Adversarially Robust Quantum-Classical Neural Network for Malware Detection

Abstract

The increasing sophistication of malware poses significant challenges to modern machine learning based detection systems. This has motivated growing interest in quantum and hybrid quantum-classical learning models as alternatives to improve representational efficiency and robustness. However, high sensitivity to quantum noise and limited scalability arises question on the deployment of Quantum Machine Learning (QML) models on Noisy Intermediate-Scale Quantum (NISQ) devices.

To address these challenges, we propose QSLP[1], a unified hybrid quantum–classical architecture for image-based malware detection that integrates robustness at both latent-space and pixel-space levels. QSLP uniquely integrates a novel defense mechanism, Quantum Noise Injection Class Center Perturbation (QNI-CCP), to add sensitivity-weighted perturbations into the latent space prior to the quantum circuit execution. Our proposed architecture consists of five stages: (i) classical feature extractor, (ii) quantum-informed perturbation injection, (iii) quantum neural processing, (iv) centroid-based regularization for stable feature learning and (v) adversarial training. Through QSLP, we investigate quantum advantage, latent-space robustness, and adversarial resilience under structured evaluation. Our simulated experiments across datasets show that integrating QNI-CCP, a part of QSLP, into classical neural network (CNN) and quantum neural network (QNN) pipelines improves both generalization performance and adversarial robustness at the latent space level. Complementing this latent-space defense, adversarial training in QSLP further strengthens the model against pixel-level perturbations. Additionally, we visualized feature-space representations and evaluated quantum circuit design choices, highlighting trade-offs between expressivity, robustness, and computational overhead within the QSLP framework. We demonstrate that carefully designed feature-space perturbations, combined with a lightweight quantum processing layer and adversarially guided robustness regularization, can enhance the robustness of hybrid malware detection systems and offer a promising direction for security applications in the NISQ era.

1 Introduction

With the growing adoption of digital technologies across various domains, cyber threats such as ransomware and malware are also evolving rapidly in complexity and frequency [30]. For instance, recent data reveals that the U.S. healthcare sector faced unprecedented challenges, with over 198 million patients impacted by ransomware [24]. Malware is equally concerning due to its serious consequences in sectors like finance, government, energy, defense, *etc.* A recent study shows that within a single year, more than two-hundred thousand never-before-seen malware variants, averaging over 600 new threats daily, have emerged, which suggests that threat actors are leveraging AI tools to rapidly create new variants that easily bypass static signature-based detection [24].

[1] **Q**uantum model with **S**ecure **L**atent-space and **P**ixel space

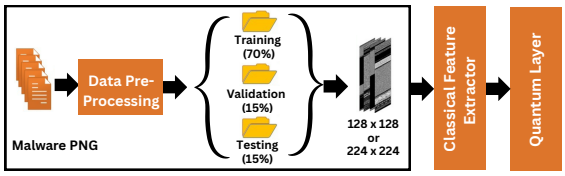


Figure 1: High level Architecture of QSLP - a Hybrid Quantum-Classical Neural Network.

As attackers increasingly employ intelligent, stealthy and dynamic malware variants, traditional defense mechanisms often fail to detect such threats. Studies have also shown that AI has empowered threat actors through advanced malware generation and obfuscation techniques [7]. Furthermore, the need for effective detection extends to real-world applications like Internet of Things (IoT) devices, where limited resources hinder the deployment of resource-hungry deep learning models.

To address these challenges, researchers have leveraged machine learning (ML) and deep learning (DL) techniques to identify hidden malicious patterns that otherwise could evade rule-based detection [27]. One of the most prominent methods used in malware detection is to convert malware binaries into grayscale images, allowing Convolutional Neural Networks (CNNs) to classify based on visual texture rather than signatures [16]. Despite this progress, traditional learning-based detection systems face major limitations. First, they lack generalization; classical models like standard CNNs trained on these grayscale representations may learn surface-level patterns unique to specific samples, rather than the underlying malicious logic. Also, malware authors increasingly use polymorphic and metamorphic techniques that alter the binary structure [8]. This limits the ability of generalization in classical models to such subtle and distributed changes, especially in the presence of noise or novel attack vectors. Models trained only on surface-level data can develop brittle decision boundaries, leading to poor generalization and adversarial vulnerability. Second, deep learning models such as CNNs are highly vulnerable to adversarial perturbations — slight modifications to input that can alter the model's output and compromise system reliability [28].

Consequently, Quantum Machine Learning (QML) has emerged as a transformative pathway. QML offers novel processing capabilities required to efficiently analyze the complex correlations within large datasets like network traffic data, potentially delivering the robust cybersecurity solutions needed for future defense [13]. Moreover, QML introduces unique properties such as superposition and entanglement to represent and process information in high-dimensional Hilbert spaces. These properties have been shown to enhance the robustness of decision boundaries [12]. Nevertheless, current QML models implemented on noisy intermediate-scale quantum (NISQ) devices tend to exhibit high sensitivity to quantum noise and have limited scalability [18]. This underscores the need

for a hybrid quantum-classical architecture capable of accurately detecting malware from transformed visual representations, while maintaining resilience to adversarial attacks and robustness against the quantum noise inherent in NISQ devices. Motivated by these challenges, in this work, we seek to systematically investigate the robustness and representational benefits of hybrid quantum–classical models through the following research questions (RQ):

RQ1: Do hybrid quantum–classical models demonstrate a measurable quantum-advantage over fully classical models in terms of performance, robustness, or computational efficiency?
RQ2: Is robustness in the latent representation space sufficient to ensure comprehensive adversarial robustness in hybrid quantum machine learning (QML) models?
RQ3: To what extent does adversarial training improve robustness and generalization in hybrid quantum–classical models?

To address these questions, we conduct a comprehensive set of experiments and propose a robust hybrid Quantum-Classical architecture, shown in Figure 1, that combines advantages of both classical and quantum models. Our contributions include:

- QSLP quantum-classical Architecture: We designed a hybrid quantum–classical model comprising a classical convolutional feature extractor and a quantum neural network, augmented with robust feature-space and pixel-space adversarial mechanisms, to process malware images while maintaining resilience against adversarial attacks (§3).
- QNI-CCP Defence Mechanism: We introduce QNI-CCP, which injects class-aware perturbations into the latent space. This mechanism operates after the classical encoder and before the quantum circuit (§3).
- Benchmarking against adversarial attacks: We evaluated all models trained at different stages against adversarial attacks to evaluate each method's effectiveness. To this end, we employ two distinct threat models: i) the Fast Gradient Sign Method (FGSM), a single-step attack, and ii) the Projected Gradient Descent (PGD) attack, an iterative method (§4).
- Quantum trade-off analysis: We study the computational overheads incurred due to quantum processing and their associated performance gains (§5).

2 Related Work

Evolution of Malware Detection: Traditional signature-based malware detection methods generally fail to cope with rapidly evolving cybersecurity threats. To address this, researchers turned to machine learning, specifically exploring image-based malware classification, where malware binaries are converted into grayscale images. Huang et al. [17] demonstrated how converting malware binaries into grayscale images allows CNNs to automatically learn discriminative patterns without the need for complex reverse engineering. Building on this, Zhong et al. [29] introduced VisMal, which applied Contrast Limited Adaptive Histogram Equalization (CLAHE) on malware images and used CNNs, achieving 96% accuracy with 4 ms per sample, showing how effective and scalable image-based methods can be. Shahnawaz et al. [22] took a dynamic approach by converting runtime Application Programming Interface (API) call arguments into enhanced grayscale images and applied CNNs, resulting in impressive classification results with over 98% accuracy across different malware families. Researchers have also applied similar image-based techniques for fragment classification [5], these approaches reveal the broader potential of image-based QML pipelines in real-world applications, including digital forensics and low-resource IoT.

However, classical image-based architectures introduced significant challenges regarding generalization. Gibert et al. [8] highlighted that while deep learning models avoid manual feature engineering, they often overfit by learning surface-level visual artifacts given by specific packers or compilers. Consequently, these models frequently fail to detect malware that employs polymorphic or metamorphic techniques, where the underlying malicious logic remains the same but the binary's visual texture is drastically altered.
Quantum Machine Learning in Cybersecurity On the other hand, QML is emerging as a field that combines quantum computing properties such as superposition and entanglement to solve problems better than classical algorithms. In general, QML has shown strong potential in cybersecurity. Shara [23] highlighted the transformative potential of quantum computing, mapping three QML paradigms: quantum-native, hybrid, and quantum-inspired across applications like anomaly detection and malware classification.

Recent research has proposed hybrid quantum-classical models to improve learning capacity and robustness. Soja Rani et al. [19] combined transfer learning (VGG16, ResNet) with a Hybrid Quantum-Classical CNN (HQCCNN) using parameterized quantum circuits, achieving 97.11% accuracy on Malimg. Similarly, Akash et al. [2] introduced a Quantum Convolutional Neural Network (QCNN) for detecting malware on smart grid devices, significantly reducing training time while improving the F1-score of up to 99.4%. Beyond malware, Akter et al. [3] showcased a case-study-driven quantum support vector machine (QSVM) that hit 95% accuracy on the Drebin215 dataset. Alluhaibi [4] reported that QML-NN and QML-SVM models gave up to 92.5% accuracy and outperformed classical models in both accuracy and speed for threat detection.

The positive impact of QML was also noticed in broader systems. Abreu et al. [1] developed a QML-IDS system using Variational Quantum Circuits, shows competitive results even under real-world NISQ noise. In parallel, a hybrid quantum architecture designed for smart city cyber-physical infrastructure [21] demonstrated how QNNs can model high-dimensional threat vectors better. Gupta et al. [11] proposed a Quantum Fuzzy Neural Network (QFNN) framework for healthcare data protection, while Said et al. [20] proposed a Quantum Entropy Q-Learning (QEQ) model to defend against DDoS attacks in smart grids, outperforming classical models.
Adversarial Vulnerabilities and Defense Mechanisms: While advancements in machine learning have been significant, they have also highlighted the weaknesses of classical neural network models to various types of attacks. Adversarial attacks are a critical security concern where crafted perturbations are added to inputs to fool models. Szegedy et al. [26] showed that applying imperceptible perturbations can lead to misclassification. Goodfellow et al. [10] introduced FGSM, a single-step perturbation method that sharply reduces accuracy, while Madry et al. [15] later introduced PGD and proposed adversarial training to restore robustness.

Crucially, QML presents promising advantages, but its robustness against adversarial attacks remains a growing concern. Lu et al. [14] showed that even QML models are vulnerable to adversarial

perturbations, proving that small changes can fool QNNs just like in classical ML. Additionally, Gong and Deng [9] proved the existence of universal adversarial examples capable of fooling multiple quantum classifiers with exponentially small perturbations.

Some researchers suggest that quantum properties can inherently improve robustness. For instance, quantum noise itself has been shown to protect quantum classifiers from adversarial attacks [6]. Suryotrisongko et al. [25] quantified this by evaluating a hardened hybrid model for botnet detection. They found that while adversarial perturbations could degrade quantum model accuracy by 19%, injecting adversarial training with randomized quantum circuits recovered an average robustness gain of 5.9%. The results motivate the exploration of adversarial training.

Overall, QML-based and hybrid models are gaining traction for malware detection and cybersecurity. However, most of them do not explicitly handle adversarial attacks or leverage class-wise feature priors. Although recent defense mechanisms align with the motivation of improving robustness, they often lack specificity in the quantum latent space. In contrast, our work systematically examines the practical advantage of a quantum processing layer within a hybrid pipeline and validates the effectiveness of latent-space regularization and adversarial defense mechanisms. The QSLP mechanism guides quantum feature gradients towards alternate class centers, improving robustness and generalization in malware detection tasks, and adversarial training prepares the model for pixel-level attacks.

3 System Design and Methodology

In this section, we propose the QSLP architecture to overcome the challenges discussed previously. The main objective of the QSLP is to integrate both QNI-CCP and adversarial training. QNI-CCP method introduces quantum-informed perturbations in feature space, not input space and implements centroid-based regularization for stable feature learning and note results. As shown in Fig. 2, the architecture of QSLP comprises five stages described below.

3.1 Dataset Preparation

For our research and model training, we used the Malimg and the Malevis datasets, which are widely adopted for image-based malware classification. Malimg has 9,339 grayscale images representing malware samples, which are derived by converting Windows malware binaries into a visual 2D image format. The dataset includes 25 different malware families. The family-wise distribution is highly imbalanced, as some families contain thousands of images while others have fewer than 100 images. This imbalance represents a real-world challenge and necessitates strategies like weighted sampling or class-balanced loss functions.

On the other hand, the Malevis dataset (modified dataset) contains a total of 14,226 images with 25 malware families and 1 benign class, each having 400-500 samples. For our experiments, we used a pre-processed version of the dataset in which images were resized to 300×300 pixels to ensure input consistency for our CNN. Although the Malevis malware images are in RGB, we converted them to grayscale to maintain uniformity for fair comparison. Malevis images encode richer semantic information such as opcode density, entropy variations, and behavioural cues across channels, resulting in higher intra-class variability and increased complexity. We partitioned the dataset into training (70%), validation (15%), and test (15%) subsets to enable model optimization and unbiased performance evaluation.

Table 1: Nomenclature and Symbol Descriptions

Symbol	Description
Classical & QNI-CCP Parameters	
$x \in \mathbb{R}^d$	Input feature vector of dimension d
y	Ground-truth class label
$f(x)$	Output of the trained model
$\mathcal{L}$	Loss function (e.g., Cross-Entropy)
S	Sensitivity vector (gradient of loss w.r.t. input features)
z	Latent feature representation (embedding) from CNN
μ_c	Centroid (mean feature vector) for class c
c'	Selected target class for adversarial perturbation
Δ_{base}	Base perturbation direction ($\mu_{c'} - z$)
δ	Sensitivity-weighted perturbation vector
$\odot$	Hadamard (element-wise) product
ϵ_q	Quantum-Informed scaling factor
α, β	Hyperparameters tuning the quantum scaling factor
δ'	Final scaled perturbation added to the feature vector
z'	Final perturbed feature vector
Quantum Circuit Parameters	
n	Number of qubits ($n = 6$ for Malimg, $n = 10$ for Malevis)
θ_i	Trainable rotation parameter (angle) for qubit i
$R_Y(\theta)$	Single-qubit rotation gate around the Y-axis
$\lvert 0\rangle, \lvert 1\rangle$	Computational basis states
$\lvert \psi^{(0)} \rangle$	Initial quantum state after data encoding
N_{CNOT}	Total number of CNOT (entangling) gates in the circuit
D	Depth of the quantum circuit (number of layers)
L	Total number of variational layers
ℓ	Index for the current variational layer
CNOT	Controlled-NOT gate (entangling operation)
$\lvert \Psi^{(L)} \rangle$	Final evolved quantum state after L layers
Z_i	Pauli-Z observable operator acting on qubit i

3.2 Convolutional Feature Extractor/Reducer:

This acts as a bridge between high-dimensional nonlinear raw data and low-dimensional input space required by quantum circuits, as qubits are limited in quantity. We use a CNN to convert the malware images into a compact latent space. Unlike Principal Component Analysis (PCA), CNNs are effective in capturing complex nonlinear data, which are common in visual data such as images. While PCA captures linear data more effectively.

The module applies a series of convolutional layers, each of which

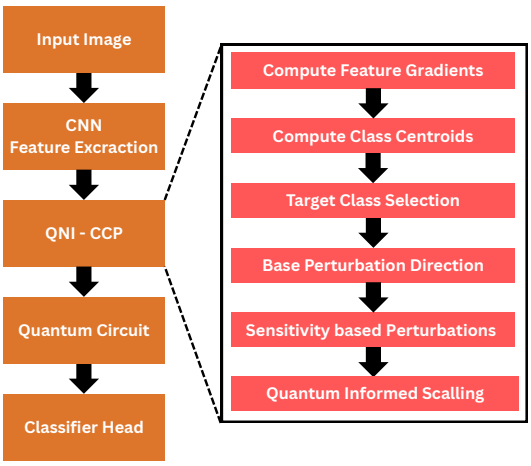


Figure 2: QSLP framework pipeline with QNI-CCP stages

is followed by steps for regularization, normalization, and nonlinear activation. Here, we used 6 layers and increased channels from 8 to 24 in all experiments. In addition to improving feature expressiveness, these operations aid in maintaining training stability and managing overfitting. The feature maps' spatial resolution is systematically decreased while maintaining pertinent semantic information by employing progressively downsampling transformations (such as strided convolutions). To reduce the dimension of inputs to 4×4, we used a stride of 2.

Regardless of the initial input resolution, an Adaptive Average Pooling operation is used at the end to compress spatial information into a 1×1 output. However, for the Malevis dataset, we had to expand ($224 \longrightarrow 2 \times N_{qubits}$) and project to catch more complex patterns. This pooling mechanism generates a spatially invariant feature descriptor and guarantees that the module can generalize across input variations. After that, a fully connected transformation is applied to the pooled output, creating a compact latent embedding that acts as the interface.

3.3 Quantum Noise Injection & Class Center Perturbation (QNI CCP):

It is well established that most of the classical adversarial training takes place in pixel space, as attacks like FGSM [10] and PGD [15] add perturbations directly into the pixel values of input images. While this can be effective for adversarial training in classical neural networks, it has a major issue in a hybrid quantum-classical pipeline. Quantum circuits today have very limited qubits, so raw images of large shapes must first be compressed into low-dimensional feature vector so that it matches the number of qubits defined in our quantum circuit before entering the quantum circuit layer. During the feature extraction stage, we could add pixel space perturbations, which may change the image details, but due to downsampling and pooling operations on CNNs, the effect is significantly reduced. As a result of this, a quantum circuit may never see much of the adversarial noise, which can give a false sense of robustness. To truly harden a hybrid model, adversarial perturbations must be applied after the classical encoder and before the quantum processor, in the same low-dimensional space where quantum gates act. This setup will directly address RQ2. The perturbations that are added to the input feature space are not random; they are properly curated to simulate adversarial training in a more meaningful and controlled way. As a result, the model learns more robust features and achieves better generalization. Correspondingly, the overarching goal of QNI-CCP is to:

- Inject targeted, class-driven perturbations into each input to bias representations toward the centroid of an alternative malware class.
- Concentrate perturbations on the most loss-sensitive features using gradient-based weighting to maximize deflection in the model output.
- Scale the perturbation magnitude according to the expressive capacity of the quantum circuit, as determined by its depth and degree of entanglement.

We now describe the steps and stages involved in QNI-CCP.

3.3.1 Feature Gradient Sensitivity mapping: The first component basically detects the importance of individual input features with respect to the model's loss function. Given an input feature vector $x \in \mathbb{R}^d$, with true label y, and a trained model $f(x)$, the classification loss is denoted by $\mathcal{L}(f(x), y)$, typically the cross-entropy loss. The gradient of loss with respect to the input features is computed as:

$$S = \nabla_x \mathcal{L}(f(x), y) \tag{1}$$

This results in a sensitivity vector $S \in \mathbb{R}^d$, where each component S_i quantifies the degree of loss of sensitivity to changes in the corresponding input feature x_i. Intuitively, large magnitudes in S indicate features that significantly influence the prediction of the model.

To ensure numerical stability and control the scale of the perturbation, the sensitivity vector is either normalized using the ℓ_2-norm or transformed using the element-wise sign function:

$$S' = \frac{S}{\|S\|_2} \quad \text{or} \quad S' = \text{sign}(S) \tag{2}$$

This sensitivity map is later used to weight perturbations in a feature-specific manner.

3.3.2 Class- Centroid Computation: To gain a compact, class-level summary of how a given class is represented in the low-dimensional feature space produced by the CNN encoder, we compute the centroid, which is the mean latent feature representation for each class in the dataset.

Let $z_i = f_{\text{feat}}(x_i) \in \mathbb{R}^d$ where d-dimensional latent feature embedding of an input image x_i, where f_{feat} is the classical CNN-based encoder. For each class c, the centroid $\mu_c \in \mathbb{R}^d$ is calculated as the average of all feature vectors z_i corresponding to the training samples in class c:

$$\mu_c = \frac{1}{N_c} \sum_{i\,:\,y_i=c} z_i \tag{3}$$

Here, N_c denotes the total number of samples belonging to class c, and y_i is the ground-truth label of the i^{th} sample. Instead of analyzing all the individual embeddings, we summarize each class with a single vector that represents the "mean" position of its samples.

This helps us measure intra-class compactness and inter-class separability, and is especially useful in our QSLP framework to design meaningful perturbations.

3.3.3 Target Class Selection and Base Perturbation Direction: After our CNN encoder and class centroid computation, we get a mean feature representation, a compact feature vector (x), and we know the true class x – assume class A. To test the model's robustness, we randomly pick a different class, assume class B. We then measure the gap between x and the centroid of class B. By selecting a random class each time, we force the model to learn to resist confusion against all possible misscalssification, not just one specific mislabel. It not only makes the model robust against random noise but also against semantically structured deviations.

Let z be the feature representation of an input image x, which is done by the CNN encoder. For each training instance, we randomly sample a target class c' such that $c' \neq y$, where y is the ground truth label of x. Then we calculate the difference between the centroid of the selected target class and the embedding of the feature z. This base vector reflects the direction a sample would move in latent space if it were to be gradually transformed into a representative example of the target class. Mathematically, this is expressed as:

$$\Delta_{\text{base}} = \mu_{c'} - z \tag{4}$$

3.3.4 Sensitivity-Weighted Perturbation: In adversarial robustness or perturbation-based testing, not all features contribute equally to the output of the model. Due to high sensitivity, some features have a stronger influence. As a result, even a small change in such dimensions will change the model's decision.

Let:

- $z \in \mathbb{R}^d$: an input sample's feature representation.
- The centroid of a target class c', calculated over feature vectors of class c', is represented by $\mu_{c'} \in \mathbb{R}^d$.
- $\Delta_{\text{centroid}} = \mu_{c'} - z$: the direction vector that points from the feature vector of the current sample to the target class centroid.
- $S = \frac{\partial L}{\partial z} \in \mathbb{R}^d$: the sensitivity of each dimension is represented by the absolute gradient of the loss L with respect to z.

Now, we define sensitivity-weighted perturbation δ as:

$$\delta = S \odot (\mu_{c'} - z) \tag{5}$$

Intuition of above equation is that we don't want to push our feature vector in directions that are not very sensitive or useful for contributing to the model's decision. We want to do it intelligently, by applying a large perturbation in directions that are more sensitive or influential in determining the outcome. Here, $(\mu_{c'} - z)$ represents the directional vector that tells us in which direction and how much we need to move our current point to reach the center of the target class.

3.3.5 Quantum Informed Scaling: Scaling based on Circuit Complexity: Here, we scale the perturbation that was obtained in Equation 5 by the quantum factor. The core motivation behind this is that classical neural networks are more expressive and resilient to big perturbations, while quantum circuits, especially near-term NISQ ones, are capacity-limited. They cannot handle large feature shifts without losing stability. Thus, the perturbations injected into the feature space need to be i) subtle and smooth, ii) should be within the expressive power of a quantum circuit and iii) must be aligned with the circuit's structural complexity.

This motivates the introduction of Quantum-Informed Scaling. Owing to the entanglement properties of quantum circuits, highly complex circuits are more sensitive to small variations in their inputs, as perturbations affecting one qubit can propagate throughout the entangled system. Therefore, our Quantum Informed Scaling dynamically adjusts the magnitude of the perturbation applied to the feature vector z based on the complexity of the quantum circuit. As circuit complexity increases, measured in terms of depth and degree of entanglement, the applied perturbation is correspondingly reduced, reflecting the increased fragility and reduced robustness of complex quantum circuits to abrupt input variations.

The vector δ is scaled by Quantum-Informed factor ϵ_q

$$\epsilon_q = \frac{1}{1 + \alpha \cdot N_{\text{CNOT}} + \beta \cdot D} \tag{6}$$

Note: The constants α and β are tunable and represent how strongly the number of CNOT gates and circuit depth affect the final scaling. In practice, set $\alpha = \beta = 1.0$ and tune further based on validation performance.

3.3.6 Apply the Perturbation: From the above equations and the perturbation derived from equation 5, the scaled perturbation obtained using the Quantum Informed factor will now be added to the original feature vector, and it only applies a large perturbation in directions that actually matter and makes the model more robust against other classes available in the dataset. Thus, the final perturbation applied is:

$$\delta' = \epsilon_q \cdot \delta = \epsilon_q \cdot [S \odot (\mu_{c'} - z)] \tag{7}$$

And the perturbed feature becomes:

$$z' = z + \delta' = z + \epsilon_q \cdot [S \odot (\mu_{c'} - z)] \tag{8}$$

3.4 Quantum Circuit Architecture

The quantum layer is implemented as a Variational Quantum Circuit (VQC), which utilizes an underlying Parameterized Quantum Circuit (PQC) structure. PQCs are quantum circuits where gate parameters are treated as trainable variables. These parameters are tuned by a classical optimizer in a hybrid quantum-classical network to achieve a desired output from the quantum circuit. Similar to the weights in classical neural networks, parameterized quantum circuits employ trainable gate parameters. Hybrid quantum-classical models exploit near-term quantum processors to generate nonlinear feature representations that complement classical learning architectures. In our framework, we integrate a six-qubit quantum layer between a convolutional feature extractor and a classical classifier. This hybrid setting allows us to evaluate the benefits of adding the quantum layer (RQ1). The quantum circuit is structured into three components: (i) a data-encoding layer, (ii) a variational ansatz, and (iii) a measurement stage as described below.

3.4.1 Data Encoding Layer: Each input sample is a 6-dimensional (Malimg) or 10-dimensional (Malevis) feature vector obtained from

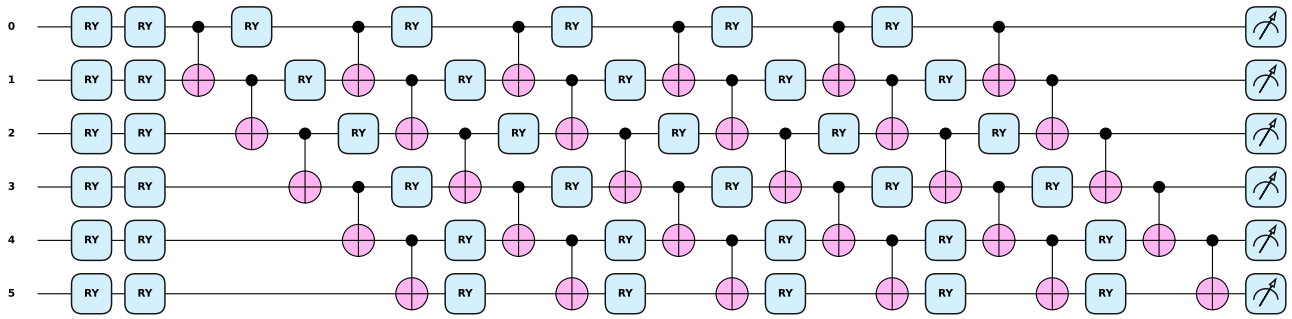


Figure 3: 6-Qubit Variational Quantum Circuit for Malimg dataset

the classical feature extractor. We embed these real data into the quantum state using angle encoding. Angle encoding uses parameterized rotations to imprint a classical vector onto n-qubit register. It uses single-qubit rotation gates (like R_x or R_y) to encode each feature of the classical data. We decided to employ 6 qubits for our Malimg model and 10 qubits for our Malevis model, after much testing and training, illustrated in the section 4.1. The practical restrictions of present quantum technology, where the number of available qubits is frequently limited, as well as performance considerations, played a role in this decision. In order to assess their effects on scalability and performance, we also investigated systems with 4 and 8 qubits throughout our experimentation phase. The classification model of the Malevis dataset was unable to classify preciously with 6 qubits, as the dataset has many complex patterns to learn. Figure 3 provides a visual representation of the quantum circuit used in the experiment with the Malimg dataset, where the circuit depth = 6 and qubits = 6 were used for data representation.

$$R_Y(\theta_i) = \exp\left(-i\,\frac{\theta_i}{2}Y\right) = \begin{pmatrix} \cos\frac{\theta_i}{2} & -\sin\frac{\theta_i}{2} \\ \sin\frac{\theta_i}{2} & \cos\frac{\theta_i}{2} \end{pmatrix} \tag{9}$$

$$R_Y(\theta_i)\,|0\rangle = \cos\tfrac{\theta_i}{2}\;|0\rangle \;+\; \sin\tfrac{\theta_i}{2}\;|1\rangle \tag{10}$$

$$\left|\psi^{(0)}\right\rangle = \bigotimes_{i=0}^{n-1} R_Y(\theta_i)\,|0\rangle = \sum_{x\in\{0,1\}^n} \left(\prod_{i=0}^{n-1} \cos\left(\frac{\theta_i}{2}\right)^{1-x_i} \sin\left(\frac{\theta_i}{2}\right)^{x_i}\right) |x\rangle \tag{11}$$

Equation 9 shows the application of parameterized $RY(\theta_i)$ rotations to each qubit initialized in the state $|0\rangle$. These rotations encode learnable parameters into the quantum circuit by modifying the amplitudes in the computational basis. Equation 10 represents the tensor product of the six single-qubit rotations, resulting in a six-qubit quantum state $\left|\psi^{(0)}\right\rangle$. This state captures the combined effects of the parameterized gates applied across all qubits. Equation **??** expands the tensor product into a superposition of all possible 2^n computational basis states $|x\rangle$, where each amplitude is a product of $\cos(\theta_i/2)$ or $\sin(\theta_i/2)$ terms depending on the bitstring $x \in \{0,1\}^n$. This expansion reveals how the parameter vector θ influences the probability amplitude of each basis state.

3.4.2 Variational Ansatz: The variational layers play a major role in learning feature representations through parameterized quantum circuit operations. Each variational layer l of our quantum circuit consists of two sub-operations: local rotations and entangling layers, both of which are parameterized and contribute in model's decision. These layers are designed to operate just like dense layers in CNNs, but with the quantum expressivity of the QNN.

For layer l:

1. Qubit Rotations: For each qubit in layer l, we apply a rotation around Y-axis of the Bloch sphere, paramterized by a trainable angle $\theta_{\ell,i}$. This rotation modifies the amplitude of the quantum state on the Bloch sphere, enabling the model to explore a richer subspace of the Hilbert space. For each qubit $i \in \{0,\ldots,n-1\}$, in layer ℓ, the variational quantum state is defined as:

$$\left|\phi_i^{(\ell)}\right\rangle = R_y(\theta_{\ell,i})\left|\psi_i^{(\ell)}\right\rangle, \quad \forall i \in \{0,\ldots,n-1\} \tag{12}$$

where $\theta_{\ell,i}$ is a trainable rotation parameter for qubit i in layer ℓ.

2. Entanglement layer: Following the local rotations around Y axis, the variational circuit includes entanglement among qubits as it allows the network to capture quantum correlation between qubits. Our entanglement layer applies linear entanglement pattern which is chosen for its simplicity and low hardware overhead. Specifically, we apply a chain of CNOT gates between consecutive qubits:

$$\left|\phi^{(\ell)}\right\rangle \longmapsto \left(\prod_{i=0}^{4} \mathrm{CNOT}_{i\to i+1}\right)\left|\phi^{(\ell)}\right\rangle \tag{13}$$

3.4.3 Measurement: After l layers of parameterized gates and entanglement operations, the quantum states are evolved and ready for measurement. The output quantum state is denoted as $\left|\Psi^{(L)}\right\rangle \in \mathbb{C}^{2^n}$, where n is the number of qubits ($n = 6$ *or* 10). To extract information from these quantum states, we measure all qubits. Specifically, for each qubit i, we compute the **expectation value** of the Pauli-Z observable Z_i, which corresponds to measuring the spin projection along the Z-axis of the Bloch sphere.

The measurement for each qubit is given by:

$$z_i = \left\langle \Psi^{(L)}\right| Z_i \left|\Psi^{(L)}\right\rangle, \quad i = 0,1,\ldots,n-1$$

Algorithm 1 Proposed Robust training in QSLP

Require: train_loader, val_loader, model, opt, sched, num_epochs, ε_q, fgsm_eps, pgd_eps, pgd_alpha, pgd_iters
Ensure: trained model

centroids ← compute_class_centroids(model, train_loader)
for epoch = 1 **to** num_epochs **do**
  if epoch mod 5 = 0 **then**
    centroids ← compute_class_centroids(model, train_loader)
  end if
  for all batch (x, y) in train_loader **do**
    logits_clean ← model(x)
    loss_clean ← CE($logits_clean, y$)
    features ← tanh(feature_extractor(x))
    pert_features ← QNI_CCP_perturb(model, x, y, centroids, ε_q)
    q_out ← stack(q_layer(f) **for** f in pert_features)
    logits_qni ← classifier(q_out)
    loss_qni ← CE($logits_qni, y$)
    x_fgsm ← FGSM(model, x, y, fgsm_eps)
    loss_fgsm ← CE($model(x_fgsm), y$)
    x_pgd ← PGD(model, x, y, pgd_eps, pgd_alpha, pgd_iters)
    loss_pgd ← CE($model(x_pgd), y$)
    centroid_reg ← $\|$features − centroids$[y]\|^2$
    loss ← 0.5 * loss_clean + 0.15 * loss_qni
      + 0.10 * loss_fgsm + 0.15 * loss_pgd
      + 0.10 * centroid_reg
    opt.zeroGrad()
    loss.backward()
    clipGrad(model)
    opt.step()
  end for
  val_acc ← evaluate(model, val_loader)
  sched.step(train_loss)
  if val_acc improves **then**
    save checkpoint
  end if
end for

3.5 Classifier Head

The classifier head serves as the final decision-making module for the model, as it transforms the quantum-rich features into a classical feature vector, used for further transformations and feature space reduction, and later converts them into final class predictions. Initially, the obtained quantum feature space from the quantum circuit is rich with entanglement and are raw expectation value, which is further converted into class predictions using a classifier head. This module carries out a number of tasks designed to hierarchically refine, compress, and transform these quantum features. It is made up of a number of dense, fully connected layers that are separated by regularization dropout and nonlinear activation functions (ReLU). The design decision to gradually reduce the feature space's dimensionality (i.e., from 64 → 32 → 16 → output) aids in highlighting the most discriminative features while lowering noise.

3.6 Adversarial training

Results obtained after QNI-CCP training reflect the model's vulnerability to pixel-level attacks (adversarial attacks), addressing RQ2. To answer RQ3, we incorporated QNI-CCP, FGSM and PGD; all attacks into the loss calculation. Algorithm 1 is the pseudo-code.

4 Evaluation

4.1 Baseline Model Performance

First, we evaluated basic hybrid CNN+QNN pipeline models and then enhanced them step by step through a detailed analysis of the results. The initial experiments were conducted on the Malimg dataset to understand the impact of each parameter and hyperparameter. The model with 64×64 dimensions and 6 qubits with 2 quantum layers achieved only 43% training accuracy with approximately 20% validation accuracy, indicating poor generalization and overfitting issues due to class imbalance. We scaled the images to 128×128 dimensions for better learning. Training with 2 quantum layers achieved about 77% accuracy with 55% validation accuracy. However, the performance remained poor and showed limited generalization. Replacing CrossEntropy with FocalLoss mitigated the class imbalance issue, and adding momentum with an additional quantum layer improved pattern learning and resulted in training accuracy of about 90%. Several classes that performed poorly were noted in the classification report. As an experiment, we implemented class-based oversampling, but it didn't improved accuracy. The experiment highlighted the critical role of architecture in performance. Table 2 gives information about a few experiments that we performed on the model and the results observed. During experiments, the number of qubits was 6. The results suggest that increasing the depth of the quantum circuit improves performance. In contrast, implying a Ring-CNOT entanglement pattern significantly reduced the accuracy from 95% to 81%.

Table 2: Hyperparameter Tuning and Architecture Optimization steps - Malimg.

Configuration Changes	Image Dim	Q-Layers	Loss Function	Accuracy
Baseline (initial config)	64^2	2	CrossEntropy	43%
Increased Input Resolution	128^2	2	CrossEntropy	77%
Loss Function change (Class Imbalance)	128^2	2	FocalLoss	80%
Added momentum & tuned alpha/beta	128^2	3	FocalLoss	90%
Ring CNOT Entanglement	128^2	6	FocalLoss	81%
Increased Q-Depth (Best Model)	128^2	**6**	**FocalLoss**	**95%**

Similarly, on the Malevis dataset, we performed many experiments by changing the values of different parameters. Initially, we applied the same hybrid architecture used for the Malimg. However, the model achieved only 33% accuracy, suggesting that complex patterns of Malevis require higher dimensionality, so major changes were required in the input dimension and qubits. Table 3 provides details of the exact architecture used for the Malevis dataset to ensure reproducibility of experiments. To better understand feature representation quality, we analyzed latent-space and frequency.

Examining the primary findings, interestingly, when the Classical CNNs for both datasets were trained on the same classification architecture, they showed very similar results. At this stage, adding a quantum layer seemed to be of no advantage, but the reality was uncovered in subsequent experiments.

The values mentioned in the table 4 were obtained from a variance-based analysis on the latent space. PCA was applied to latent vectors across the training set to compute the variance ratio of each principal component. However, solely relying on this can be misleading as PCA is an unsupervised technique. The results suggest that most variance can be captured within 4-6 principal components; however, variance preservation does not guarantee semantic separability, as confirmed by previous experiment results. The primary reason behind this could be the similarity in the global visual pattern. Moreover, adversarial conditions indicated increased variance dispersion, justifying the need for higher qubits for robustness and correctness in predictions.

Table 3: Summary of Experimental Settings for Malevis Dataset with QSLP Architecture

Category	Configuration
Dataset	Malevis (image-based malware dataset)
Image Resolution	224×224
Normalization	Mean = 0.5, Std = 0.5
Training Augmentation	Horizontal flip, rotation ($\pm 10°$)
Latent Feature Dimension	10
Number of Qubits	10
Embedding Strategy	Angle encoding (RY gates)
Circuit Depth	6 variational layers
Entanglement	Linear CNOT chain
Quantum Output	Pauli-Z expectation values
Optimizer	AdamW
Learning Rate	1×10^{-4} to 5×10^{-4}
Batch Size	16
Weight Decay	5×10^{-3}
Loss Function	Weighted FocalLoss ($\gamma = 1.0$)
Class Imbalance Handling	Balanced class weights
QNI-CCP Parameters	$\alpha = 1.0,\ \beta = 1.0$
Adversarial Attacks	FGSM, PGD
FGSM ϵ	0.03
PGD ϵ	0.1
PGD Step Size	0.01
PGD Iterations	7

Table 4: Variance Ratio of Principal Components in various conditions

Dimensions	1	2	3	4	5	6	7	8	9	10
Malimg	0.50	0.37	0.067	0.04	0.02	0.00	–	–	–	–
Malevis	0.80	0.15	0.03	0.02	0.005	0.003	0.001	0.00	0.00	0.00
				(On Adversarial Input $\epsilon = 0.3$)						
Malimg	0.41	0.21	0.16	0.10	0.06	0.05	–	–	–	–
Malevis	0.67	0.15	0.06	0.04	0.03	0.02	0.01	0.0086	0.0021	0.00

4.2 Evaluation Metrics

Since the models underwent various types of training, comprehensive evaluation metrics are required to judge their performance. In this section, we discuss about accuracy, precision, recall and f1-score, which are the most common parameters used to judge the model's classification ability on a prediction level (output level). Beyond prediction accuracy, analyzing the model's decision parameters in latent space provides insight into its behaviour. Latent-Space-level features such as Entropy, Confidence, and feature variance gives insights about how the model thinks after the feature extraction, discussed in subsection 4.4.

Accuracy alone can be misleading when the dataset is class-imbalanced. For example, if there are 50% data for a certain class, the model can achieve 50% accuracy by predicting only one class. The F1-score is the harmonic mean of precision and recall, providing a more reliable evaluation under class imbalance.

Table 5 provides evaluation metrics for successfully implementing different models. After QNI-CCP training, the evaluation scores showed no improvement in both datasets. QNI-CCP training simulates class-aware conditions during training and controlled perturbations in feature space that result in improved generalization. Adversarial training significantly improved the training accuracy in the Malevis dataset.

Table 5: Performance of the hybrid QNN model under different training stages for both datasets.

Model Info	Accuracy	F1-score	Precision	Recall
Dataset 1: Malimg				
Learning rate: 0.0005 (adaptive) Optimizer: Adam with momentum Loss function: FocalLoss Quantum layers: 6 Weight decay: 0.005	0.89 (t) 0.95 (v)	0.88 (m) 0.95 (w)	0.89 (m) 0.95 (w)	0.89 (m) 0.96 (w)
After QNI-CCP	0.97 (t) 0.97 (v)	0.93 (m) 0.97 (w)	0.93 (m) 0.96 (w)	0.94 (m) 0.97 (w)
After Adversarial training and QNI-CCP	0.96 (t) 0.95 (v)	0.87 (m) 0.93 (w)	0.90 (m) 0.94 (w)	0.88 (m) 0.94 (w)
Dataset 2: Malevis				
Hybrid QNN model	0.87 (t) 0.80 (v)	0.87 (m) 0.84 (w)	0.86 (m) 0.88 (w)	0.90 (m) 0.83 (w)
After QNI-CCP	0.86 (t) 0.79 (v)	0.85 (m) 0.82 (w)	0.85(m) 0.88 (w)	0.88 (m) 0.80 (w)
After Adversarial training and QNI-CCP	0.92 (t) 0.81 (v)	0.85 (m) 0.83 (w)	0.85 (m) 0.87 (w)	0.88 (m) 0.82 (w)

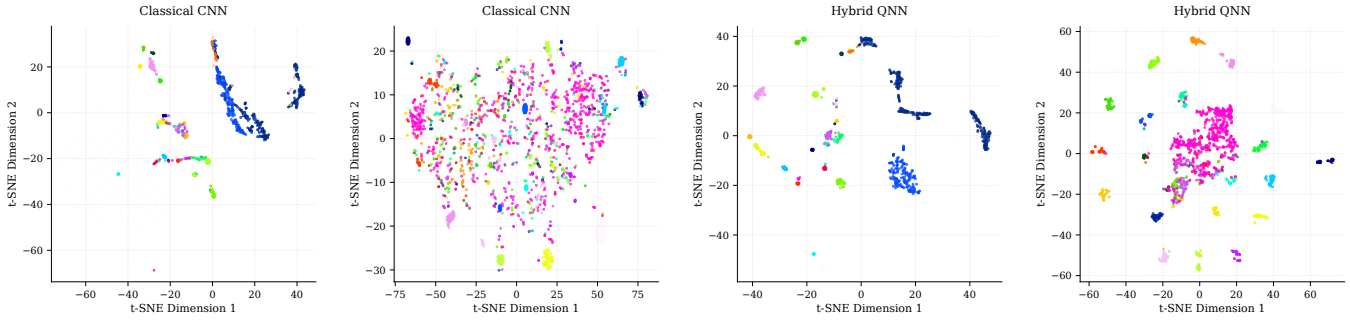
Note: t – training, v – validation, m – macro, w – weighted

4.3 Additional training

As discussed earlier, to improve robustness against various attacks, and evaluate its impact on evaluation metrics, we applied two main methods: (1) Noise Injection and (2) Adversarial Training

Firstly, to evaluate potential quantum-advantage (RQ1), we visualized feature-space with t-SNE plots, and here the results given by quantum models were quite impressive. The t-SNE (t-Distributed Stochastic Neighbor Embedding) plot is used to show higher dimensional data in lower dimensions like 2D or 3D. The patterns are more clearly visible in Figure 4. Figure 4a illustrates the t-SNE projection of the feature representations learned by the classical model, showing how the samples cluster in the latent space. In contrast, Figure 4b shows the t-SNE projection for the quantum-enhanced hybrid model, where the class clusters are more compact and better separated, indicating improved representational structure. This behaviour is consistent across both datasets and provides evidence of enhanced classification ability. These visualisations suggest that hybrid quantum-classical models can process data better than classical models even before QNI-CCP and adversarial training.

In the first set of experiments, we evaluated the impact of QNI-CCP as a standalone robustness mechanism. Unlike normal gradient-based noise injection, QNI-CCP introduces structured perturbations in the feature-space guided by class centroids and model sensitivity. It encourages smooth and stable representations of features in the latent-space from the early stages of training. Later, to make models robust against this attack, we performed adversarial training. We implemented QNI-CCP and adversarial training by modifying



a Classical Models: The overlapping clusters in both datasets reflect high feature-space variance and low class separation, resulting in reduced robustness against adversarial attacks. The feature space visualizations for both models are compared on the Malimg (left) and Malevis (right) datasets.

b Hybrid QNN (prior to QNI-CCP): The clusters show clearer separation between malware families, showing structured feature learning and improved discriminative ability. The advantage demonstrated across datasets, the Malimg (left) and the Malevis (right).

Figure 4: Comparison of feature space representation using two-dimensional t-distributed Stochastic Neighbor Embedding (t-SNE 2) for Classical and Quantum Models.

loss functions that accumulate loss from all methods. For the Malimg dataset, it achieved 96% training accuracy after 23 epochs of training, and for the Malevis dataset, it achieved 92% training accuracy. Such impressive results motivated us to propose the QSLP architecture for hybrid quantum-classical models.

4.4 Classical vs. Quantum model learning

In this section, we quantify and compare the results of the QNI-CCP trained model with previous models. We compute the confidence metric to assess how certain the models are in their predictions. For High-confidence patterns, 90% is the threshold.

Entropy: It measures how evenly the model uses its learned feature patterns for predictions. Higher entropy suggests features are used more uniformly despite being dominated by a few strong features.
Feature Variance: It provides information about the spread of activation strengths across features. low variance suggests more balanced feature usage.
Unique Patterns: It is the number of distinct activation configurations the model produces across the dataset.
High-confidence Patterns: The percentage of predictions where the model's confidence (probability) is above a set threshold (≥ 0.90). A higher value suggests correct predictions.
Average Confidence: The mean prediction probability assigned to the predicted class across all samples, indicating the model's overall certainty in predictions.
Confidence Standard Deviation: Measures how much the model's confidence varies across samples. Low values suggest consistent confidence across different parameters.

Table 6 shows the key change in Feature Variance across both the datasets. The lower feature variance indicates patterns are more uniformly used rather than dominated by heavy parameters. Feature variance scores for QNNs are significantly lower compared to the classical CNN models, depicting patterns are more evenly distributed in the classical models, which hinders the robustness. Noticeably, the number of unique patterns for both models of both datasets remain the same, but the arrangement of features in feature space differs, as shown in Figure 4.

Interestingly, the Entropy and percentage of high-confidence patterns do not follow a common trend for both datasets because both datasets are significantly different from each other. The Malimg dataset has visually distinguishable classes, whereas the Malevis dataset has information like malware entropy and op-code density across different channels. However, average confidence and entropy jointly can provide insight into overconfident behaviour. In the Malimg dataset, the QNI-CCP model achieves the highest confidence, but there is a slight drop in entropy. We attribute the high confidence values to the uniform structure of the Malimg dataset. In the Malevis dataset, although the hybrid QNN slightly lowers entropy (3.0901) and confidence variance compared to QNI-CCP, the average confidence remains lower than that of the classical model. This suggests that robust regularization mechanisms may yield smoother and more conservative decision boundaries at the expense of predictive certainty in complex datasets like Malevis.

Table 6: Pattern learning analysis of Classical CNN, Hybrid QNN, and QNI-CCP across both datasets.

Metric	Malimg			Malevis		
	CNN	QNI	H-QNN	CNN	QNI	H-QNN
Entropy	2.5300	2.5215	2.5452	3.0966	3.1137	3.0901
Feature Variance	7.7084	**0.0996**	0.1274	0.3746	**0.0269**	0.0404
Unique Patterns	24	24	24	26	26	26
High-conf. patterns	78.1%	99.4%	73.7%	74.8%	61.3%	54.8%
Average confidence	0.936	0.996	0.910	0.885	0.819	0.818
Confidence std. dev.	0.107	0.027	0.128	0.204	0.244	0.199

Note:
H-QNN: basic hybrid quantum-classical model,
QNI: QNI-CCP trained hybrid quantum-classical model
CNN: Classical model used for classification

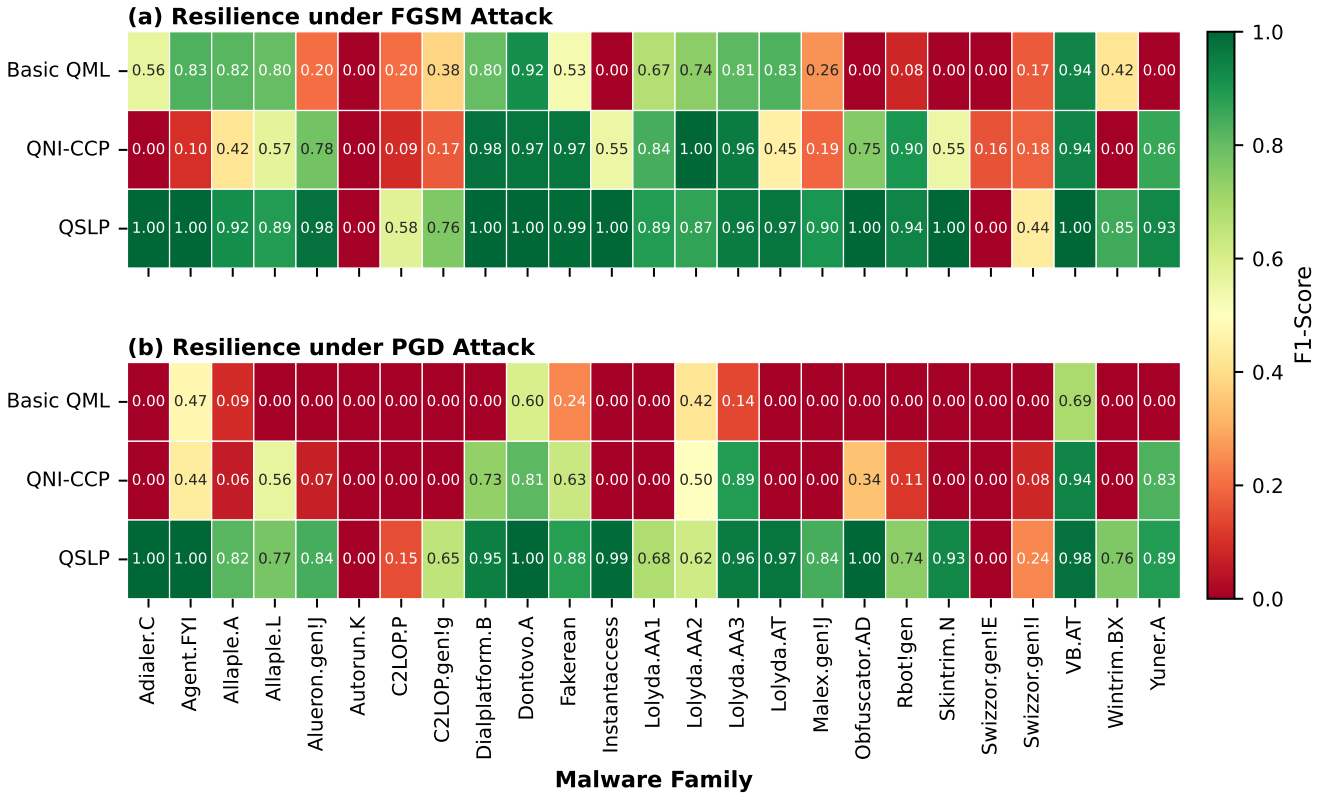


Figure 5: Class-wise Adversarial Resilience Analysis: The heatmaps compare the F1-score of the Basic QML model, QSLP model, and proposed Adversarially Robust model under (a) FGSM and (b) PGD attacks, for the Malimg dataset.

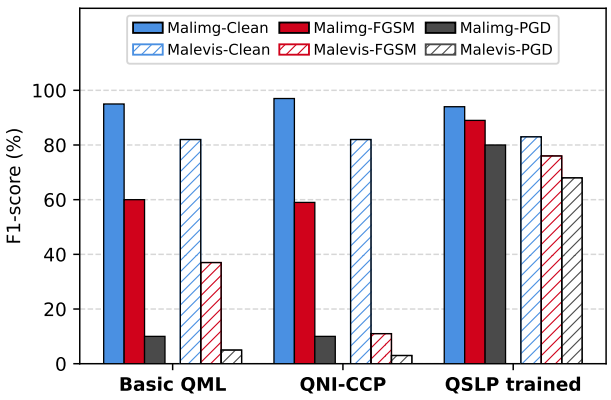


Figure 6: F1-score comparison for different classification models under adversarial attacks such as FGSM and PGD.

4.5 Adversarial Robustness

Finally, we evaluated models by performing two types of adversarial attacks - PGD and FGSM. In FGSM, the perturbation-budget (ϵ) was 0.03. While the PGD attack was performed with the following configuration: Maximum Perturbation (ϵ) = 0.1, Step Size (α) = 0.01 and Number of Iterations (k) = 7. FGSM applies a single-step perturbation in the direction of the gradient of the loss with some scaling. PGD can be explained by iterative refinements of perturbations, which allows exploration of weaknesses in decision boundaries.

Figure 5 provides a clear visualization of how f1-scores changed after applying QNI-CCP and QSLP, particularly after adversarial training. Adversarial training provided robustness to all models against FGSM and PGD attacks. A general trend across datasets is that QNI-CCP training is not sufficient against adversarial attack, but adversarial training is proven to be highly effective.

Figure 6 compares overall F1-scores under normal conditions and adversarial attacks. Basic classification models almost failed against the PGD attack as they showed less than 10% accuracy in both datasets. Models trained only with QNI-CCP also did not perform well (addressing RQ2), but the models trained with QSLP demonstrate superior resilience against both attacks after adversarial training. This addresses RQ3 by proving the effectiveness of adversarial training on our hybrid models. As discussed earlier, we also integrated QNI-CCP during adversarial training to make models robust at both pixel and latent-space levels.

5 Discussion

Our experiments demonstrate the potential for quantum advantage in AI and cybersecurity through the proposed QSLP framework.

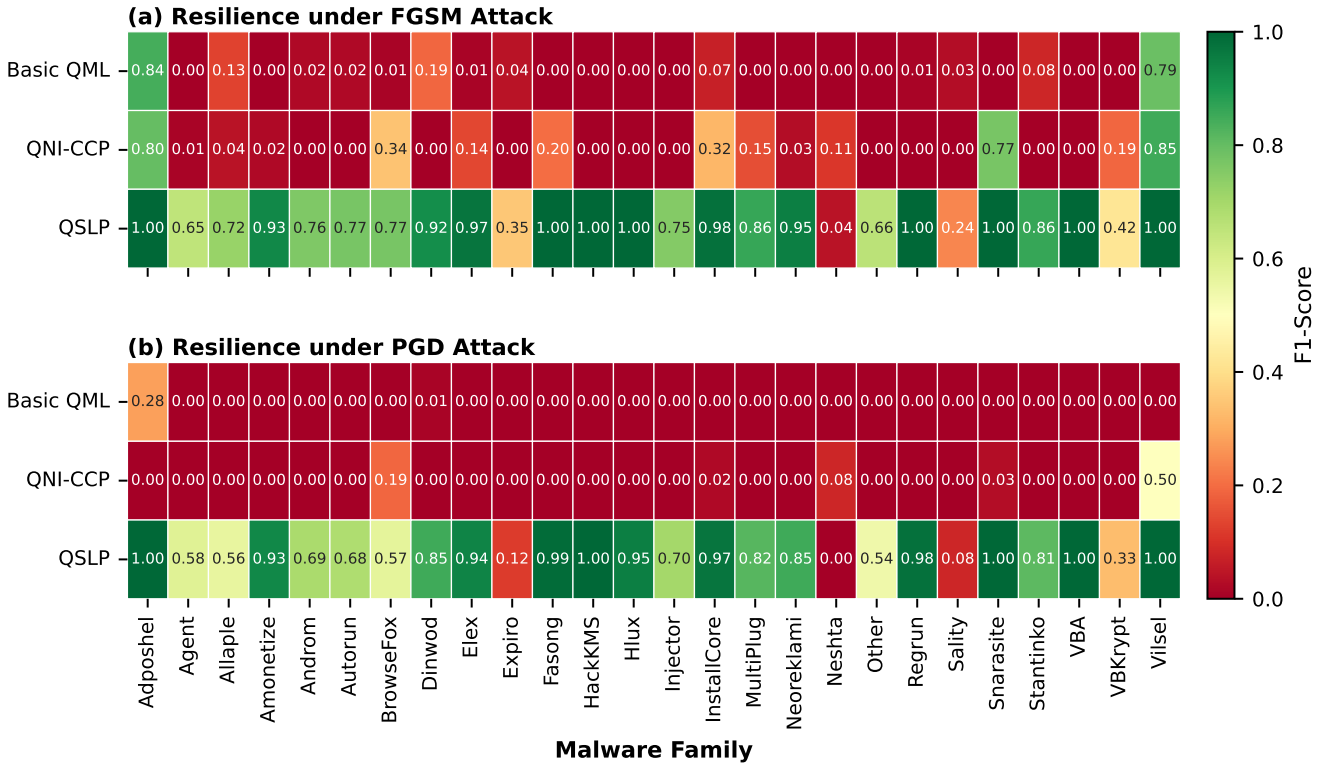


Figure 7: Class-wise Adversarial Resilience Analysis: The heatmaps compare the F1-score of the Basic QML model,QSLP model, and proposed Adversarially Robust model under (a) FGSM and (b) PGD attacks, for the Malevis dataset.

Nonetheless, certain gaps remain, and directions for future research are discussed in the following section.

Limitations: Although we show that our proposed QSLP, a hybrid quantum-classical pipeline, shows promising results, we would like to highlight the following limitations in the current work:

(1) **Simulation vs. real hardware:** All quantum experiments were performed with simulators and noise models. Real NISQ devices have device-specific noise, gate errors, and connectivity constraints that may degrade performance or require different circuit design and error mitigation strategies.
(2) **Limited qubit count and scalability:** The architecture uses a small number of qubits (6 and 10 in the reported runs). Scaling to larger qubit counts or deeper ansätze can increase expressivity, but it would also amplify the noise, calibration requirements, and runtime costs. Hence, these parameters must be carefully re-evaluated for more complex circuits.
(3) **Restricted adversary models:** The adversarial evaluation was limited to FGSM and PGD on the pixel-level. Other attacks (CW, DeepFool, transfer-based, gradient-free and adaptive attacks targeting centroid computation) were not evaluated and could reveal additional weaknesses.
(4) **Hyperparameter sensitivity and reproducibility:** QNI-CCP introduces tunable hyperparameters (e.g., α, β, scaling schedules, centroid update policies). Model performance is sensitive to these choices; a thorough hyperparameter study and code release are required to ensure reproducibility.
(5) **Computation and latency overhead:** Injecting feature-space perturbations, computing gradients *w.r.t.* latent features, and executing quantum circuit evaluations (shots) introduce significant computational and inference latency. Hence, for resource-constrained or real-time environments, these costs may limit practical applicability unless additional optimization strategies are employed.

Future Work: With the aim of complementing the current findings and informing further research on the proposed framework, we discuss the potential extensions of our proposed approach and outline directions for future research in this domain.

(1) **Generalization via Larger Datasets**: While the experiments demonstrated robustness on the Malimg and Malevis datasets, training on a more diverse dataset could enable the model to capture more complex patterns, such as obfuscated binaries or zero-day attacks.
(2) **Integration with Federated and Privacy-Preserving Learning:** Furthermore, a quantum-enhanced federated learning approach can be explored to develop robust malware detection models, enabling multiple organizations to collaborate.
(3) **Further Adversarial Robustness**: The model can be trained with other adversarial attack data like Carlini–Wagner (CW),

DeepFool, etc. Additionally, the QSLP framework can be extended to diverse applications such as phishing detection and network intrusion detection to establish its effectiveness.

(4) **Cross-Domain Applications of QML**: This proposed classifier can be used for applications beyond malware detection, such as financial forecasting, environmental modeling, or detecting manipulated multimedia, with higher accuracy.

(5) **Hardware-Aware Quantum Implementations**: While the current implementation relies on a quantum simulator, implementing the framework on real quantum devices represents a natural direction for future work.

6 Conclusion

This work addresses questions related to the benefits and adversarial robustness of hybrid quantum–classical models for image-based malware detection. Through comprehensive experimentation and result analysis, we successfully addressed the mentioned research questions. Our findings show that hybrid quantum–classical models learn more structured and discriminative latent representations than classical CNNs, but that latent-space regularization alone is insufficient to defend against strong adversarial attacks. We further demonstrate that adversarial training remains effective within hybrid pipelines and is essential for achieving robustness under both pixel-level and iterative adversarial perturbations.

We proposed QSLP, a hybrid quantum-classical architecture that integrates QNI-CCP with adversarial training. Experiments across two datasets validate benefits of QNI-CCP into a CNN–QNN pipeline to improve generalization, enhance feature representation and robustness, while preserving the predictive performance. Overall, our results highlight the necessity of a dual-level defense strategy—spanning both latent-space and pixel-space perturbations for robust malware detection in the NISQ era.

References

[1] Diego Abreu, Christian Esteve Rothenberg, and Antonio Abelém. 2024. QML-IDS: Quantum Machine Learning Intrusion Detection System. In *2024 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 1–6.

[2] Alve Rahman Akash, BoHyun Ahn, Alycia Jenkins, Ameya Khot, Lauren Silva, Hugo Tavares-Vengas, and Taesic Kim. 2023. Quantum Convolutional Neural Network-based Online Malware File Detection for Smart Grid Devices. In *2023 IEEE Design Methodologies Conference (DMC)*. IEEE, 1–5.

[3] Mst Shapna Akter, Hossain Shahriar, Sheikh Iqbal Ahamed, Kishor Datta Gupta, Muhammad Rahman, Atef Mohamed, Mohammad Rahman, Akond Rahman, and Fan Wu. 2023. Case study-based approach of quantum machine learning in cybersecurity: Quantum support vector machine for malware classification and protection. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 1057–1063.

[4] Reyadh Alluhaibi. 2024. Quantum Machine Learning for Advanced Threat Detection in Cybersecurity. *International Journal of Safety & Security Engineering* 14, 3 (2024).

[5] Qian Chen, Qing Liao, Zoe L Jiang, Junbin Fang, Siuming Yiu, Guikai Xi, Rong Li, Zhengzhong Yi, Xuan Wang, Lucas CK Hui, et al. 2018. File fragment classification using grayscale image conversion and deep learning in digital forensics. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 140–147.

[6] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Dacheng Tao, and Nana Liu. 2021. Quantum noise protects quantum classifiers against adversaries. *Physical Review Research* 3, 2 (2021), 023153.

[7] Nicola Galloro, Mario Polino, Michele Carminati, Andrea Continella, and Stefano Zanero. 2022. A systematical and longitudinal study of evasive behaviors in windows malware. *Computers & security* 113 (2022), 102550.

[8] Daniel Gibert, Carles Mateu, and Jordi Planes. 2020. The rise of deep learning for detection of malware binaries: A survey. *Future Generation Computer Systems* 110 (2020), 625–639.

[9] Weiyuan Gong and Dong-Ling Deng. 2022. Universal adversarial examples and perturbations for quantum classifiers. *National Science Review* 9, 6 (2022), nwab130.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[11] Kishu Gupta, Deepika Saxena, Pooja Rani, Jitendra Kumar, Aaisha Makkar, Ashutosh Kumar Singh, and Chung-Nan Lee. 2024. An intelligent quantum cyber-security framework for healthcare data management. *IEEE Transactions on Automation Science and Engineering* (2024).

[12] Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. 2019. Supervised learning with quantum-enhanced feature spaces. *Nature* 567, 7747 (2019), 209–212. doi:10.1038/s41586-019-0980-2

[13] Moe Hdaib, Sutharshan Rajasegarar, and Lei Pan. 2024. Quantum deep learning-based anomaly detection for enhanced network security. *Quantum Machine Intelligence* 6, 1 (2024), 26.

[14] Sirui Lu, Lu-Ming Duan, and Dong-Ling Deng. 2020. Quantum adversarial machine learning. *Physical Review Research* 2, 3 (2020), 033212.

[15] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[16] Lakshmanan Nataraj, Shanmugavadivel Karthikeyan, Grégoire Jacob, and Biswanath S Manjunath. 2011. Malware images: visualization and automatic classification. In *Proceedings of the 8th International Symposium on Visualization for Cyber Security*. 1–7.

[17] Anson Pinhero, ML Anupama, P Vinod, Corrado Aaron Visaggio, N Aneesh, S Abhijith, and S AnanthaKrishnan. 2021. Malware detection employed by visualization and deep neural network. *Computers & Security* 105 (2021), 102247.

[18] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (Aug. 2018), 79. doi:10.22331/q-2018-08-06-79

[19] Soja S and Mouleeswaran. 2024. Image Based Malware Detection Using Transfer Learning and Hybrid Quantum Machine Learning Models. *Nanotechnology Perceptions* (06 2024). doi:10.62441/nano-ntp.vi.1527

[20] Dhaou Said, Miloud Bagaa, Aziz Oukaira, and Ahmed Lakhssassi. 2024. Quantum entropy and reinforcement learning for distributed denial of service attack detection in smart grid. *IEEE Access* (2024).

[21] Vita Santa Barletta, Danilo Caivano, Mirko De Vincentiis, Anibrata Pal, and Michele Scalera. 2024. Hybrid quantum architecture for smart city security. *Journal of Systems and Software* 217 (2024), 112161.

[22] Md Shahnawaz, Bishwajit Prasad Gond, and Durga Prasad Mohapatra. 2025. Dynamic Malware Classification of Windows PE Files using CNNs and Greyscale Images Derived from Runtime API Call Argument Conversion. *arXiv preprint arXiv:2505.24231* (2025).

[23] Jollanda Shara. 2023. Quantum Machine Learning and Cybersecurity. 12 (06 2023), 47–56.

[24] SonicWall. 2025. 2025 SonicWall Cyber Threat Report. https://www.sonicwall.com/resources/white-papers/2025-sonicwall-cyber-threat-report Accessed: 2026-01-06.

[25] Hatma Suryotrisongko, Yasuo Musashi, Akio Tsuneda, and Kenichi Sugitani. 2022. Adversarial robustness in hybrid quantum-classical deep learning for botnet dga detection. *Journal of Information Processing* 30 (2022), 636–644.

[26] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. (2013). arXiv:1312.6199 [cs.LG]

[27] Danish Vasan, Mamoun Alazab, Sobia Wassan, Babak Safaei, and Qin Zheng. 2020. Image-Based malware classification using ensemble of CNN architectures (IMCEC). *Computers & Security* 92 (2020), 101748. doi:10.1016/j.cose.2020.101748

[28] Zihao Wang, Kar Wai Fok, and Vrizlynn L. L. Thing. 2025. Network attack traffic detection with hybrid quantum-enhanced convolution neural network. *Quantum Machine Intelligence* 7, 1 (2025), 1–20. doi:10.1007/s42484-025-00278-0

[29] Fangtian Zhong, Zekai Chen, Minghui Xu, Guoming Zhang, Dongxiao Yu, and Xiuzhen Cheng. 2022. Malware-on-the-brain: Illuminating malware byte codes with images for malware classification. *IEEE Trans. Comput.* 72, 2 (2022), 438–451.

[30] Fangtian Zhong, Xiuzhen Cheng, Dongxiao Yu, Bei Gong, Shuaiwen Song, and Jiguo Yu. 2023. Malfox: Camouflaged adversarial malware example generation based on conv-gans against black-box detectors. *IEEE Trans. Comput.* 73, 4 (2023), 980–993.