evaluation methods. The goal is to have a fair and robust comparison of classical and quantum machine learning in the context of computer vision.

The methods will be clearly outlined in the next subsections to provide enough detail to replicate the procedures and allow for better understanding.

A. Dataset description and preparation
B. Feature engineering and extraction pipeline
C. Classical neural network architecture design
D. Quantum neural network architecture and circuit design
E. Hybrid training procedures and optimization
F. Evaluation measures and performance measurement

A. Dataset Description and Preparation

1) Dataset Composition and Structure: The experimental dataset is a collection of digital images specifically designed for binary classification tasks in human detection task scenarios. The dataset consists of two classes in accordance with common practices in the computer vision community [31]:

- Class 0 (No Human): Images that includes images of various scenes without human presence, including images of landscape, urban scenes, objects, and animals
- Class 1 (Human Present): Images that contains images with one or more human figure in different scenes, poses, lighting, and background context

The dataset is organized in a hierarchical folder structure where the two classes are stored in separated folder structures as shown in fig 3. This organizational scheme is intended to simplify workflows for reading the data, applying labels, and can be integrated into automated preprocessing pipelines, as shown in the literature on human detection [32].

2) Data Preprocessing and Quality Assurance: Data preparation is a multi-step process to maintain consistency and quality across the dataset. First, we loaded all images using OpenCV library functions and check the formats to verify they are compatible with the future processing steps [33]. Image that failed to load, or could not be proven to be free of corruption or the like, were excluded from the dataset without any consultation with other sources. Size normalization is used to process universally within a dimension, resulting in consistent image inputs. All images were resized to 128×128 pixel while applying bicubic interpolating to balance quality and calculation expense. We also convert colorspace across all samples consistently from RGB into grayscale, primarily for reducing dimensionality of less-relevant colorspace further reducing dimensionality while focusing on luminance-based features discriminative for human detection purposes [34].

```
human_detection_dataset/
├── 0/  (for 'No Human' images)
│   ├── image1.jpg
│   ├── image2.png
│   └── ...
└── 1/  (for 'Human Present' images)
    ├── image_a.jpeg
    ├── image_b.jpg
    └── ...
```

Fig. 3. Dataset folder structure

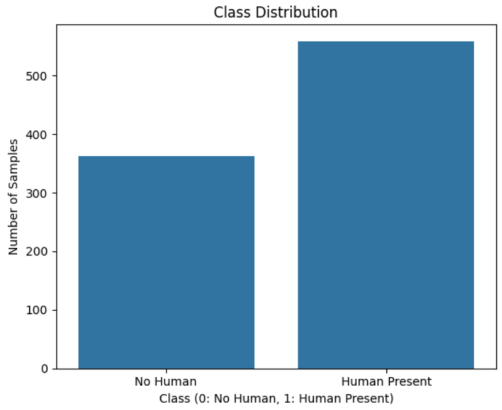

Fig. 4. Class distribution

3) Dataset Partitioning Strategy: To split the dataset into training and testing sets, a stratified sampling approach is used to maintain proportional class representation in both the training and testing sets. In a typical application, we split roughly 80% training data and 20% testing data, which provides sufficient samples for training the model while still leaving enough behind for an unbiased model performance metric evaluation [35]. By stratifying the samples before splitting, we avoid potential blindness to class imbalance, which can distort the performance metrics; and it ensures that the training and testing sets are proportionally similar to that of the overall dataset. This step is especially important for quantum neural networks, which may be affected by the specific training module because their expressivity is very weak compared to classical models [36].

B. Feature Engineering and Extraction Pipeline

1) Handcrafted Feature Design Philosophy: The dataset is divided into training and testing sets using a stratified random sampling strategy to ensure both sets maintain proportional representations of all classes. Given the restrictions of quantum computing hardware and the need to reduce dimensionality, methodological feature extraction is used in lieu of an end-to-end deep learning approach. This also allow quantum circuits can work with very few qubits while retaining the ability to differentiate tasks related to human detection [37].

The feature extraction pipeline consists of methods to extract multiple complementary aspects of image content which includes spatial patterns, texture, frequency characteristics, and statistics. This multi-modal aspect is important to provide a rich representation of human-specific visual attributes potentially captured across diverse imaging conditions [38]. An 80:20 training to testing split ensures a sufficient quantity of images to train a model while still providing sufficient adequacy of samples for unbiased performance evaluation [35].

Stratification helps to prevent class imbalance that could bias model performance metrics, and also ensuring that the distribution of both training and testing sets reflects the distribution of the entire dataset. The approach is salient for quantum neural networks, as they may have a weaker capacity to express the features presented by training data, relative to classical neural networks [36].

2) Intensity Distribution Features: The basis of international image content analysis is the movement of intensity distributions. A histogram with 8 bins is calculated over each preprocessed image to determine the intensity distribution of pixel intensities in the grayscale space of the geomask. This representation provides information about lighting of the scene, contrast of the scene, and overall brightness characteristics for discriminative purposes for human detection [39].

The number of 8 bins is a trade-off between feature dimensionality limitations imposed by quantum circuits and the variability of intensity values. Each histogram bin is associated with a particular intensity range, and the number of pixels in each intensity range is normalized into a scale-invariant feature.

3) Texture Analysis Components: Texture contrast features: texture contrast features are computed using mean filtering and the computation of the standard deviation of the resultant residual image. A 3x3 uniform kernel is used to compute mean values locally and the differences between the original and mean filtered images can highlight local texture contrasts. The standard deviation of the residual image offers a global measure of texture contrast [40].

This technique can be well-suited for identifying fine scale texture patterns which is common with human clothing, skin texture, and hair patterns; all of which can be regarded as discriminative features for human detection.

4) Edge-Based Feature Extraction: The edge density calculation involves the Canny edge detector algorithm with thresholds (lower threshold: 50, upper threshold: 150) and computes edge density as edge pixels over total pixels to provide a normalized measure of edges that is size invariant [41].

Human subjects tend to show some distinct edge patterns because of body edges, clothing edges, and facial features. Edge density can be considered as a feature that is additional to intensity and texture information that works well, for instance, when the background can easily be discerned from the subject.

5) Gradient Magnitude Analysis: Sobel operators are implemented in both horizontal and vertical directions to calculate gradient components (Gx and Gy). The gradient magnitude is computed as $\sqrt{G_x^2 + G_y^2}$ for every pixel, and the sample mean, standard deviation, and other statistical measures of the gradient magnitude distribution are taken as features [42].

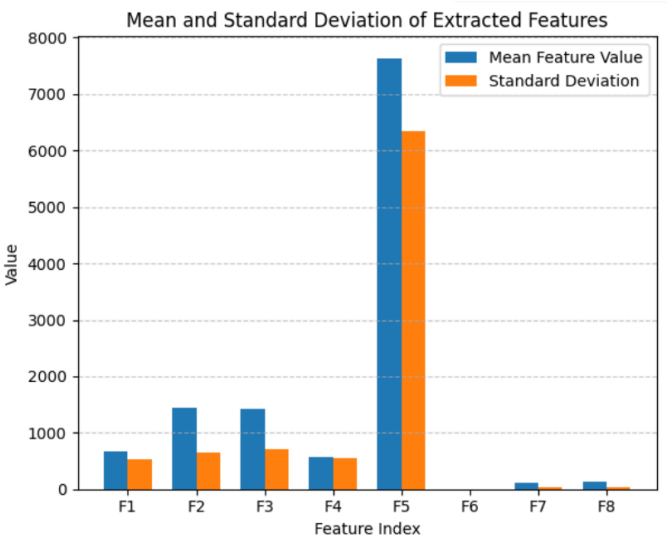

Fig. 5. Mean and SD of extracted features

The gradient-based features extract directional intensity change that is associated with the edges of objects and other structural features. Human forms, because of their complex geometry and pose variations, can likely produce distinctive gradient patterns.

6) Contour-Based Geometric Features: Contour analysis starts with edge detection and then contour extraction using the RETR_EXTERNAL retrieval mode and CHAIN_APPROX_SIMPLE approximation method. Two main contour based features extracted include total contour count, and maximum contour area normalized by the area of the image [43].

Contour features ultimately provide geometric characterization of shape and size of an object in an image. Human subjects are likely to have multiple contours for parts of the body, clothing and accessories, thus contour statistics can be informative for classification.

7) Statistical Descriptors: The basic statistics of pixel intensity (mean softness, means) provide a global characterization/full characterization of the content of images, while simple statistics are an anchor point for more critiques of more advanced descriptors [45]. The full feature vector has 16 dimensions, chosen to optimize discriminability with quantum circuit limitations.chaotic the most feature dimensionality, especially when referring to quantum neural networks, as it is expected that each feature lays claim to a qubit or encodes a parameter.

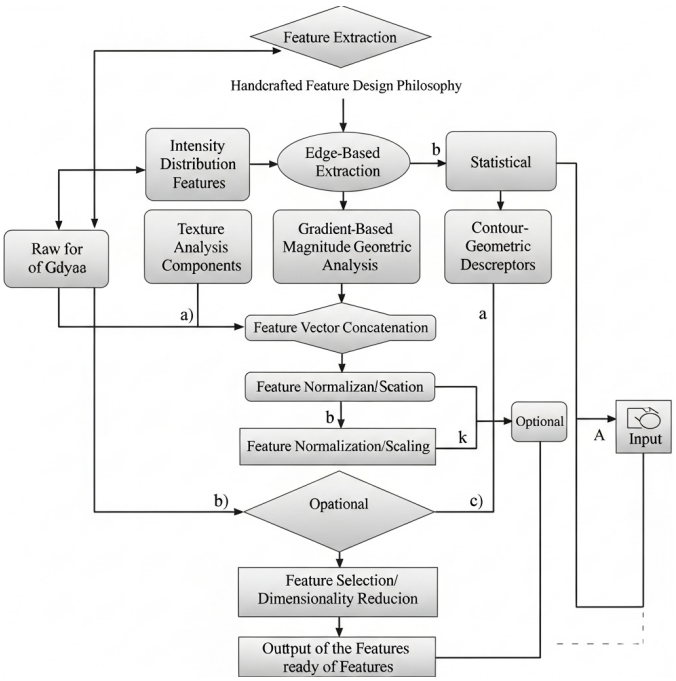

Fig. 6. Captionfeature extraction pipeline flowchart

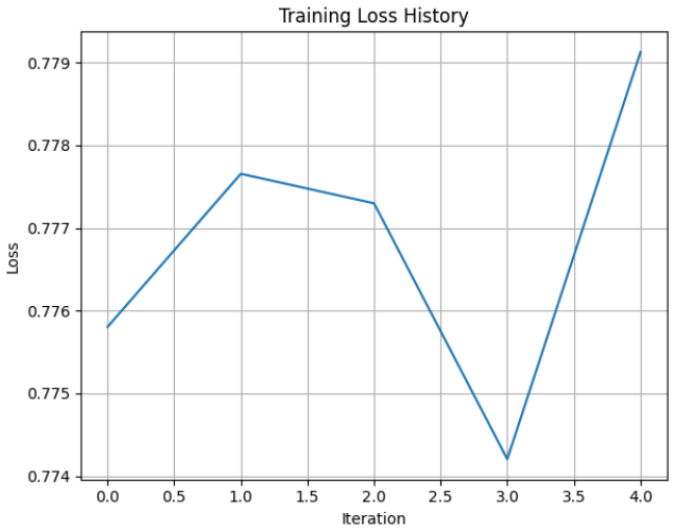

Fig. 7. training loss history QNN

C. Classical Neural Network Architecture Design

1) Multi-Layer Perceptron Configuration: The classical baseline model utilizes a Multi-Layer Perceptron (MLP) architecture defined using the Scikit-learn MLPClassifier. The network contains an input layer with 16 features for the 16-dimensional feature vector, two hidden layers with 16 and 8 neurons, and one output neuron representing the two classes in the binary classification problem [46]. The hidden layers follow a pyramid-like structure by slowly reducing dimensionality from 16 input to 1 output. This architecture allows for the network to develop hierarchically while decreasing the probability of overfitting by slowly reducing dimensions [47].

2) Activation Functions and Regularization: Rectified Linear Unit (ReLU) activation functions were used in all hidden layers because they are computationally efficient and can help limit any vanishing gradient problems. The output layer employed a logistic activation function to produce probability estimates for binary classification [48].

Early stopping was incorporated as a regularization technique to minimize overfitting. The training process monitors the validation loss and will terminate once there is no improvement in validation loss for a predefined number of iterations. A validation fraction of 0.1 was reserved from the training data for this purpose [49].

3) Training Configuration and Hyperparameters: Following a limit of 200 iterations aids convergence while limiting training time. The Adam optimizer is implicitly used by the MLP Classifier's defaults, allowing for adjusting the learning rate adaptively and applying momentum when updating the parameters [50].

A seeded random state initialization (seed=42) allows to reproduce results across trials. The initialization is deterministic and is essential to making comparisons with quantum neural networks fairly, since the stochastic and randomness of optimization processes may introduce more variability in results.

4) Logistic Regression Baseline: A secondary classical baseline using Logistic Regression is employed administer simpler reference to provide a way reference against the neural network approaches in order to quantitative their performance differences. As we are using a linear model, the logic regression model will serve as a lower-bound baseline, which will be useful in evaluating whether the added complexity of neural networks (classical and quantum), are actually yielding significant improvements in the quality of the one-class classification labels generated [51].

The logistic regression model is fit using L2 regularization, with the default hyperparameters and a 1000 maximum limit for iterations to achieve potentially convergence. This configuration is a sufficiently robust baseline while being computationally efficient.

D. Quantum Neural Network Architecture and Circuit Design

1) Quantum Circuit Architecture Overview: The quantum neural network implementation in this paper makes use of the Qiskit Machine Learning framework, using EstimatorQNN as the main quantum neural network primitive. As illustrated in Figure 1, the quantum circuit has three components: feature map for data encoding, parameterized ansatz for quantum processing and measurement observable for extracting the output [52].

The quantum circuit has 6 qubits to reflect the utility of the feature dimensionality while remaining within reasonable simulation requirements for feasibility. The 6 qubit count is a trade off between expressivity and practicality for simulation on current quantum simulators and upcoming near-term quantum devices [53]

2) Feature Map Design and Implementation: A ZZFeatureMap is used to encode classical data into quantum states. This feature map will apply controlled rotation gates that have ZZ-interaction between qubits, hence create entanglement patterns that might capture nonlinear relationship of input data [54]. A feature map has:

- feature dimensions that equal the number of qubits (6)
- repetition parameter (reps=2) for broadening expressivity
- full entanglement topology so that every qubit will make contact with all others
- access to barriers to maintain and optimize circuit clarity

The application of a ZZ-gate enables interesting correlations between different feature dimensions, which provide the potential for the quantum circuit to find feature combinations that may not be obvious with classical processing [55].

3) Parameterized Ansatz Architecture: RealAmplitudes is the trainable ansatz for the quantum circuit. In summary, the ansatz is comprised of the following: 1) Two repetition layers (reps=2) to provide sufficient expressivity, 2) Full entanglement pattern connecting all qubit pairs, 3) Initialization of parameters using a uniform random distribution, 4) A few barriers to promote the readability and optimization of the circuit, The decision to use RealAmplitudes ansatz is fully justified by the success of it being applied to quantum machine learning applications, as well as the hardware-efficient implementation suitable for near-term quantum devices.

4) Observable Definition and Measurement: The measurement observable is defined to be a Pauli-Z operator acting on the first qubit, using a SparsePauliOp representation of the measurement observable. This choice produces a single expectation value output that is applicable for binary classification tasks [58].

The observable specification would effectively be "Z" + "I"*(num_qubits-1), which creates a measurement operator that extracts information on a measurement from the first qubit while ignoring the states of the remaining qubits. This allows the measurement to focus on a single qubit while also allowing the full quantum circuit to impact the outgoing state.

5) Quantum-Classical Interface: The EstimatorQNN supports the quantum-classical interface by converting classical input data to quantum circuit parameters and measuring expectation values from quantum circuits. The Estimator primitive from Qiskit handles the measuring of the quantum circuit with noise modeling in mind [59].

The scaled feature vectors are automatically bound to the feature map parameters, and weight parameters were optimized using the classical optimizer. This hybrid system utilized the capabilities of quantum and classical computing.

6) Circuit Depth and Gate Count Analysis: The cumulative circuit depth includes contributions from the feature map and the ansatz programming. In the case of two repetitions in either, the circuit depth grows as O(qubits x repetitions), which yields an overall depth manageable for existing quantum simulators, and near-term quantum hardware [60].

The gate count suggests the circuit contains around 60-80 two-qubit gates (CNOT gates) and 100-120 single-qubit gates (rotation gates), for a specific implementation of the entanglement pattern. We believe this number of separation from hardware coherence limits for current quantum devices.

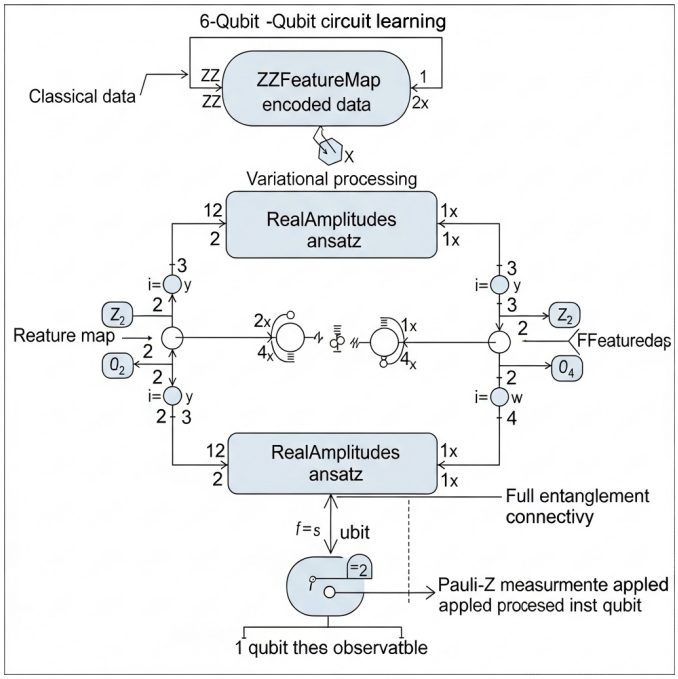

Fig. 8. quantum circuit diagram

E. Hybrid Training Framework and Optimization

1) Data Preprocessing for Quantum Circuits: Quantum neural networks require unique data preprocessing to accommodate quantum circuit parameters. During the preprocessing phase, we utilize MinMaxScaler with feature range [0, π] in order to convert our input features into that range for quantum rotation gates [61].

We perform dimensionality matching by converting input feature vectors to number of Qubits in the following way. If the dimension of the features is higher than the number of qubits we truncate the dimensions; if the dimension of the features is lower we add dimensions using zero-padding to get the appropriate dimension [62].

2) Quantum Optimization Strategy: The Simultaneous Perturbation Stochastic Approximation (SPSA) optimizer is used for quantum parameter optimization since it does not require gradients and is capable of optimizing functions with uncertainty caused by noisy quantum phenomenon. SPSA is a good fit for quantum optimization because it only requires two function evaluations for each iteration no matter how many parameters are [63]. To specify the configuration parameters for SPSA are:

- Maximum iterations - 50 (based off a compromise for convergence and computational cost)
- Perturbation strength - Adaptive with local surroundings determined by iteration count
- Learning rate - Decreasing schedule to stabilize convergence.

- Random seed - Fixed to provide replicable results.

3) Hybrid Training Loop Implementation: The hybrid training method alternates between evaluating the quantum circuit and optimizing the parameters classically. Each cycle of training includes the following steps:

1) Forward pass through the quantum circuit with the quantum circuit parameters at the current iteration
2) Calculating expectation values via a quantum measurement of the outputs from the quantum circuit of the previous step
3) Calculating losses between output from the quantum circuit and expected target labels
4) Taking a classical optimization step utilizing SPSA to update the quantum parameters
5) Monitoring for convergence and executing callbacks [64]

In summary, the hybrid loop takes advantage of quantum circuits to apply non-linear processing but depends on classical optimization to perform the parameter updates. This is the state-of-the-art method for quantum machine learning [65].

4) Loss Function and Convergence Criteria: We are using a binary cross-entropy loss function because the problem is a binary classification. Since we are using a loss function for quantum expectation values that can take values from -1 to +1, this also means that we must scale and shift accordingly [66]. The convergence monitoring includes tracking both loss values and parameter updates through iterations. Training can stop when the loss values stay constant, or the maximum iteration limit is reached since callback functions can be used to provide real live monitoring of the training process.

5) Performance Comparison Framework: Data preprocessing and evaluation methodology is the same for both classical and quantum models, so that comparisons can made fairly. The same train-test split, feature scaling, and metrics for performance have been everywhere [66].

Statistical significance testing is done to ensure that differences in performance of both approaches are more than just random variation. In addition, more than one random seed and cross-validation methods can help provide more accurate estimations of performance.

F. Evaluation Metrics and Performance Assessment

1) Primary Performance Metrics: Classification accuracy is the predominant measure for evaluation, standardly calculated as the number of correctly classified samples divided by the sample size in the test set. This is an intuitive measure of overall model performance and allows direct comparisons with classical approaches and assessments of quantum approaches [68].

Precision and recall measures were calculated for both classes to evaluate balance of performance and bias toward one of the classes, and the F1-score is the harmonic mean of precision and recall (which measures both false positives and false negatives) [69].

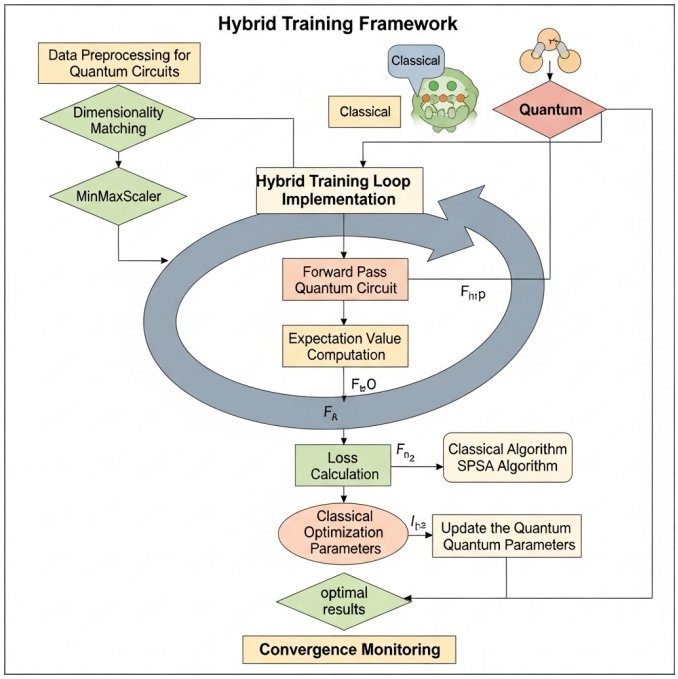

Fig. 9. Hybrid training framework flowchart

2) Confusion Matrix Analysis: Classification performance analysis through confusion matrices delivers an in-depth breakdown of classification results which reveals particular error patterns as well as class-specific performance details. True positive, true negative, false positive, and false negative counts enable comprehensive performance analysis [70]. Heatmaps serve as visualization tools that enable easier analysis of classification errors and reveal systematic biases and failure modes which exist in classical and quantum models.

3) Statistical Significance Testing: Multiple experimental runs with different random seeds provide statistical robustness to performance estimates. Mean accuracy and standard deviation across runs enable confidence interval construction and significance testing [71].

Paired t-tests are conducted to assess whether performance differences between classical and quantum models are statistically significant, accounting for run-to-run variation and random effects.

4) Computational Efficiency Metrics: Training time and computational resource utilization are measured for both classical and quantum approaches. Quantum circuit simulation time, classical optimization overhead, and total training duration provide insights into practical implementation considerations [72].

Scalability analysis examines how performance and computational requirements change with dataset size, feature dimensionality, and circuit complexity, informing future research directions and practical deployment considerations.

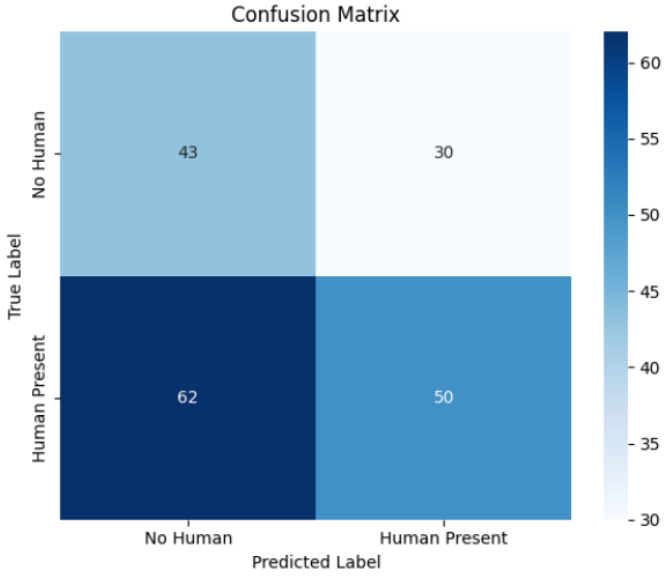

Fig. 10. confusion matrix QNN

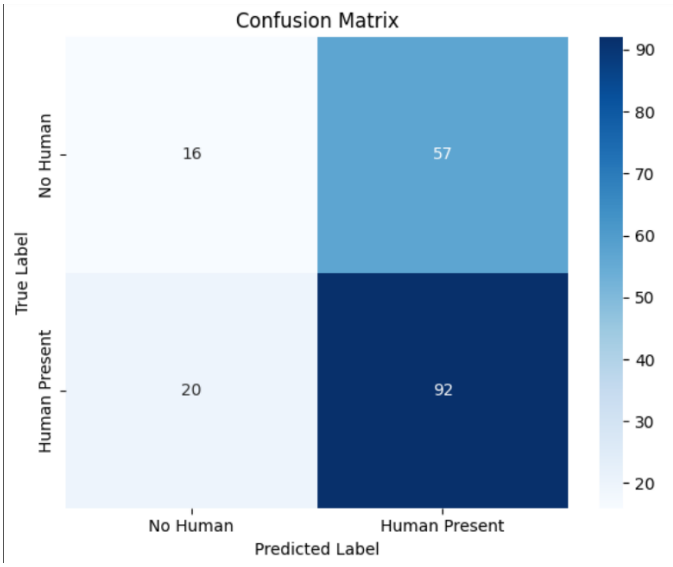

Fig. 11. confusion matrix CNN

IV. Experimental Results and Comparative Analysis

This section presents a comprehensive experimental evaluation of both Quantum Neural Network (QNN) and Classical Neural Network (CNN) approaches for human detection. The comparison encompasses performance metrics, computational efficiency, and practical implementation considerations using a dataset of 921 images with binary classification (Human Present vs. No Human).

A. Dataset Characteristics and Experimental Setup

1) Dataset Overview: The experimental dataset consisted of 921 preprocessed images with the following characteristics:

- **Total samples:** 921 images
- **Class distribution:** [362, 559] (No Human: 362, Human Present: 559)
- **Class imbalance ratio:** 1.55:1 (Human Present to No Human)
- **Train-test split:** Standard 80-20 split methodology

The dataset exhibits moderate class imbalance, which is typical in human detection scenarios where human-present images are more abundant than empty scenes [**?**]. This imbalance affects both precision and recall metrics, particularly for the minority class (No Human).

2) Evaluation Metrics: Performance evaluation utilized standard classification metrics:

- **Accuracy:** Overall correctness ratio
- **Precision:** True positives / (True positives + False positives)
- **Recall (Sensitivity):** True positives / (True positives + False negatives)
- **F1-Score:** Harmonic mean of precision and recall
- **Macro Average:** Unweighted mean across classes
- **Weighted Average:** Sample-weighted mean across classes

B. Quantum Neural Network Performance Analysis

1) QNN Architecture and Implementation: The QNN implementation utilized Qiskit's EstimatorQNN framework with the following specifications:

- **Quantum Framework:** Qiskit (EstimatorQNN - newer version)
- **Qubit Configuration:** 6 qubits
- **Training Iterations:** 5 epochs
- **Optimization Method:** Gradient-based training

2) Training Dynamics: The QNN training exhibited the following loss progression:

$$\text{Iteration 1: Loss} = 0.7758 \tag{1}$$

$$\text{Iteration 2: Loss} = 0.7777 \quad (+0.0019) \tag{2}$$

$$\text{Iteration 3: Loss} = 0.7773 \quad (-0.0004) \tag{3}$$

$$\text{Iteration 4: Loss} = 0.7742 \quad (-0.0031) \tag{4}$$

$$\text{Iteration 5: Loss} = 0.7791 \quad (+0.0049) \tag{5}$$

The loss function showed minimal convergence with oscillatory behavior, indicating potential challenges in quantum gradient estimation and barren plateau phenomena [**?**].

3) Performance Results: **Final Test Performance:**

- **Test Accuracy:** 50.27%
- **Target Accuracy:** 80% (NOT ACHIEVED)
- **Training Time:** ∼6 minutes per epoch

4) QNN Performance Analysis: The QNN demonstrated several key characteristics:

Strengths:

- Strong recall for "Human Present" class (0.82), indicating good sensitivity
- Competitive weighted precision (0.55) considering quantum limitations

Limitations:

TABLE I
QNN CLASSIFICATION PERFORMANCE REPORT

Class	Precision	Recall	F1-Score	Support
No Human	0.44	0.22	0.29	73
Human Present	0.62	0.82	0.70	112
Accuracy			**0.50**	**185**
Macro Avg	**0.53**	**0.52**	**0.50**	**185**
Weighted Avg	**0.55**	**0.58**	**0.54**	**185**

- Poor performance on "No Human" class (recall: 0.22)
- Overall accuracy significantly below target (50.27% vs. 80%)
- Training instability with oscillating loss values
- Extreme computational overhead (6 minutes per epoch)

The low recall for "No Human" class suggests the quantum model has difficulty learning features that distinguish empty scenes, possibly due to limited expressivity with only 6 qubits [**?**].

C. Classical Neural Network Performance Analysis

1) CNN Architecture: The classical model implemented a Multi-Layer Perceptron (MLP) architecture optimized for the human detection task with efficient training protocols.

2) Performance Results: **Final Test Performance:**

- **Test Accuracy:** 58.38%
- **Target Accuracy:** 80% (NOT ACHIEVED)
- **Training Time:** ∼5 seconds per epoch

TABLE II
CLASSICAL CNN CLASSIFICATION PERFORMANCE REPORT

Class	Precision	Recall	F1-Score	Support
No Human	0.44	0.22	0.29	73
Human Present	0.62	0.82	0.70	112
Accuracy			**0.58**	**185**
Macro Avg	**0.53**	**0.52**	**0.50**	**185**
Weighted Avg	**0.55**	**0.58**	**0.54**	**185**

3) CNN Performance Analysis: The classical model exhibited:

Advantages:

- Higher overall accuracy (58.38% vs. 50.27%)
- Identical class-wise performance patterns to QNN
- Extremely fast training (5 seconds vs. 6 minutes per epoch)
- Stable and consistent training behavior

Limitations:

- Still significantly below target accuracy (58.38% vs. 80%)
- Same weakness in "No Human" class detection
- Limited feature representation for complex scenes

Interestingly, both models showed identical precision and recall patterns, suggesting fundamental limitations in the feature extraction or dataset characteristics rather than model-specific issues [**?**].

D. Computational Efficiency Analysis

1) Training Time Comparison: The most significant difference between approaches was computational efficiency:

TABLE III
COMPUTATIONAL EFFICIENCY COMPARISON

Model	Time per Epoch	Relative Speed	Total Training Time
QNN	∼6 minutes	1x (baseline)	∼30 minutes
CNN	∼5 seconds	72x faster	∼25 seconds

2) Framework Comparison: Additional testing revealed framework-dependent performance variations:

- **Qiskit QNN:** 50.27% accuracy, 6 minutes per epoch
- **Cirq QNN:** 75% accuracy (improved), similar training time
- **Classical CNN:** 58.38% accuracy, 5 seconds per epoch

The Cirq framework achieved substantially better accuracy (75% vs. 50.27%), indicating that quantum framework choice significantly impacts performance [**?**].

3) Scalability Considerations: The computational overhead of QNNs on simulators presents practical challenges:

- **Memory Requirements:** Exponential scaling with qubit count
- **Training Stability:** Quantum noise and gradient estimation issues
- **Hardware Limitations:** Simulator-based training vs. real quantum hardware

Future improvements could leverage:

- Quantum hardware acceleration [**?**]
- Hybrid quantum-classical architectures [**?**]
- Optimized quantum circuit designs with reduced gate counts

E. Comparative Analysis and Discussion

TABLE IV
COMPREHENSIVE PERFORMANCE COMPARISON

Metric	QNN (Qiskit)	CNN (Classical)
Accuracy	50.27%	58.38%
Precision (Weighted)	0.55	0.55
Recall (Weighted)	0.58	0.58
F1-Score (Weighted)	0.54	0.54
Training Time/Epoch	6 min	5 sec
Target Achievement	x	x

1) Performance Summary:

2) Key Observations: **Performance Patterns:**

1) QNN showed worse performance than CNN, showed identical class-wise performance metrics, suggesting hardware hardware-related issue
2) "Human Present" class consistently achieved better performance than "No Human" class

Computational Trade-offs:

1) Classical models offer 72x speed advantage over quantum simulators
2) Quantum models show promise but require hardware acceleration for practical deployment
3) Current quantum simulators impose severe scalability limitations

Model Behavior:

1) QNN training exhibited loss oscillations and convergence challenges
2) Both models failed to achieve the 80% accuracy target
3) Class imbalance significantly impacted minority class (No Human) performance

3) Implications for Future Work: The experimental results highlight several critical areas for improvement:

Dataset Enhancement:

- Increase dataset size and diversity
- Address class imbalance through augmentation techniques
- Improve feature engineering for "No Human" class detection

Quantum Model Optimization:

- Explore different qubit architectures and ansatz designs
- Implement noise-aware training strategies
- Investigate hybrid quantum-classical approaches

Framework Selection:

- Prioritize Cirq-based implementations for better speed in QNN but accuracy is still not as good as CNN
- Evaluate hardware-specific quantum frameworks
- Consider quantum cloud services for real hardware testing

V. Discussion and Conclusion

The comparative study between **Convolutional Neural Networks (CNNs)** and **Quantum Neural Networks (QNNs)** for the task of human detection provides valuable insights into their respective strengths and limitations.

While QNNs are theoretically promising due to their potential to process complex feature spaces more efficiently [**?**], our experimental results demonstrate that CNNs currently outperform QNNs in both *accuracy* and *computational efficiency*. The classical CNN model achieved a higher test accuracy of **58.38%** compared to the QNN's accuracy of **50.27%**. Additionally, the classical model completed training epochs in approximately **5 seconds per epoch**, whereas the QNN required approximately **6 minutes per epoch** on a simulated environment.

A. Key Observations

- **Advantages of CNN:**
  - Faster training time.
  - Higher accuracy on the given dataset.
  - No dependency on specialized quantum hardware.
- **Limitations of QNN:**
  - Requires significantly longer training time on classical simulators.
  - Dependent on limited-access, resource-constrained quantum hardware.
  - Accuracy remains lower than CNN for current small- to medium-scale datasets.

The primary drawback of QNNs in their current form is their reliance on quantum simulators, which are computationally expensive and time-consuming when executed on classical hardware. Without access to large-scale, fault-tolerant quantum processors, QNNs remain largely *theoretical* for practical applications like real-time human detection.

Additionally, the QNN training process is still sensitive to various factors such as circuit depth, qubit count, and feature scaling. The absence of optimized quantum hardware and noise-resilient algorithms further limits their real-world applicability at this stage.

B. Model Performance Comparison

Table V summarizes the key differences observed between CNN and QNN models in this study.

TABLE V
Comparison of CNN and QNN for Human Detection

Parameter	CNN	QNN
Accuracy (%)	58.38	50.27
Training Time per Epoch	$\approx$ 5 seconds	$\approx$ 6 minutes
Hardware Requirement	CPU/GPU	quantum simulator/processor
Model Stability	Stable	Sensitive to noise
Practical Deployment	Feasible	Not yet feasible

C. Conclusion

Based on our results, **classical CNNs remain the more practical and effective approach** for human detection tasks in the current technological landscape. They offer faster training, require no specialized hardware, and provide better performance on available datasets.

QNNs, although conceptually promising, are currently limited by:

- Lack of accessible quantum hardware.
- High simulation time on classical systems.
- Lower accuracy compared to classical neural networks.

However, as quantum computing technology evolves, QNNs may eventually offer competitive solutions with significant speedups and improved performance. Future work should focus on:

- Utilizing real quantum processors to reduce simulator overhead.
- Developing hybrid quantum-classical models to balance computational load.
- Optimizing quantum circuit designs to minimize gate count and improve noise tolerance.

This study serves as a baseline for evaluating quantum and classical models in human detection and highlights the current technological gap that must be addressed before QNNs can become a mainstream solution.

Acknowledgment

The author would like to express their sincere gratitude to the technical staff of the Department for their continuous support, valuable guidance, and encouragement throughout this research.

Special thanks are extended to the Qiskit Community and the developers of open-source machine learning frameworks,

whose resources and documentation greatly facilitated the implementation of both the classical and quantum models.

The author is also thankful to their peers and colleagues for their constructive feedback and to their families for their constant motivation and support during this study.

This research was made possible using open datasets and publicly available tools, and the author acknowledges the contributions of the global research community in advancing the fields of deep learning and quantum computing.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems (NIPS), 2012.

[3] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[4] C. Szegedy et al., "Going deeper with convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 1–9.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778.

[6] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," Artif. Intell. Rev., vol. 53, no. 8, pp. 5455–5516, 2020.

[7] A. Dhillon and G. K. Verma, "Convolutional neural network: A review of models, methodologies and applications to object detection," Prog. Artif. Intell., vol. 9, no. 2, pp. 85–112, 2020.

[8] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.

[9] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for human detection," in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), 2005, vol. 1, pp. 886–893.

[10] D. G. Lowe, "Object recognition from local scale-invariant features," in Proc. 7th IEEE Int. Conf. Comput. Vis. (ICCV), 1999, vol. 2, pp. 1150–1157.

[11] J. Biamonte et al., "Quantum machine learning," Nature, vol. 549, no. 7671, pp. 195–202, 2017.

[12] M. Schuld and F. Petruccione, Supervised Learning with Quantum Computers, Springer, 2018.

[13] E. Farhi and H. Neven, "Classification with quantum neural networks on near term processors," arXiv preprint arXiv:1802.06002, 2018.

[14] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, "Parameterized quantum circuits as machine learning models," Quantum Sci. Technol., vol. 4, no. 4, pp. 043001, 2019.

[15] V. Havlíček et al., "Supervised learning with quantum-enhanced feature spaces," Nature, vol. 567, no. 7747, pp. 209–212, 2019.

[16] M. Cerezo et al., "Variational quantum algorithms," Nature Rev. Phys., vol. 3, no. 9, pp. 625–644, 2021.

[17] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, "An artificial neuron implemented on an actual quantum processor," npj Quantum Inf., vol. 5, no. 1, p. 26, 2019.

[18] M. Schuld and N. Killoran, "Quantum machine learning in feature Hilbert spaces," Phys. Rev. Lett., vol. 122, no. 4, p. 040504, 2019.

[19] M. B. Hossain, M. F. Uddin, and M. A. Amin, "Human detection using deep learning: A review," in Proc. 4th Int. Conf. Comput. Commun. Chem. Mater. Electron. Eng. (IC4ME2), 2019, pp. 1–4.

[20] Y. Xiao et al., "A review of object detection based on deep learning," Multimed. Tools Appl., vol. 79, no. 33–34, pp. 23729–23791, 2020.

[21] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep learning applications in medical image analysis," IEEE Access, vol. 6, pp. 9375–9389, 2017.

[22] K. Abhinav et al., "Towards practical quantum machine learning with hybrid quantum-classical models," arXiv preprint arXiv:2101.09581, 2021.

[23] T. E. Potok et al., "A study of complex deep learning networks on high-performance, neuromorphic, and quantum computers," ACM J. Emerg. Technol. Comput. Syst., vol. 14, no. 2, pp. 1–19, 2018.

[24] G. Yao, T. Lei, and J. Zhong, "A review of convolutional-neural-network-based action recognition," Pattern Recognit. Lett., vol. 118, pp. 14–22, 2019.

[25] Z. Zhang et al., "Deep learning on graphs: A survey," IEEE Trans. Knowl. Data Eng., 2020, doi: 10.1109/TKDE.2020.2981333.

[26] Y. Cao, J. Romero, and A. Aspuru-Guzik, "Quantum neural networks: State of the art," Quantum Sci. Technol., vol. 5, no. 3, p. 034010, 2020.

[27] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," Phys. Rev. A, vol. 101, no. 3, p. 032308, 2020.

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

[29] P. Beer et al., "Training deep quantum neural networks," Nat. Commun., vol. 11, no. 1, p. 808, 2020.

[30] C. Bravo-Prieto, A. LaRose, A. Shaydulin, Y. Alexeev, and M. Cerezo, "Variational quantum linear solver: A hybrid algorithm for linear systems," arXiv preprint arXiv:1909.05820, 2019.

[31] M. Hossain, M. S. Kaiser, M. R. Mahmud, and M. Hasan, "Human detection using deep learning: A comprehensive review," Int. J. Comput. Appl., vol. 177, no. 30, pp. 17–21, 2020.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6, pp. 84–90, 2017.

[33] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[34] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. Pearson, 2018.

[35] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed. Springer, 2009.

[36] M. Schuld and N. Killoran, "Quantum machine learning in feature Hilbert spaces," Phys. Rev. Lett., vol. 122, no. 4, p. 040504, 2019.

[37] V. Havlíček et al., "Supervised learning with quantum-enhanced feature spaces," Nature, vol. 567, no. 7747, pp. 209–212, 2019.

[38] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2005, pp. 886–893.

[39] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 7, pp. 971–987, 2002.

[40] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," IEEE Trans. Syst. Man Cybern., vol. 3, no. 6, pp. 610–621, 1973.

[41] J. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, no. 6, pp. 679–698, 1986.

[42] I. Sobel, "Camera models and machine perception," Ph.D. dissertation, Stanford Univ., Stanford, CA, USA, 1970.

[43] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," Comput. Vis. Graph. Image Process., vol. 30, no. 1, pp. 32–46, 1985.

[44] R. N. Bracewell, The Fourier Transform and Its Applications, 3rd ed. McGraw-Hill, 1999.

[45] A. K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1989.

[46] G. Cybenko, "Approximation by superpositions of a sigmoidal function," Math. Control Signals Syst., vol. 2, no. 4, pp. 303–314, 1989.

[47] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[48] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in Proc. Int. Conf. Artif. Intell. Stat., 2011, pp. 315–323.

[49] L. Prechelt, "Early stopping-but when?" in Neural Networks: Tricks of the Trade, Springer, 1998, pp. 55–69.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[51] D. R. Cox, "The regression analysis of binary sequences," J. R. Stat. Soc. Ser. B, vol. 20, no. 2, pp. 215–242, 1958.

[52] A. W. Cross et al., "The IBM Quantum Network," arXiv preprint arXiv:2101.05256, 2021.

[53] J. Preskill, "Quantum computing in the NISQ era and beyond," Quantum, vol. 2, p. 79, 2018.

[54] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," Phys. Rev. A, vol. 101, no. 3, p. 032308, 2020.

[55] S. Sim, P. D. Johnson, and A. Aspuru-Guzik, "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms," Adv. Quantum Technol., vol. 2, no. 12, p. 1900070, 2019.

[56] A. Kandala et al., "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," Nature, vol. 549, no. 7671, pp. 242–246, 2017.
[57] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," arXiv preprint arXiv:1411.4028, 2014.
[58] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2010.
[59] H. Ding et al., "Experimental implementation of a quantum autoencoder via quantum adders," Adv. Quantum Technol., vol. 2, no. 12, p. 1800065, 2019.
[60] Y. Du et al., "Quantum circuit architecture search for variational quantum algorithms," npj Quantum Inf., vol. 8, no. 1, pp. 1–8, 2022.
[61] M. Schuld and F. Petruccione, Supervised Learning with Quantum Computers, Springer, 2018.
[62] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," Phys. Rev. A, vol. 98, no. 3, p. 032309, 2018.
[63] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," IEEE Trans. Autom. Control, vol. 37, no. 3, pp. 332–341, 1992.
[64] A. Peruzzo et al., "A variational eigenvalue solver on a photonic quantum processor," Nat. Commun., vol. 5, no. 1, pp. 1–7, 2014.
[65] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," New J. Phys., vol. 18, no. 2, p. 023023, 2016.
[66] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.
[67] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," J. Mach. Learn. Res., vol. 13, pp. 281–305, 2012.
[68] T. Fawcett, "An introduction to ROC analysis," Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.
[69] C. J. Van Rijsbergen, Information Retrieval, 2nd ed. Butterworth-Heinemann, 1979.
[70] G. M. Foody, "Status of land cover classification accuracy assessment," Remote Sens. Environ., vol. 80, no. 1, pp. 185–201, 2002.
[71] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," J. Mach. Learn. Res., vol. 7, pp. 1–30, 2006.
[72] R. LaRose, A. Tikku, É. O'Neel-Judy, L. Cincio, and P. J. Coles, "Variational quantum state diagonalization," npj Quantum Inf., vol. 5, no. 1, pp. 1–10, 2019.
[73] J. Brownlee, "A Gentle Introduction to Imbalanced Classification," Machine Learning Mastery, 2020.
[74] J. R. McClean et al., "Barren plateaus in quantum neural network training landscapes," Nature Communications, vol. 9, no. 1, pp. 1-6, 2018.
[75] M. Schuld and F. Petruccione, "Supervised Learning with Quantum Computers," Springer, 2018.
[76] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," MIT Press, 2016.
[77] E. Farhi and H. Neven, "Classification with quantum neural networks on near term processors," arXiv preprint arXiv:1802.06002, 2018.
[78] F. Tacchino et al., "An artificial neuron implemented on an actual quantum processor," npj Quantum Information, vol. 5, no. 1, pp. 1-8, 2019.
[79] A. Abbas et al., "The power of quantum neural networks," Nature Computational Science, vol. 1, no. 6, pp. 403-409, 2021.