applied sciences

IMPACT FACTOR 2.5

CITESCORE 5.5

Article

Sensitivity Analysis of Variational Quantum Classifiers for Identifying Dummy Power Traces in Side-Channel Analysis

Seungun Park and Yunsik Son

Special Issue

Advances in Intelligent Systems—2nd edition

Edited by

Prof. Dr. Zong Woo Geem

MDPI

https://doi.org/10.3390/app16073243

applied sciences

MDPI

Article

Sensitivity Analysis of Variational Quantum Classifiers for Identifying Dummy Power Traces in Side-Channel Analysis †

Seungun Park and Yunsik Son *

Department of Computer Science and Artificial Intelligence, Dongguk University, Seoul 04620, Republic of Korea; cystem@dgu.ac.kr

* Correspondence: sonbug@dongguk.edu

† This paper is an extended version of our papers submitted in Park, S.; Kim, J.; Son, Y. Quantum Neural Network Approach for Identifying Dummy Power Traces in Side-Channel Analysis. In Proceedings of the 26th International Symposium on Advanced Intelligent Systems (ISIS 2025), Cheongju, Republic of Korea, 6–9 November 2025.

Abstract

The application of quantum machine learning (QML) to security-relevant problems has attracted growing attention, yet its practical behavior in realistic workloads remains insufficiently characterized. This paper investigates the feasibility and limitations of variational quantum classifiers (VQCs) for identifying dummy power traces in side-channel analysis (SCA). A controlled benchmarking framework is developed to evaluate training stability, sensitivity to key design parameters, and resource–performance trade-offs under realistic constraints. To move beyond idealized simulation, hardware-relevant factors, including finite measurement budgets and device noise, are incorporated, and inference robustness under degraded operating conditions is assessed. The results show that VQCs can capture meaningful discriminative patterns in structured side-channel data, although robustness and performance depend strongly on encoding strategy, circuit depth, and measurement conditions. These findings provide an empirical assessment of the potential and limitations of QML for side-channel security and offer practical guidance for future research.

Keywords: quantum machine learning; quantum neural network; variational quantum classifier; side-channel analysis; pattern recognition

1. Introduction

Quantum machine learning (QML) has emerged as a promising research direction at the intersection of quantum computing and data-driven modeling [1,2]. As quantum hardware advances toward larger, more reliable devices, QML is often viewed as a promising methodology that may eventually complement or accelerate certain learning tasks by leveraging quantum-state representations and measurements. At the same time, the current landscape is dominated by noisy intermediate-scale quantum (NISQ) devices, where limited qubit counts, finite sampling budgets, and hardware noise impose nontrivial constraints on practical learning performance [3]. These constraints motivate a careful, empirical understanding of what QML can and cannot do today, and under which design and resource conditions it can become useful for real-world applications [4].

Learning-based methods are increasingly shaping modern information security [5]. Recent studies span intrusion detection [6], software weakness analysis [7], and side-channel analysis (SCA) [8,9]. Related security analysis workflows have also been explored for smart contracts [10]. Beyond general pattern recognition tasks, QML has increasingly been explored in security-relevant classification. These areas include intrusion and anomaly

Academic Editor: Zong Woo Geem

Received: 28 February 2026
Revised: 23 March 2026
Accepted: 25 March 2026
Published: 27 March 2026

detection [11], distributed denial-of-service (DDoS) traffic analysis [5,12], malicious URL and phishing detection [13,14], malware classification [15], and attack detection in cyber–physical environments [16]. In these applications, variational quantum classifiers (VQCs) that use parameterized quantum circuits (PQCs) have been explored both as direct learning models and within hybrid quantum–classical workflows, where preprocessing, feature reduction, and encoding design often play a central role in practical performance [11,17,18]. This growing body of work suggests that QML is becoming relevant to information security not only as a long-term conceptual possibility, but also as an empirical framework for studying how near-term quantum models behave on structured security data [19].

In machine learning (ML), major advances have often followed not merely incremental improvements to fixed classifier families, but also rethinking architecture, supervision, and task formulation. Sequence modeling shifted from recurrent and convolutional designs to attention-based architectures [20]; self-supervised learning enabled strong representations to be learned from large unlabeled data before task-specific fine-tuning [21]; and object detection was reformulated as an end-to-end set prediction problem, reducing reliance on hand-designed pipeline components [22]. More recently, diffusion-based detection models have further revisited how challenging detection problems are formulated, including settings with substantial background interference and uncertain object characteristics in industrial inspection [23,24]. Viewed in this way, the practical question for QML is not simply whether a quantum model can be applied to a new dataset, but which formulations, encodings, and hybrid learning strategies remain trainable, stable, and operationally meaningful under NISQ constraints [4].

SCA remains a major threat to cryptographic implementations [25–27]. SCA exploits unintended physical leakage, such as power consumption, electromagnetic emission, or execution time, to infer sensitive information or to distinguish internal execution states. In response, many countermeasures aim to reduce exploitable leakage by obfuscating observable behavior, for example, by inserting dummy operations or otherwise randomizing execution patterns [28]. However, the effectiveness of such countermeasures relies crucially on whether an adversary can still separate real traces from dummy traces in the observed traces. If dummy and real operations are distinguishable at the signal level, an attacker may recover a more informative subset of trace segments, possibly undermining the intended protection. This makes dummy trace detection a practically meaningful and structurally challenging classification problem, providing a useful testbed for evaluating QML behavior on security-relevant data.

Side-channel traces pose challenges that differ from many standard benchmark datasets: they are noisy, often non-stationary, and exhibit a time-ordered structure in which subtle local differences can carry security-relevant information. Moreover, applying VQCs, one of the most widely used QML paradigms on NISQ hardware, to such data raises practical questions that go beyond the choice of a model family [2,4]. These questions are closely tied to data encoding choices and trainability issues such as barren plateaus [29–31]. In particular, performance can be highly sensitive to classical-to-quantum data encoding, normalization ranges, quantum circuit depth, optimizer dynamics, and the measurement budget required to obtain stable gradients and predictions. Without a systematic evaluation of these factors, it is difficult to assess whether QML-based approaches are merely conceptually appealing or genuinely actionable in SCA contexts.

This study presents a practical benchmark investigation of applying a VQC to distinguish dummy traces from real traces in SCA using power measurements. The VQC is adopted as a canonical QML baseline to examine feasibility, training stability, sensitivity to design parameters, and resource–performance trade-offs under NISQ constraints [2,4]. Experimental results indicate that VQC models can extract discriminative patterns from

side-channel inputs and achieve meaningful classification performance. Because robustness under degraded operating conditions is a recurring concern across ML domains [32], this paper explicitly examines when VQC training remains stable, when degradation emerges under practical execution conditions, and which architectural or optimization choices most strongly influence model behavior.

To that end, a structured exploration of the VQC design space and deployment-relevant constraints on real quantum devices is conducted. First, the effect of normalization range on learnability is analyzed, motivated by the fact that typical encodings, such as rotation-angle encodings, directly map normalized values to quantum gate parameters. Second, the impact of circuit depth, controlled via the number of repeated layers, is examined in terms of expressivity versus trainability, including the potential for optimization difficulties as circuits deepen. Third, multiple optimizers commonly used in variational algorithms are compared to highlight differences in convergence behavior and robustness. Finally, finite-shot budgets and a hardware-informed noise model are investigated as unavoidable deployment factors. In the experiments, these effects are quantified primarily through repeated inference runs with fixed trained parameters to isolate execution-induced variability. Limited shots introduce estimation variance in measured expectations, and device noise can distort both the training signal and the resulting decision boundary. Performance trends across these dimensions provide an empirically grounded perspective on QML behavior in a security-relevant, noisy, and structured dataset setting.

Overall, this work serves as a feasibility assessment of VQC-based learning on side-channel inputs, a controlled benchmark framework for examining experimentally relevant design factors, and practical guidance for applying QML to SCA settings, including countermeasure evaluation. A preliminary version of this work was presented at ISIS 2025 as an initial feasibility study of a quantum-learning-based approach to dummy power-trace identification [33]. The present article substantially extends that conference version through more systematic experiments, expanded sensitivity analysis, and a broader evaluation of practical configuration execution factors.

The contributions of this work are as follows:

- VQC formulation for side-channel dummy detection: A VQC-based binary classification model is presented for distinguishing real operations from dummy operations in power-based side-channel traces, establishing a practical starting point for applying QML to hiding distinguishability analysis.
- Security-relevant benchmark formulation: Dummy trace detection is formulated as a controlled binary classification benchmark for studying QML behavior on side-channel data, motivated by the role of distinguishability in evaluating hiding-based countermeasures.
- Controlled sensitivity analysis of the presented VQC: Using a standard hybrid quantum–classical VQC pipeline, systematic evaluation is conducted on how scaling range, circuit depth, optimizer choice, training-data fraction, finite-shot budgets, and hardware noise affect trainability, predictive performance, and inference robustness.
- Scope-bounded empirical guidance: Stable and unstable operating regimes for the studied configuration are identified, and practical guidance for applying VQCs to side-channel settings is provided.

2. Related Work

2.1. Learning-Based Side-Channel Analysis

SCA exploits unintended physical leakage, such as power consumption, electromagnetic emissions, or execution time, to infer internal states of a computation or recover secrets from cryptographic implementations. Classical SCA includes simple power analysis (SPA)

and differential analysis (DPA), which correlate measured power with key-dependent intermediate values to extract secret keys [25]. Established foundations describe SCA threat models, leakage sources, and evaluation methodologies [26,27]. Beyond traditional key-recovery attacks, modern SCA increasingly encompasses broader security objectives, including program behavior inference and reverse engineering from side-channel signals.

Before the recent dominance of deep learning (DL), learning-based SCA had already been studied through classical profiling and ML perspectives. Early work showed that ML can be effectively applied to side-channel trace classification and that model choice and parameterization can significantly affect attack performance [34]. Subsequent studies further clarified the relationship between classical profiling methods and supervised learning by comparing template attacks with Bayesian classifiers and other ML approaches [35,36]. More recent work also indicates that DL should not be viewed only as a replacement for classical profiling, but can also be combined with it, as exemplified by DL-assisted template attacks [37]. Taken together, these studies show that learning-based SCA spans a continuum from classical statistical profiling and feature-based ML to modern deep representation learning.

Building on these classical profiling and ML foundations, DL-based approaches have become a central pillar in modern SCA. In profiling settings, DL models learn discriminative representations directly from traces, often reducing the need for handcrafted feature extraction and enabling attacks under more realistic noise and variability conditions. A systematic consolidation of this trend is provided by the systematization of knowledge (SoK) literature on DL-based SCA (DL-SCA), which categorizes how learning is used across the attack pipeline and shows both the capabilities and limitations of DL-based adversaries [8,38]. Early end-to-end profiling strategies based on convolutional neural networks (CNNs) demonstrated that DL can be effective without explicit alignment or manual point-of-interest selection, and that augmentation strategies can improve robustness against jitter-style misalignment [39]. Subsequent work further investigated architectural and methodological choices for DL-SCA, proposing systematic methodologies for CNN design and evaluation under common SCA constraints [40]. These results collectively suggest that learning-based models are well-suited to the noisy, high-dimensional, and temporally structured characteristics of side-channel traces.

A closely related line of research extends DL-SCA from key recovery to side-channel-based disassembly (SCBD), where the goal is to infer program-level information, such as executed instructions, control flow, and higher-level behaviors, from leakage. This direction is security-relevant for malware analysis, firmware and binary reverse engineering, and the evaluation of software confidentiality in embedded and IoT deployments. Early studies showed the feasibility of inferring instruction-level behavior from power leakage for Internet of Things (IoT) devices [41,42]. Early SCBD approaches based on electromagnetic or power leakage were also demonstrated in prior hardware-security studies [43,44]. Later work expanded the scope by addressing instruction sequence identification on pipelined platforms [45]. It also extended the analysis to AVR, bit-level, and ARM Cortex-M0 disassembly with additional robustness considerations [46–48]. SCBD has also been explored at design time, where simulated traces are used to assess vulnerability prior to fabrication, enabling security feedback earlier in the hardware design process [49]. Practical feasibility has further been studied on complex platforms such as systems-on-chip, emphasizing challenges in real measurement settings and the granularity of recoverable information [50].

Recent SCBD-oriented studies emphasize the importance of realistic datasets and systematic evaluation frameworks. In a fully simulated IoT environment, feature engineering techniques based on rolling windows have been proposed, including moving log-transformed temporal interaction features, which improve both countermeasure de-

tection and instruction inference and demonstrate strong performance in these tasks [9]. Complementary to direct disassembly, contextual leakage modeling attempts to predict power traces from assembly-level information. A context-aware DL framework has been introduced to generate instruction-level power traces from assembly code, supporting automated and scalable leakage evaluation and enabling downstream SCBD analyses [51,52]. These studies indicate that learning-based methods can capture fine-grained relationships between execution semantics and observed leakage, motivating careful evaluation of how modeling choices affect robustness and generalization.

2.2. Countermeasures and Their Distinguishability

Defending against SCA usually involves reducing exploitable leakage or disrupting an attacker's ability to align and correlate traces. Classical countermeasure families include masking, which randomizes sensitive intermediate values to decorrelate leakage from secrets, and hiding, which aims to obscure leakage by introducing temporal jitter, additional noise, or execution randomization [26]. Related work has also explored countermeasures across leakage modalities, where both measurement-side and implementation-side defenses are relevant [27]. Even when the leakage source cannot be eliminated, many practical defenses attempt to increase the effective difficulty of analysis by perturbing the observable signal structure.

Among hiding techniques, dummy operation insertion and shuffling are commonly used to randomize execution and reduce alignment. In particular, dummy insertion aims to increase the attacker's complexity by interleaving real operations with non-secret-dependent operations. However, the security benefit is critically dependent on distinguishability. If an adversary can reliably separate real and dummy segments, they can filter traces, re-establish alignment, or recover informative subsets, undermining the intended protection. Vulnerabilities have been identified in which dummy and real operations remain distinguishable across multiple implementation methods and compiler optimization levels. Case studies on AES implementations employing insertion and shuffling demonstrate that these hiding techniques may still leak distinguishable side-channel patterns, and corresponding countermeasure strategies have been explored [28]. Learning-based approaches have also been developed to detect dummy operations directly from side-channel signals. Dummy detection can be formulated as a supervised learning problem, enabling automated classification to identify dummy operations. Experimental results show that dummy insertion can be effectively attacked using DL-based classifiers [53]. These results motivate treating distinguishability not as an incidental artifact, but as a primary security metric for evaluating hiding schemes.

DL strengthens this perspective by enabling models to capture subtle and localized leakage patterns that persist even under intentional obfuscation. Prior work has demonstrated that CNN-based profiling attacks with data augmentation can weaken misalignment-based countermeasures by learning invariance to jitter and temporal perturbations [39]. Systematic evaluations have shown that architectural and methodological choices significantly affect the robustness of DL-SCA under common defense mechanisms. These results highlight the importance of model design and training methodology when evaluating attack effectiveness [40]. Recent studies focusing on dummy-based hiding have shown that models trained on datasets augmented with generated dummy traces can effectively distinguish genuine and dummy operations [54]. Learning-based pipelines have also demonstrated the ability to recover execution semantics and detect hiding schemes from side-channel signals in simulated environments. These results suggest that hiding countermeasures can be evaluated and potentially bypassed using data-driven analysis [9]. Overall, prior work indicates that distinguishing dummy and real operations is both prac-

tically relevant and closely related to the effectiveness of hiding countermeasures under realistic conditions.

2.3. Quantum Machine Learning on NISQ Hardware

QML investigates how quantum computation methods can be used to perform or accelerate learning tasks. Quantum computation manipulates qubits, which can exist in superpositions of basis states and can be correlated through entanglement. Computation is implemented as a sequence of quantum gates, and information is extracted by measurement. Importantly, measurement results are probabilistic. Therefore, estimating quantities such as expectation values typically requires repeated circuit executions, often referred to as shots. Current devices are commonly described as NISQ hardware, where limited qubit counts and non-negligible noise constrain circuit depth and algorithmic reliability [3].

In the long term, fault-tolerant quantum computing (FTQC) is expected to mitigate noise through quantum error correction [55–57]. Foundational proposals include quantum error-correcting codes and fault tolerance frameworks, as well as topological approaches that aim to encode information in more noise-resilient degrees of freedom [58–60]. However, near-term QML research largely operates without full error correction and must, therefore, explicitly account for finite sampling and hardware noise. This makes it essential to characterize not only algorithmic expressivity but also resource–performance trade-offs and training stability in realistic conditions.

Within QML, supervised learning can be implemented via various paradigms. Standard references provide a broad conceptual and algorithmic introduction to supervised learning on quantum computers and discuss key design choices such as data encoding and measurement strategies [2]. A widely cited review further contextualizes QML as a field, summarizing algorithmic building blocks and the challenges that arise in connecting theoretical speedups and practical implementations [1]. On NISQ devices, one of the most prevalent practical approaches is variational quantum algorithms (VQAs), which use PQCs optimized by a classical optimizer in a hybrid loop [4]. VQCs can be viewed as VQA instances designed for classification, where classical inputs are embedded into quantum circuits and measurement results define decision functions. Related work on quantum circuit learning provides early formulations of hybrid training for near-future devices and clarifies how parameterized circuits can represent learnable models [61]. Quantum feature-space methods, including quantum-enhanced feature maps, further motivate the use of quantum circuits as nonlinear embeddings for supervised learning [29].

Variational models face challenges during training. One significant issue is the emergence of barren plateaus, where gradients vanish exponentially with increasing system size for certain circuit families and initialization choices, leading to optimization stagnation [30,31]. In addition, finite-shot estimation introduces stochasticity into objective evaluations and gradients, and device noise can distort both training signals and predictions. Consequently, practical performance on NISQ hardware depends strongly on choices, such as data normalization and encoding ranges, circuit depth, optimizer type, and shot budgets, factors that often matter less in classical deterministic inference.

2.4. Parameterized Quantum Circuit Learning for Security Tasks

Recent work has begun to explore the potential of QML in security-oriented classification tasks. Studies have investigated quantum learning models for intrusion and anomaly detection [11], DDoS traffic analysis [5,12], malicious URL and phishing detection [13,14], malware classification [15], and attack detection in cyber–physical environments [16]. In many of these studies, PQCs are employed as learnable models within hybrid quantum–classical pipelines, where classical preprocessing, feature reduction, and encoding design

remain central components of the overall workflow [11,17,18]. These efforts primarily aim to evaluate whether NISQ quantum models can complement or integrate with existing ML pipelines used in practical security analysis.

Within this landscape, PQC-based classifiers, commonly instantiated as VQCs, have emerged as one of the most practical supervised learning approaches for NISQ devices [2,4]. These models encode classical inputs into PQCs and optimize circuit parameters through a hybrid quantum–classical training loop. More recent works suggest that this basic VQC paradigm is being extended in two complementary ways for classical data classification. One direction embeds VQC-based quantum modules within broader hybrid architectures, such as pipelines with classical processing or residual models that incorporate VQC blocks [62,63]. The other direction seeks to reduce classical scaffolding and move toward more self-contained trainable quantum classifiers, including models based on quantum convolutional neural networks (QCNNs) and their recent variational or shadow-assisted extensions [64–66]. Empirical studies have examined their applicability to a variety of security datasets and have begun to assess whether quantum feature embeddings or hybrid architectures can improve classification performance and robustness in security-relevant settings [5,19]. However, most existing evaluations focus on conventional cybersecurity datasets, such as network traffic, malware features, or URL attributes, whose data structures resemble those of standard ML benchmarks.

In contrast, the application of PQC-based learning to physical side-channel workloads remains relatively limited compared to other cybersecurity domains. Most existing studies evaluate quantum models on datasets derived from network traffic, malware attributes, or other structured security features, where the data format resembles conventional ML benchmarks. Side-channel traces, however, represent time-ordered physical measurements that capture subtle variations in device behavior during execution. Their statistical properties and noise characteristics differ substantially from typical tabular or feature-engineered security datasets. These differences suggest that the behavior of PQC-based models on side-channel data may not directly follow observations reported for other security tasks.

2.5. Summary of Research Gap

Prior work shows that learning-based methods can extract meaningful information from physical leakage in both key-recovery and program-inference settings [8,41,67]. SCBD has also matured into a practical reverse-engineering and evaluation toolchain [49,50]. At the same time, countermeasure effectiveness increasingly depends on distinguishability: hiding countermeasures such as dummy insertion can fail if an attacker learns to separate dummy instructions from observed traces [28,53,54]. These trends establish a clear security motivation for studying dummy detection as a benchmark problem, especially under realistic noise and alignment conditions.

In parallel, QML has developed a strong methodological foundation, featuring variational quantum models emerging as a practical tool family for near-term supervised learning [2–4]. However, the gap lies in the intersection: there is limited empirical characterization of how VQCs behave on side-channel datasets, where performance may be dominated by encoding ranges, circuit depth, trainability, optimizer stability, and shot budgets. Moreover, hardware-relevant effects such as finite shots and noise are not optional details but core determinants of learnability in NISQ settings.

Therefore, a key open problem is the development of a controlled evaluation framework that bridges security-relevant side-channel workloads and the distinguishability between dummy and real side-channel traces by QML models under the practical constraints and sensitivities of NISQ-era QML systems. Addressing this gap can clarify what

QML can reliably learn from side-channel inputs, which design regimes are stable, and what resource costs are required for meaningful performance.

3. Methodology

3.1. Scope and Benchmark Objective

This work is designed as a controlled benchmark study to characterize how the behavior of a VQC changes under practical design choices and resource constraints. This paper treats a representative binary side-channel classification task as a convenient workload, enabling systematic analysis of how model configuration and execution conditions influence training stability and predictive performance. Figure 1 shows the benchmark workflow and factors varied in the training and inference-time robustness analyses.

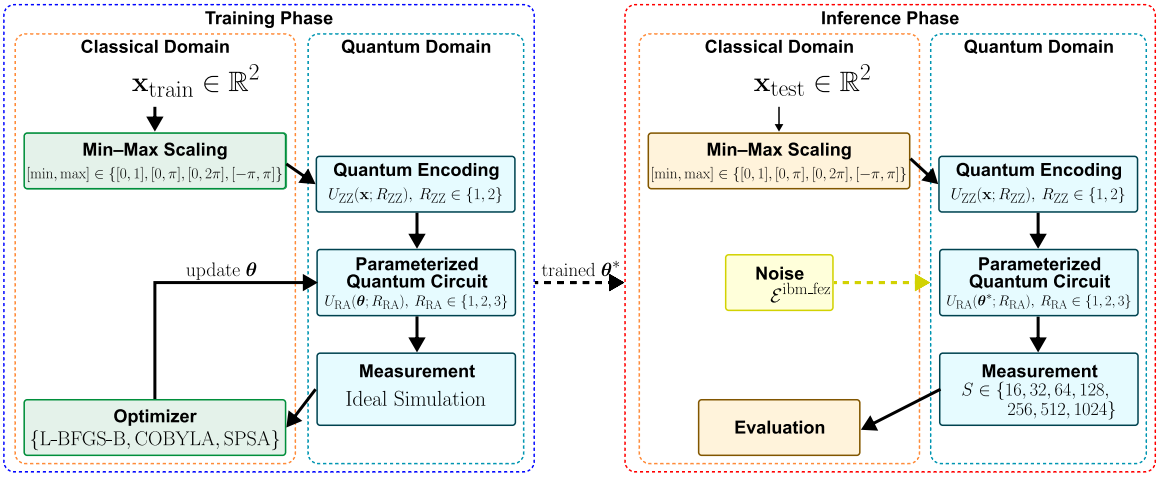

Figure 1. Workflow of the hybrid quantum–classical VQC benchmark. The trained parameters $\boldsymbol{\theta}^*$ are used for inference.

Experimental Factors

Table 1 summarizes the experimental factors. To isolate the impact of individual choices, the benchmark primarily follows a one-factor-at-a-time evaluation strategy. In each experiment, a single factor is varied while the remaining configuration is held at a reference setting, except where an exploratory sweep is explicitly defined.

Table 1. Experimental factors and options considered in the benchmark.

Factor	Symbol/Option	Role in This Study
Scaling range	$[\min, \max]$	Min–max scaling target range: $[0,1]$, $[0,\pi]$, $[0,2\pi]$, $[-\pi,\pi]$.
Feature map depth	R_{ZZ}	Repetitions in ZZFeatureMap; controls encoding depth and correlation structure.
Ansatz depth	R_{RA}	Repetitions in RealAmplitudes; controls expressivity vs. trainability.
Optimizer	L-BFGS-B/COBYLA/SPSA	Compares optimizer dynamics under the same circuit family and scaling.
Measurement budget	S shots	Evaluates stability/performance under finite-shot estimation.
Noise condition	$\mathcal{E}$	Evaluates robustness under noisy execution.

3.2. Preprocessing

Formally, a labeled dataset is defined as:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}, \quad \mathbf{x}_i \in \mathbb{R}^d, \quad y_i \in \{0,1\} \tag{1}$$

where N is the total number of samples and d is the feature dimension. y_i denotes a binary label indicating whether the sample is dummy or a real trace. Each $\mathbf{x}_i$ is a low-dimensional feature vector derived from power measurements. In this paper, the feature dimension is fixed to $d = 2$, corresponding to the trace position and power value, and the number of qubits is set to match d, enabling a minimal yet interpretable VQC configuration analysis under NISQ-era constraints.

Angle-parameterized encodings map normalized classical values to rotation angles or phase parameters in quantum gates. Consequently, the normalization range can significantly affect circuit behavior, gradient magnitude, and optimization dynamics.

Let x denote a raw feature value for one dimension, and let $x_{\min}$ and $x_{\max}$ be the minimum and maximum values estimated from the training set for that dimension. To map x into a target range $[a, b]$, min–max scaling is applied:

$$\tilde{x} \;=\; a + \frac{x - x_{\min}}{x_{\max} - x_{\min}}(b - a) \tag{2}$$

Four scaling ranges are evaluated:

$$[\min, \max] \in \Big\{ [0,1],\ [0,\pi],\ [0,2\pi],\ [-\pi,\pi] \Big\} \tag{3}$$

The $[0, 2\pi]$ range is commonly used for rotation-angle encodings, while $[-\pi, \pi]$ enables a symmetric angular range around zero. The $[0, 1]$ and $[0, \pi]$ ranges are included to assess whether restricting angles to a small interval improves trainability at the possible cost of reduced expressivity.

3.3. Variational Quantum Classifier

3.3.1. Quantum Computing Preliminaries

A quantum circuit on n qubits acts on a complex quantum state initialized to $|0\rangle^{\otimes n}$. A parameterized circuit prepares a state:

$$|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle = U(\mathbf{x}, \boldsymbol{\theta})\, |0\rangle^{\otimes n} \tag{4}$$

where $U(\mathbf{x}, \boldsymbol{\theta})$ is a composition of gates determined by the input $\mathbf{x}$ and trainable parameters $\boldsymbol{\theta}$. The final state is measured in the computational basis, producing a bitstring $z \in \{0,1\}^n$. The probability of observing z is

$$p_{\boldsymbol{\theta}}(z \mid \mathbf{x}) = |\langle z \rangle\, \psi(\mathbf{x}, \boldsymbol{\theta})|^2 \tag{5}$$

In practice, especially on NISQ hardware, probabilities are not directly accessible and are instead estimated from repeated circuit executions, called shots.

3.3.2. Circuit Structure

A canonical VQC structure is adopted, consisting of a data-encoding feature map and a trainable variational ansatz:

$$U(\mathbf{x}, \boldsymbol{\theta}) = U_{\mathrm{RA}}(\boldsymbol{\theta}; R_{\mathrm{RA}})\, U_{\mathrm{ZZ}}(\mathbf{x}; R_{\mathrm{ZZ}}) \tag{6}$$

where U_{ZZ} denotes the ZZFeatureMap with repetition number R_{ZZ}, and U_{RA} denotes the RealAmplitudes ansatz with repetition number R_{RA}. These repetition numbers control the encoding depth and variational expressivity, respectively. Figure 2 illustrates the structure of the VQC used in this study.

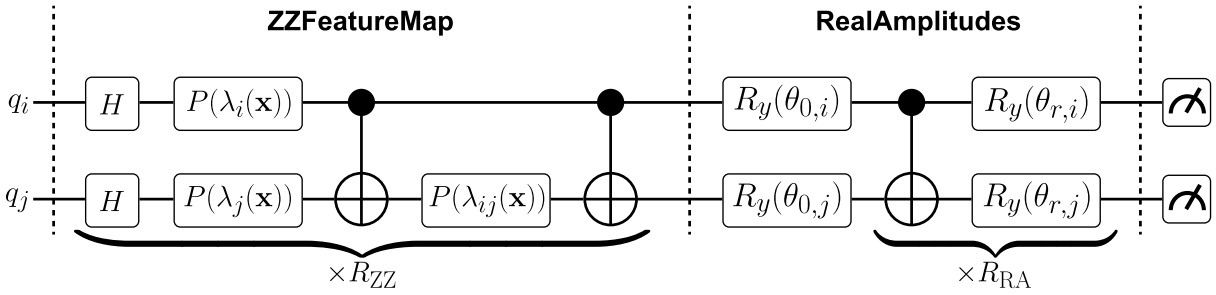

Figure 2. The structure of the VQC used in this study. Solid lines indicate qubit wires, filled circles represent control qubits, and the $\oplus$ symbols denote target operations in controlled gates.

ZZFeatureMap

The second-order Pauli-Z evolution feature map (ZZFeatureMap) is used with full entanglement and repetition depth R_{ZZ}. For an input $\mathbf{x} \in \mathbb{R}^d$, a single repetition is implemented as a Hadamard layer followed by phase-encoding and pairwise ZZ interactions realized through the standard CX–P–CX pattern:

$$U_{ZZ}(\mathbf{x}; R_{ZZ}) = \prod_{r=1}^{R_{ZZ}} \left(\left(\prod_{i<j} \mathrm{CX}_{i,j}\, P_j(\lambda_{ij}(\mathbf{x}))\mathrm{CX}_{i,j} \right) \left(\prod_{k=1}^{n} P_k(\lambda_k(\mathbf{x})) \right) H^{\otimes n} \right) \tag{7}$$

where P denotes a single-qubit phase gate and CX denotes the controlled NOT gate. The phase angles $\lambda_k(\mathbf{x})$ and $\lambda_{ij}(\mathbf{x})$ are defined by the Qiskit feature-map template.

RealAmplitudes

The RealAmplitudes ansatz is used with full entanglement and repetition depth R_{RA}. The circuit alternates parameterized single-qubit R_y rotation layers with entangling layers composed of CX gates:

$$U_{RA}(\boldsymbol{\theta}; R_{RA}) = \prod_{r=1}^{R_{RA}} (U_{ent}\, U_{rot}(\boldsymbol{\theta}_r)) U_{rot}(\boldsymbol{\theta}_0) \tag{8}$$

where $U_{rot}(\boldsymbol{\theta}_r) = \prod_{k=1}^{n} R_y(\theta_{r,k})$ and $U_{ent} = \prod_{i<j} \mathrm{CX}_{i,j}$ under full entanglement. The parameter vector $\boldsymbol{\theta}$ contains all trainable angles across the repeated layers, and R_{RA} controls the variational depth.

3.3.3. Output Rule for Binary Classification

As described in the preliminaries, circuit execution followed by measurement induces a probability distribution over computational basis states. In the VQC implementation used in this work, the sampled measurement statistics are internally mapped to class probabilities through the built-in interpretation mechanism.

Given an input $\mathbf{x}$, the classifier produces estimated class probabilities $p_{\boldsymbol{\theta}}(y = c \mid \mathbf{x})$, from which the predicted label is defined as:

$$\hat{y}(\mathbf{x}) = \arg\max_{c \in \{0,1\}} p_{\boldsymbol{\theta}}(y = c \mid \mathbf{x}) \tag{9}$$

where $\mathcal{I} : \{0,1\}^n \to \{0,1\}$ denotes the interpretation map. Then, $p_{\boldsymbol{\theta}}(y = c \mid \mathbf{x}) = \sum_{z:\mathcal{I}(z)=c} p_{\boldsymbol{\theta}}(z \mid \mathbf{x})$.

3.4. Training Objective and Optimizers

Training adjusts $\boldsymbol{\theta}$ to minimize an empirical risk on the training set. Using class probabilities, the standard binary cross-entropy loss $\mathcal{L}$ is adopted. Optimization follows

the hybrid quantum–classical loop: a classical optimizer proposes $\boldsymbol{\theta}$, and the quantum device or simulator evaluates $\mathcal{L}(\boldsymbol{\theta})$ via repeated circuit executions. This study compares three representative optimizers that capture common trade-offs in QML, especially under expensive function evaluations and measurement noise.

L-BFGS-B

A limited-memory quasi-Newton method that uses gradient information and enforces simple box constraints. It typically converges in relatively few iterations on smooth objectives and is attractive when gradients are available. However, its performance can degrade when the objective is noisy because quasi-Newton curvature updates rely on consistent gradient and step information.

COBYLA

A gradient-free constrained optimizer that builds local linear approximations of the objective and constraints. It is commonly used in VQAs when analytic gradients are unavailable or unreliable. COBYLA can handle constraints naturally, but it may require many function evaluations and can scale unfavorably with the number of parameters.

SPSA

A stochastic approximation method that estimates a gradient using random simultaneous perturbations. Crucially, it requires only two objective evaluations per iteration regardless of parameter dimension, making it well-suited to QML settings where each evaluation entails many shots and is affected by noise. SPSA is generally robust to measurement noise, although it may exhibit larger iteration-to-iteration variance and depends on step-size hyperparameters.

3.5. Finite-Shot and Noisy Execution Conditions

Variational quantum circuits are evaluated through repeated measurements, and the resulting class probabilities are estimated from a finite number of shots. Let $\{z^{(s)}\}_{s=1}^{S}$ denote the measured bitstrings obtained from S independent circuit executions for a fixed input $\mathbf{x}$, where $z^{(s)}$ represents the outcome observed at shot s. The empirical distribution over outcomes is estimated as:

$$\widehat{p}_{\boldsymbol{\theta}}(z \mid \mathbf{x}) = \frac{1}{S} \sum_{s=1}^{S} \mathbb{I}\left[z^{(s)} = z\right] \tag{10}$$

where $z \in \{0,1\}^n$ is a computational-basis bitstring and $\mathbb{I}[\cdot]$ denotes the indicator function.

This sampling-based estimate replaces the ideal probability distribution available in statevector simulations. As the shot budget S decreases, the variance of the empirical estimator increases, introducing stochastic fluctuations into both the predicted class probabilities and the optimization trajectory. Consequently, the shot count functions as a resource parameter that directly affects inference robustness.

In addition to sampling uncertainty, hardware noise further perturbs circuit behavior under realistic NISQ conditions. Let the noiseless output state be expressed as:

$$\rho(\mathbf{x}, \boldsymbol{\theta}) = U(\mathbf{x}, \boldsymbol{\theta})\, \rho_0\, U(\mathbf{x}, \boldsymbol{\theta})^{\dagger} \tag{11}$$

To model noisy execution, we switch from the statevector notation to the density-operator formalism. For the noiseless case, $\rho(\mathbf{x}, \boldsymbol{\theta}) = |\psi(\mathbf{x}, \boldsymbol{\theta})\rangle\langle\psi(\mathbf{x}, \boldsymbol{\theta})|$, with $\rho_0 = (|0\rangle\langle 0|)^{\otimes n}$ and $(\cdot)^{\dagger}$ denoting the Hermitian adjoint.

Under noisy execution, the effective state becomes

$$\rho_{\text{noisy}}(\mathbf{x}, \boldsymbol{\theta}) = \mathcal{E}(\rho(\mathbf{x}, \boldsymbol{\theta})) \tag{12}$$

where $\mathcal{E}$ represents a quantum noise channel modeling gate errors and readout imperfections. Measurement is then performed on ρ_{noisy}, yielding a distorted outcome distribution.

Finite-shot estimation and hardware noise jointly influence the effective decision boundary learned by the classifier. Sampling noise introduces statistical variability, whereas hardware noise shifts the underlying measurement distribution. The experimental design, therefore, evaluates VQC configurations across multiple shot budgets and noise conditions to characterize sensitivity, stability, and robustness under deployment-relevant constraints. In this study, shot and noise effects are evaluated via inference-time sweeps with fixed trained parameters; shot/noise-aware training is left for future work.

3.6. Evaluation Metrics

This study evaluates each configuration from three complementary perspectives: classification effectiveness, optimization behavior, and computational cost. Because the task is to distinguish real and dummy traces under varying circuit and execution conditions, performance should be assessed not only by overall correctness, but also by how errors are distributed and how efficiently the model can be trained.

Standard binary classification metrics are reported on the held-out test set. Let TP, TN, FP, and FN denote the entries of the confusion matrix. The metrics are defined as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{13}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{14}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{15}$$

$$\text{F1} - \text{score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

Accuracy provides an overall summary of prediction correctness across the test set. Precision measures how reliably the model identifies traces predicted as positive samples, whereas recall reflects how completely the model recovers the actual positive samples. In the present benchmark, these metrics are useful because some VQC configurations may produce different error tendencies even when their overall accuracy is similar. The F1-score is, therefore, treated as a primary metric, since it summarizes the precision–recall trade-off in a single value and is more informative when one type of error becomes dominant under a particular circuit depth, optimizer, scaling range, shot budget, or noise condition.

In addition to predictive performance, this study evaluates optimization efficiency using the number of iterations required for convergence and the measured wall-clock training time. The iteration count indicates how quickly an optimizer approaches a stable solution under the same circuit configuration, while the execution time reflects the practical computational cost of obtaining that solution. These two quantities are reported together because a smaller number of iterations does not necessarily imply shorter runtime: each optimizer may incur a different per-iteration cost depending on its update rule and objective evaluations. Accordingly, iterations capture convergence behavior, whereas runtime captures end-to-end training efficiency.

4. Experiments

4.1. Experimental Setup

Experiments were conducted on a publicly released side-channel dataset that contains labeled instances corresponding to real operations and inserted dummy operations. The source dataset is organized into multiple execution contexts, including RSA, DES, and Hash, and is intended to support the evaluation of hiding techniques under learning-based distinguishability analyses [54]. The dataset is generated in a controlled environment and distributed publicly for reproducible evaluation. Each entry in the dataset includes a sequence index, a power measurement value, an instruction opcode, a context identifier, and a binary label indicating whether the instruction corresponds to an original operation or an inserted dummy operation.

For the present benchmark, the classification task is formulated as a binary discrimination problem between real and dummy operations. From each entry, a two-dimensional feature vector is constructed using the normalized sequence position and power value. The dataset is divided into training and test subsets for each execution context while maintaining balanced class distributions between real and dummy samples. Table 2 summarizes the dataset composition and the experimental split used in this study.

Table 2. Dataset composition and experimental split.

Context	Train Data		Test Data		Total
	Real	Dummy	Real	Dummy	
RSA	5480	5477	1370	1370	13,697
Hash	7896	7895	1974	1974	19,739
DES	63,930	63,929	79,912	79,912	287,683

Due to the large size of the DES context and the associated runtime cost of repeated VQC training, only a fraction of the DES training set is used for the systematic sensitivity analyses. Specifically, after the initial train/test split, the DES training subset is further downsampled to 20% of its original size. This reduction allows efficient experimentation while preserving sufficient statistical diversity for evaluating model behavior. Consequently, the majority of the systematic analyses in the following sections focus on the RSA context, while the other contexts are primarily used for complementary validation.

The implementation is based on Qiskit and Qiskit Machine Learning. Unless stated otherwise, results in this section correspond to analytic/statevector execution. Additional experiments vary shot budgets and noisy execution conditions; the corresponding results are reported separately. Table 3 summarizes the software and hardware environment used for the experiments.

Table 3. Software and hardware environment.

Component	Specification
Qiskit	IBM, Armonk, NY, USA (version 2.2.3)
Qiskit Machine Learning	IBM, Armonk, NY, USA (version 0.9.0)
Python	Python Software Foundation, Wilmington, DE, USA (version 3.12.12)
CPU	Intel Core i9-13900KF, Intel Corporation, Santa Clara, CA, USA
Memory	128 GB DDR4 RAM, Samsung Electronics Co., Ltd., Seoul, Republic of Korea
OS	Ubuntu 24.04.2 LTS in WSL2, Canonical Ltd., London, UK

4.2. Circuit Depth

A grid sweep over circuit depth is conducted by varying the feature map repetitions ($R_{ZZ} \in \{1,2\}$) and ansatz repetitions ($R_{RA} \in \{1,2,3\}$) using L-BFGS-B as the optimizer. Table 4 reports test metrics and execution diagnostics for RSA under analytic/statevector execution and $[0, 2\pi]$ scaling.

Table 4. Classification performance versus circuit depth on the RSA dataset.

R_{ZZ}	R_{RA}	Accuracy	Precision	Recall	F1-Score	Iterations	Time (s)
1	1	0.570	0.566	0.602	0.583	9	1514
1	**2**	**0.893**	**0.886**	**0.901**	**0.894**	**12**	**2999**
1	3	0.889	0.888	0.890	0.889	15	5284
2	1	0.657	0.793	0.426	0.554	14	2521
2	2	0.741	0.879	0.560	0.684	18	5018
2	3	0.742	0.875	0.564	0.686	16	6080

Bold values indicate the selected configuration that achieved the best performance in this experiment and was subsequently fixed in other experimental settings.

The highest performance in this sweep is obtained at $(R_{ZZ}, R_{RA}) = (1,2)$ with an accuracy of 0.893 and an F1-score of 0.894. Increasing the ansatz depth to $R_{RA} = 3$ at the same feature-map depth yields a slightly lower F1-score (0.889). Configurations with $R_{RA} = 1$ show substantially lower F1-scores (0.583 for $(1,1)$ and 0.554 for $(2,1)$), indicating that the shallow ansatz is insufficient for stable discrimination in this setting. Increasing the feature-map depth to $R_{ZZ} = 2$ reduces F1 in all cases in this sweep; the best configuration remains below 0.70 F1.

Figure 3 summarizes mean F1-score trends across RSA, Hash, and DES. RSA and Hash exhibit a consistent qualitative pattern: performance improves markedly when moving from $R_{RA} = 1$ to $R_{RA} \in \{2,3\}$ under $R_{ZZ} = 1$, while increasing the feature-map repetitions to $R_{ZZ} = 2$ degrades F1 across the grid. For Hash, the best observed setting is $(R_{ZZ}, R_{RA}) = (1,3)$ with an F1-score of 0.808, closely followed by $(1,2)$ at 0.801. For DES, results are reported under a reduced training budget (train fraction = 0.2) due to runtime constraints; the best observed setting under this budget is $(1,2)$ with an F1-score of 0.696.

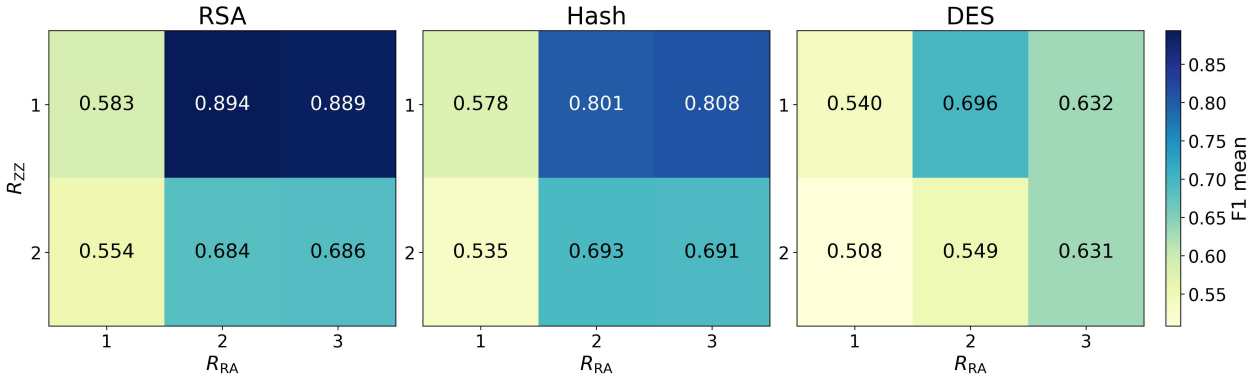

Figure 3. Mean F1-score over the (R_{ZZ}, R_{RA}) depth grid across RSA, Hash, and DES.

Runtime generally increases with deeper circuits, particularly when increasing the ansatz repetitions. For example, $(1,2)$ requires 2999 s, while $(1,3)$ increases to 5284 s and $(2,3)$ to 6080 s in the recorded environment. Although runtime is not strictly monotonic over all configurations, deeper settings such as $R_{RA} = 3$ and $R_{ZZ} = 2$ consistently incur higher costs than the selected reference setting $(1,2)$.

4.3. Optimizer Sensitivity

Based on the depth sweep, the configuration $(R_{ZZ}, R_{RA}) = (1, 2)$ is selected as the reference depth for optimizer comparisons. Table 5 reports metrics aggregated across multiple random seeds.

Table 5. Optimizer comparison at $(R_{ZZ}, R_{RA}) = (1, 2)$ (mean ± std over seeds).

Optimizer	Accuracy	F1-Score	Iterations	Time (s)
L-BFGS-B	**0.893 ± 0.001**	**0.894 ± 0.001**	**12**	**2999**
COBYLA	0.887 ± 0.004	0.887 ± 0.005	75	1233
SPSA	0.883 ± 0.001	0.884 ± 0.001	300	15,496

Bold values indicate the selected configuration that achieved the best performance in this experiment and was subsequently fixed in other experimental settings.

L-BFGS-B achieves the highest mean F1-score (0.894). In terms of optimizer dynamics, the average iteration counts differ substantially: L-BFGS-B converges in approximately 12 iterations, COBYLA performs 75 iterations, and SPSA reaches the maximum iteration budget (300).

Although L-BFGS-B converges in far fewer iterations than COBYLA, wall-clock time does not scale linearly with the number of iterations. In VQC training, the dominant cost is the number of circuit evaluations required by the optimizer to estimate the objective value at each iteration. Quasi-Newton methods such as L-BFGS-B may internally perform additional objective evaluations, making each iteration comparatively expensive. By contrast, COBYLA is a derivative-free trust-region method that often updates parameters using a lighter-weight sequence of function evaluations per step, which can reduce per-iteration cost even when the total iteration count is larger. As a result, COBYLA can achieve a shorter wall-clock time despite requiring more iterations, while SPSA incurs the largest runtime due to consuming the full iteration budget and repeatedly querying the circuit throughout the stochastic approximation procedure.

Figure 4 visualizes representative optimization trajectories. L-BFGS-B exhibits a steep early decrease in objective value and stabilizes within a few iterations, consistent with fast convergence under analytic/statevector execution. COBYLA also reduces the objective rapidly at the beginning, but shows larger transient fluctuations before stabilizing at a comparable final objective level. SPSA shows an initial drop to a similar objective value within the first few iterations, followed by a long plateau with only small variations. This behavior is consistent with SPSA quickly reaching a neighborhood of a stationary region under the chosen hyperparameters, after which the stochastic gradient approximation yields limited further improvement in the deterministic setting.

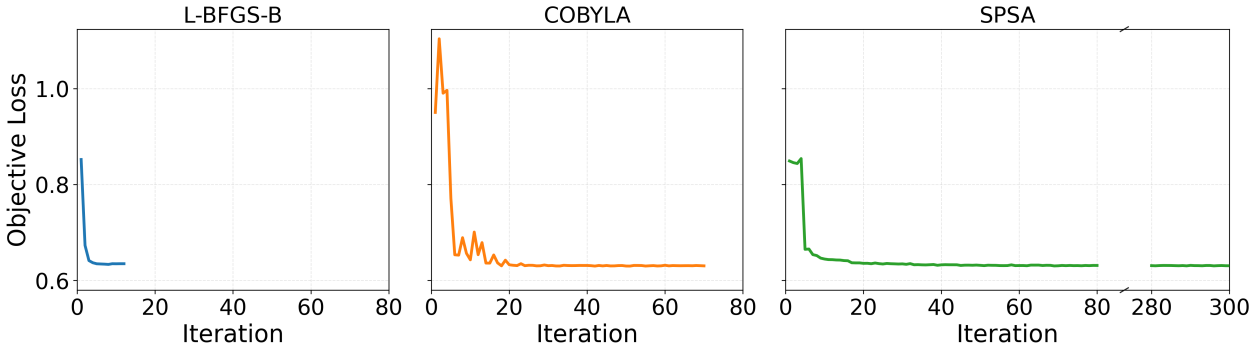

Figure 4. Representative training loss trajectories for optimizers at $R_{ZZ} = 1$, $R_{RA} = 2$. For SPSA, the break in the x-axis indicates an omitted iteration range (80–280) for visualization clarity.

4.4. Training Data Fraction Sensitivity

To assess robustness under reduced training data, the training set fraction is varied while keeping the circuit depth fixed at $(1, 2)$ and using the same set of optimizers. Table 6 reports results for training data fractions 0.5, 0.2, and 0.1.

Table 6. Effect of training data fraction.

Optimizer	Train Fraction	Accuracy	F1-Score	Iterations	Time (s)
L-BFGS-B	0.5	0.892 ± 0.003	0.893 ± 0.004	13	1645
	0.2	0.894 ± 0.001	0.895 ± 0.001	13	667
	0.1	0.702 ± 0.001	0.698 ± 0.001	14	352
COBYLA	0.5	0.888 ± 0.007	0.888 ± 0.008	76	650
	0.2	0.888 ± 0.004	0.888 ± 0.005	73	249
	0.1	0.669 ± 0.047	0.667 ± 0.048	74	130
SPSA	0.5	0.887 ± 0.001	0.886 ± 0.001	300	7991
	0.2	0.889 ± 0.001	0.889 ± 0.001	300	3129
	0.1	0.703 ± 0.001	0.699 ± 0.001	300	1574

A notable difference across optimizers emerges in variability under the low-data regime. At a 0.1 training fraction, COBYLA exhibits substantially larger run-to-run variation (F1-score 0.667 ± 0.048) than L-BFGS-B (0.698 ± 0.001) and SPSA (0.699 ± 0.001). This suggests that, when training data are scarce, optimizer dynamics can materially affect stability even if mean performance is similar.

Training diagnostics further clarify the cost–stability trade-off. L-BFGS-B converges within a small number of iterations (approximately 13–14) across all fractions, while COBYLA performs roughly 73–76 iterations consistently, and SPSA reaches the maximum iteration budget (300) for all fractions. Runtime decreases as the training fraction is reduced for all optimizers, but the relative ordering is preserved: SPSA remains the most expensive option (e.g., 7991 s at 0.5) despite a comparable predictive performance to COBYLA and L-BFGS-B in the moderate-data regimes.

4.5. Scaling Range Sensitivity

To characterize sensitivity to min–max scaling ranges, experiments compare multiple target ranges, including $[0, 1]$, $[0, \pi]$, $[0, 2\pi]$, and $[-\pi, \pi]$. Since angle-based encodings directly map normalized values to gate parameters, the scaling range determines the effective parameter regime explored during optimization. Table 7 reports the resulting performance and training diagnostics under a fixed VQC configuration.

Table 7. Scaling range sensitivity at the reference VQC configuration.

Scaling Range	Accuracy	Precision	Recall	F1-Score	Iterations	Time (s)
$[0, 1]$	0.558	0.545	0.700	0.613	16	4354
$[0, \pi]$	0.727	0.682	0.849	0.756	13	3552
$\mathbf{[0, 2\pi]}$	**0.893**	**0.886**	**0.901**	**0.894**	**12**	**2999**
$[-\pi, \pi]$	0.891	0.888	0.894	0.891	17	4606

Bold values indicate the selected configuration that achieved the best performance in this experiment and was subsequently fixed in other experimental settings.

As shown in Table 7, the scaling range has a first-order impact on classification performance. The narrow range $[0, 1]$ yields the lowest performance (accuracy 0.558, F1-score 0.613), with a precision–recall imbalance (precision 0.545 vs. recall 0.700), indicating that restricting angles to a small interval can limit the effective separability learned by the

circuit in this setting. Expanding the range to $[0, \pi]$ substantially improves performance (accuracy 0.727, F1-score 0.756), suggesting that a wider angular domain enables more discriminative decision boundaries while remaining trainable.

The strongest performance is obtained for the full-period range $[0, 2\pi]$ (accuracy 0.893, F1-score 0.894), with a balanced precision–recall profile (precision 0.886, recall 0.901). The centered range $[-\pi, \pi]$ achieves a similarly high accuracy (0.891) but a slightly lower F1-score (0.891), with marginally higher precision (0.888) and lower recall (0.894) compared to the $[0, 2\pi]$ setting. Notably, the optimization diagnostics differ across scaling choices: $[0, 2\pi]$ converges in fewer iterations (12) and a lower runtime (2999 s) than $[-\pi, \pi]$ (17 iterations, 4606 s), while $[0, \pi]$ achieves a moderate performance with a relatively low runtime (3552 s).

These results indicate that scaling range selection is not a cosmetic preprocessing choice but a core configuration parameter for VQC training. In the evaluated setting, ranges that cover a full 2π span yield the best performance and the most efficient convergence, whereas overly narrow ranges can markedly degrade accuracy and F1-score.

4.6. Finite-Shot and Noisy Execution Results

Finite-shot sampling and noisy execution conditions are evaluated to approximate deployment beyond idealized simulation. In this setting, the trained model is held fixed, and performance is measured under repeated inference runs with different sampling seeds.

4.6.1. Finite-Shot Sensitivity

Table 8 reports inference performance as a function of the shot budget under noiseless conditions. An analytic baseline (Ideal) is included for reference. As the shot budget increases, both accuracy and F1-score improve, and the variability across repeated runs decreases. At very low shot budgets (e.g., 16 and 32), the min–max range indicates large run-to-run fluctuations, whereas the dispersion becomes small at 512 and 1024 shots. At 1024 shots, the mean F1-score (0.892) approaches the analytic baseline (0.894), indicating that sampling-induced variability becomes negligible at this budget for the studied configuration.

Table 8. Inference robustness under finite shots.

Shots	Accuracy	F1-Score	Accuracy (Min–Max)	F1 (Min–Max)
Ideal	0.893 ± 0.001	0.894 ± 0.001	0.893–0.893	0.894–0.894
16	0.839 ± 0.033	0.835 ± 0.037	0.768–0.878	0.752–0.876
32	0.855 ± 0.024	0.854 ± 0.024	0.805–0.877	0.803–0.875
64	0.867 ± 0.018	0.867 ± 0.018	0.835–0.885	0.833–0.885
128	0.875 ± 0.011	0.876 ± 0.011	0.855–0.894	0.855–0.895
256	0.886 ± 0.007	0.887 ± 0.008	0.874–0.895	0.873–0.896
512	0.887 ± 0.006	0.888 ± 0.006	0.878–0.896	0.880–0.898
1024	0.891 ± 0.004	0.892 ± 0.004	0.884–0.897	0.886–0.898

4.6.2. Noise Model from Real Device

To complement noiseless shot sweeps, a hardware-informed noise emulation is considered using a noise model derived from an IBM backend (ibm_fez). A noise-strength scaling experiment is conducted by multiplying the calibrated hardware error rates by a scale factor. Figure 5 summarizes the combined effect of shot budgets and noise scaling on classification performance.

Across the evaluated conditions, shot budget is the dominant factor affecting performance. Low-shot regimes (16 and 32 shots) exhibit a noticeably lower F1-score compared with higher-shot regimes (256 and 1024 shots), regardless of noise scaling. In contrast, increasing the noise scale produces only gradual changes in performance, while the ordering

across shot budgets remains consistent. These results indicate that sampling uncertainty is the primary limiting factor in low-shot regimes, whereas moderate increases in hardware-level noise do not induce abrupt performance degradation.

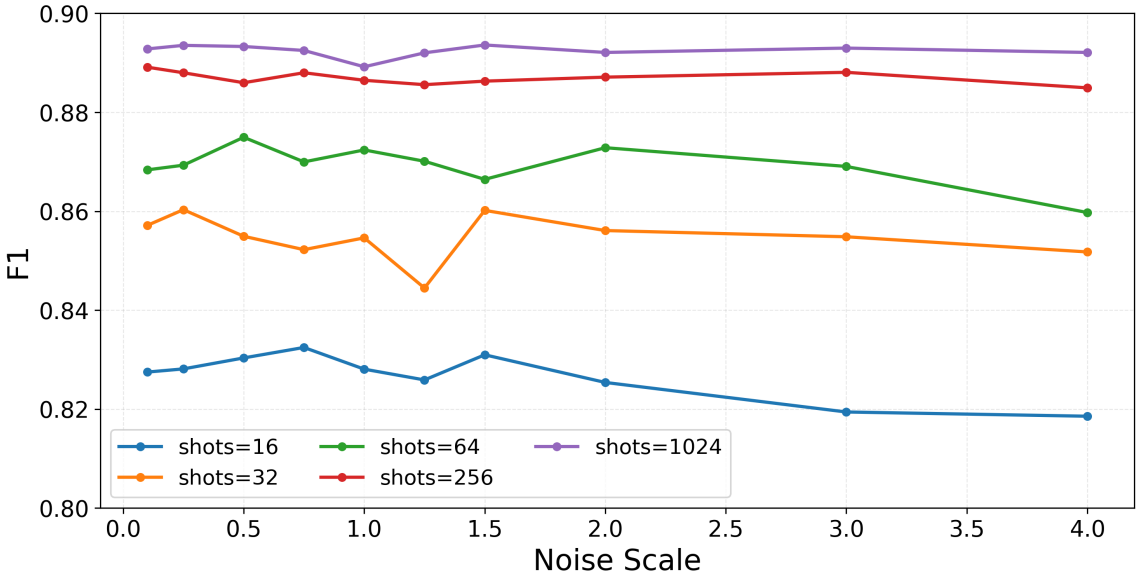

Figure 5. F1-score under combined shot budgets and ibm_fez-derived noise scaling.

Table 9 provides detailed numerical results at 1024 shots, together with the corresponding physical error-rate parameters of the ibm_fez noise model. Here, e_{1q}, e_{2q}, and e_{ro} denote the average error rates of single-qubit gates, two-qubit entangling gates, and measurement readout, respectively. These parameters are uniformly scaled according to the applied noise factor to emulate progressively stronger noise conditions relative to the calibrated hardware baseline.

Table 9. Noise-scale sweep at 1024 shots using an ibm_fez-derived noise model.

Scale	Accuracy	F1-Score	e_{1q}	e_{2q}	e_{ro}
0.10	0.892	0.893	0.000031	0.000263	0.001447
0.25	0.892	0.894	0.000078	0.000658	0.003616
0.50	0.892	0.893	0.000155	0.001315	0.007233
0.75	0.892	0.893	0.000233	0.001973	0.010849
1.00	0.888	0.889	0.000310	0.002631	0.014465
1.25	0.891	0.892	0.000388	0.003289	0.018082
1.50	0.892	0.894	0.000466	0.003946	0.021698
2.00	0.891	0.892	0.000621	0.005262	0.028931
3.00	0.892	0.893	0.000931	0.007893	0.043396
4.00	0.891	0.892	0.001242	0.010523	0.057861

At 1024 shots, performance remains stable across the tested noise scaling range, despite substantial increases in gate and readout error rates. This observation further confirms that, under sufficient sampling budgets, the evaluated VQC configuration maintains a robust classification performance under realistic hardware noise conditions.

4.6.3. Comparison with Representative Classical Baselines

Prior studies have shown that dummy-oriented detection can be performed effectively by high-capacity classical learning models, especially deep architectures designed to distinguish genuine operations from inserted dummy operations [53,54]. Accordingly, Table 10

contextualizes the present VQC under a capacity-controlled setting. To this end, representative classical baselines, such as support vector machine (SVM) [68], multilayer perceptron (MLP) [69], recurrent neural network (RNN) [70], and bidirectional RNN (Bi-RNN) [71], are configured with intentionally small parameter counts that remain within the same order of magnitude as the reference VQC.

Table 10. Comparison with classical baselines.

Model	Accuracy	F1-Score	Trainable Parameters
Linear SVM	0.570 ± 0.000	0.520 ± 0.000	3
MLP	0.720 ± 0.051	0.733 ± 0.061	13
RNN	0.831 ± 0.004	0.855 ± 0.004	13
Bi-RNN	0.805 ± 0.004	0.835 ± 0.002	11
VQC (Proposed)	$\mathbf{0.893 \pm 0.001}$	$\mathbf{0.894 \pm 0.001}$	**6**

Bold values indicate the proposed model.

Under this configuration, the VQC achieves the best performance among the compared models, with an accuracy of 0.893 and an F1-score of 0.894, using six trainable parameters. The linear SVM is insufficient for the nonlinear structure of the task, while the compact MLP improves performance but remains clearly below the VQC. The strongest classical baselines in this setting are the recurrent variants, with the RNN achieving 0.831 accuracy and 0.855 F1-score, and the Bi-RNN achieving 0.805 accuracy and 0.835 F1-score; yet, both still underperform compared to the VQC despite using a comparable or larger number of parameters. These results suggest that, in the present minimal benchmark, the canonical VQC provides a favorable expressivity–capacity trade-off relative to the tested small classical baselines.

Nevertheless, this comparison should be interpreted with caution. It does not imply that VQCs universally outperform classical models for dummy trace detection, as prior work already indicates that larger classical architectures can perform very strongly on hiding-related distinguishability tasks [53,54]. Instead, the results suggest that the current VQC has not exhausted its modeling capacity at this scale and motivate future investigation of larger VQC configurations, richer encodings, and higher-dimensional inputs. While this is not evidence of a general scaling law, it lends support to the hypothesis that expanded VQC models may be worth exploring for more demanding side-channel workloads, including key-inference settings and SCBD, where the target structure is richer than binary dummy discrimination and may require stronger representational power [8,41]. Design-time and system-level SCBD studies further suggest that richer workloads remain relevant evaluation targets [49,50].

5. Discussion

5.1. Depth-Dependent Regimes and Performance Trends

The depth sweep highlights that circuit depth is a first-order factor in VQC behavior, but improvements are not monotonic [4]. In the tested grid, the best performance is obtained at a moderate depth, $(R_{ZZ}, R_{RA}) = (1, 2)$, achieving an F1-score of 0.894. Increasing the ansatz depth to $R_{RA} = 3$ at the same feature-map depth yields a comparable but slightly lower F1-score (0.889), while the shallow ansatz setting $(1, 1)$ performs substantially worse (F1-score 0.583). These results suggest that a minimal ansatz may be insufficient to model the discriminative structure present in the power traces, whereas moderate expressivity is beneficial [30].

By contrast, increasing the feature-map depth from $R_{ZZ} = 1$ to $R_{ZZ} = 2$ reduces performance across all tested ansatz depths in the RSA sweep. This behavior is consistent with

the broader understanding that deeper variational constructions can introduce optimization difficulty and sensitivity in NISQ-era regimes [4,31]. While the present experiments are conducted with a small number of qubits, the observed degradation indicates that additional data re-uploading and an entangling structure in the feature map can still shift the optimization landscape in unfavorable ways, even at a small scale.

From a practical perspective, the depth sweep supports a conservative strategy for configuration selection: begin with a shallow feature map and tune ansatz depth within a small range. For the present workload, $R_{ZZ} = 1$ combined with $R_{RA} \in \{2, 3\}$ forms a stable high-performance region, whereas $R_{ZZ} = 2$ and $R_{RA} = 1$ are comparatively unreliable settings in this testbed.

5.2. Optimizer Choice

Optimizer selection materially changes both convergence behavior and computational cost, even under the same circuit structure. At the selected depth $(1, 2)$, L-BFGS-B attains the highest mean F1-score (0.894 ± 0.001), followed by COBYLA (0.887 ± 0.005) and SPSA (0.884 ± 0.001). Although the mean performance gap is moderate, training dynamics differ sharply. L-BFGS-B converges in roughly 12 iterations on average, COBYLA requires about 75 iterations, and SPSA reaches the maximum iteration budget (300) in the reported runs.

Runtime diagnostics indicate that iteration counts alone are insufficient to compare optimizers, as per-iteration costs can differ substantially. In the experiments, SPSA incurs a markedly larger wall-clock cost than the others, while COBYLA remains the fastest despite a larger iteration count, reflecting the practical importance of measuring cost in addition to final predictive metrics. These observations support treating optimizer choice as an integral part of the VQC configuration space rather than as a secondary implementation detail [3,4].

A pragmatic takeaway is that L-BFGS-B is a strong default for small VQC instances when analytic execution is available and stable optimization is desired, whereas COBYLA can be attractive for rapid scanning due to its low wall-clock cost. SPSA remains relevant for scenarios where stochasticity is unavoidable, but the high iteration cost observed here motivates careful budgeting and early-stopping criteria when using SPSA.

5.3. Data-Efficiency Behavior Under Reduced Training Fractions

Training data fraction sweeps show that performance is relatively stable in moderate-data regimes but degrades under aggressive data reduction [8,38]. At training fractions of 0.5 and 0.2, F1 remains near the full-data regime for all three optimizers, with limited run-to-run variability except for modest fluctuations under COBYLA. At a training fraction of 0.1, mean F1 decreases substantially to 0.667–0.699, and COBYLA exhibits noticeably larger variability.

This pattern suggests that, for the present representation and circuit family, the learned decision boundary depends on a nontrivial amount of labeled data to remain stable. For experimental design, this supports explicitly reporting training fractions and discouraging interpreting single-run results as representative in low-data settings. More broadly, this finding reinforces the importance of reporting mean and variance statistics when characterizing VQC behavior, as data scarcity can amplify stochasticity and increase sensitivity to initialization [4].

5.4. Implications of Scaling Ranges and NISQ Constraints

Scaling-range sensitivity is a first-order factor because angle-based encodings map normalized values directly to gate parameters [2,29]. In our experiments, $[0, 2\pi]$ yields the strongest performance (F1 = 0.894), while overly narrow ranges such as $[0, 1]$ markedly degrade separability (F1 = 0.613). Intermediate ranges $[0, \pi]$ show intermediate behavior, and the centered range $[-\pi, \pi]$ achieves similar accuracy but slightly lower F1 than $[0, 2\pi]$.

These results indicate that scaling range is not a cosmetic preprocessing choice but a core configuration parameter that should be explicitly documented.

Regarding deployment-relevant constraints, inference-time sweeps with fixed trained parameters show that shot budget is the dominant driver of performance variability: low-shot regimes (e.g., 16–32 shots) produce substantial run-to-run fluctuations, while budgets of 512–1024 shots approach the analytic baseline and yield stable predictions. Under sufficient shot budgets, an ibm_fez-derived noise model with scaled error rates induces only gradual performance changes, suggesting that sampling uncertainty can be a more immediate bottleneck than moderate noise increases for the studied configuration [3].

Runtime should likewise be interpreted with care when considering practical deployment. The execution times reported in this study were measured as simulator-side wall-clock costs accumulated over repeated hybrid optimization steps and are, therefore, most informative as comparative indicators across configurations rather than as direct estimates of real-hardware training latency. In actual quantum platforms, circuit execution occurs on much shorter physical timescales, whereas end-to-end runtime can still be affected by shot repetitions, transpilation, and backend-level overhead. Accordingly, the present results suggest that practical feasibility depends not only on final predictive performance but also on how configuration choices interact with sampling cost, optimization burden, and the execution environment.

5.5. Limitations and Avenues for Future Work

Several limitations should be considered when interpreting the findings. First, the present study uses a deliberately minimal two-dimensional input representation based on the normalized sequence position and power value. This design choice is useful for isolating configuration-level effects in a controlled setting, but it also limits generalization by constraining both circuit size and encoding complexity. Therefore, the current benchmark does not fully capture the trainability challenges that may arise when richer side-channel representations require more qubits or deeper data-encoding structures.

In particular, the conclusions regarding depth-dependent behavior, optimizer preference, scaling-range sensitivity, and data efficiency are most directly applicable to low-dimensional side-channel representations of this form. If longer trace windows, full-trace inputs, or higher-dimensional encodings are used instead, the amount of information exposed to the model, the required number of qubits, the encoding strategy, and the effective optimization landscape may all change substantially. Under such conditions, the favorable configuration regime identified in this study, including the preference for a shallow feature map with moderate ansatz depth, may shift, and the relative stability or efficiency of the tested optimizers may also differ. Therefore, the current findings should be interpreted as benchmark-level evidence for VQC behavior in a minimal and interpretable setting, rather than as a universal conclusion for all side-channel input formulations.

Second, the primary results are derived from a simulated quantum computing environment and analytic execution. Although this study separately evaluates finite-shot inference and noisy execution, the optimizer comparison and most systematic configuration sweeps are still conducted under simulator-oriented conditions. As a result, the reported convergence speed, iteration counts, and wall-clock runtime ordering should not be interpreted as directly hardware-equivalent performance indicators. On real quantum backends, backend-specific noise structure, calibration drift, job latency, and repeated circuit execution overhead may alter both trainability and computational cost, especially in low-shot regimes where sampling fluctuations are already pronounced. Accordingly, the present results provide controlled evidence for configuration sensitivity under simulated

NISQ-like conditions, but additional hardware validation is required before extending these efficiency-related conclusions to deployment settings.

Third, only a single circuit family and a limited depth grid are explored. Accordingly, the reported design guidance is specific to the chosen feature map and ansatz combination. In particular, the observed preference for a moderate ansatz depth, the degradation under deeper feature-map settings, and the sensitivity to scaling range may depend on the expressivity and inductive bias of the selected circuit family. Alternative encodings, ansatz families, initialization schemes, and regularization strategies may yield different expressivity–trainability trade-offs and may shift the boundary between stable and unstable operating regimes. Therefore, the current recommendations should be interpreted as configuration-aware observations for the selected canonical VQC setting, rather than as architecture-independent guidance.

Future work can address these limitations by extending inputs to windowed or higher-dimensional representations while managing qubit constraints via dimensionality reduction or structured encodings, expanding circuit families and studying entanglement patterns and re-uploading strategies, performing shot- and noise-aware training and evaluating error mitigation techniques, and running selected configurations on real quantum backends to validate sensitivity trends under hardware calibration dynamics. These directions would further strengthen practical guidance for applying variational models to security-relevant side-channel data.

6. Conclusions

This paper presented a controlled experimental study of VQCs on a security-relevant side-channel workload. Specifically, it presented a VQC-based binary classification formulation for distinguishing real operations from dummy operations in power-based side-channel traces, established dummy trace detection as a controlled benchmark for studying QML behavior on side-channel data, and systematically characterized how key VQC design choices and NISQ-era execution constraints affect performance and robustness. Using a small number of qubit inputs and a canonical circuit family, systematic experiments were conducted across circuit depth, optimizers, and training-data fractions.

The experiments identified clear configuration-dependent regimes. Performance improved substantially when increasing the ansatz depth from $R_{\mathrm{RA}} = 1$ to $R_{\mathrm{RA}} = 2$, whereas further increasing to $R_{\mathrm{RA}} = 3$ yielded no additional benefit within the evaluated range while incurring a higher runtime. In contrast, increasing the feature-map repetitions consistently degraded performance, indicating that deeper data encoding is not necessarily beneficial even in few-qubit settings. Optimizer selection significantly affected both convergence behavior and runtime cost. Training data fraction sweeps showed that performance remained stable at moderate fractions but declined substantially at aggressive reductions, reinforcing the need for variability-aware reporting.

Overall, the study provides empirically grounded guidance for configuring VQCs on side-channel inputs: depth and optimizer choices should be treated as primary knobs, and deployment-relevant constraints such as scaling ranges, finite shots, and noise should be explicitly documented and evaluated. These findings help clarify the practical behavior and limitations of QML models in a security-relevant dataset setting and support more reproducible, configuration-aware experimentation at the intersection of QML and SCA.

Author Contributions: Conceptualization, S.P. and Y.S.; methodology, S.P. and Y.S.; software, S.P.; validation, S.P.; formal analysis, S.P. and Y.S.; investigation, S.P.; resources, S.P.; data curation, S.P.; writing—original draft preparation, S.P.; writing—review and editing, S.P. and Y.S.; visualization, S.P.; supervision, Y.S.; project administration, Y.S.; funding acquisition, Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2026-RS-2020-II201789) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation). This work was supported by the IITP (Institute of Information & Communications Technology Planning & Evaluation)-ICAN (ICT Challenge and Advanced Network of HRD) grant funded by the Korea government (Ministry of Science and ICT) (IITP-2026-RS-2023-00260248). This study was supported by the commercialization promotion agency for R&D outcomes grant funded by the Korea government(MSIT) (RS-2025-02315174). This work was supported by the Commercialization Promotion Agency for R&D Outcomes (COMPA) grant funded by the Korea government (Ministry of Science and ICT) (RS-2025-02311988).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The SCADataset used in this study is publicly available at https://github.com/PLASS-Lab/SCADataset (accessed on 27 February 2026) [54].

Acknowledgments: A preliminary version of this work was presented at the 26th International Symposium on Advanced Intelligent Systems (ISIS2025), Cheongju, Korea, under the title "Quantum Neural Network Approach for Identifying Dummy Power Traces in Side-Channel Analysis". This article substantially extends the conference paper through additional experiments, expanded analysis, and significant methodological refinements.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Biamonte, J.; Wittek, P.; Pancotti, N.; Rebentrost, P.; Wiebe, N.; Lloyd, S. Quantum Machine Learning. *Nature* **2017**, *549*, 195–202. [CrossRef] [PubMed]
2. Schuld, M.; Petruccione, F. *Supervised Learning with Quantum Computers*, 1st ed.; Springer: Cham, Switzerland, 2018. [CrossRef]
3. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
4. Cerezo, M.; Arrasmith, A.; Babbush, R.; Benjamin, S.; Endo, S.; Fujii, K.; Mcclean, J.; Mitarai, K.; Yuan, X.; Cincio, L.; et al. Variational quantum algorithms. *Nat. Rev. Phys.* **2021**, *3*, 625–644. [CrossRef]
5. Küçükkara, M.Y.; Atban, F.; Bayılmış, C. Quantum-Neural Network Model for Platform Independent Ddos Attack Classification in Cyber Security. *Adv. Quantum Technol.* **2024**, *7*, 2400084. [CrossRef]
6. Kim, J.; Park, S.; Cha, J.; Son, E.; Son, Y. Novel Synthetic Dataset Generation Method with Privacy-Preserving for Intrusion Detection System. *Appl. Sci.* **2025**, *15*, 10609. [CrossRef]
7. Nguyen, D.H.; Seo, A.; Nnamdi, N.P.; Son, Y. False Alarm Reduction Method for Weakness Static Analysis Using BERT Model. *Appl. Sci.* **2023**, *13*, 3502. [CrossRef]
8. Picek, S.; Perin, G.; Mariot, L.; Wu, L.; Batina, L. SoK: Deep Learning-based Physical Side-channel Analysis. *ACM Comput. Surv.* **2023**, *55*, 227. [CrossRef]
9. Alabdulwahab, S.; Cheong, M.; Seo, A.; Kim, Y.T.; Son, Y. Enhancing deep learning-based side-channel analysis using feature engineering in a fully simulated IoT system. *Expert Syst. Appl.* **2025**, *266*, 126079. [CrossRef]
10. Son, Y.; Lee, Y. A Smart Contract Weakness and Security Hole Analyzer Using Virtual Machine Based Dynamic Monitor. *J. Logist. Inform. Serv. Sci.* **2022**, *9*, 36–52. [CrossRef]
11. Abreu, D.; Rothenberg, C.E.; Abelém, A. QML-IDS: Quantum Machine Learning Intrusion Detection System. In *Proceedings of the 2024 IEEE Symposium on Computers and Communications (ISCC), Paris, France, 26–29 June 2024*; IEEE: New York, NY, USA, 2024; pp. 1–6. [CrossRef]
12. Sarvade, V.P.; Kulkarni, S.A.; Raj, C.V. A Hybrid Classical-Quantum Neural Network Model for DDoS Attack Detection in Software-Defined Vehicular Networks. *Information* **2025**, *16*, 722. [CrossRef]
13. Eze, L.; Chaudhry, U.B.; Jahankhani, H. Quantum-Enhanced Machine Learning for Cybersecurity: Evaluating Malicious URL Detection. *Electronics* **2025**, *14*, 1827. [CrossRef]
14. Shahriyar, M.F.; Tanbhir, G.; Raihan Chy, A.M.; Tanzin, M.A.A.A.; Mashrafi, M.J. PhishVQC: Optimizing Phishing URL Detection with Correlation Based Feature Selection and Variational Quantum Classifier. In *Proceedings of the 2025 3rd International Conference on Intelligent Systems, Advanced Computing and Communication (ISACC), Silchar, India, 27–28 February 2025*; IEEE: New York, NY, USA, 2025; pp. 1226–1231. [CrossRef]
15. Barrué, G.; Quertier, T. Quantum Machine Learning for Malware Classification. *arXiv* **2023**, arXiv:2305.09674. [CrossRef]

16. Ogiesoba-Eguakun, O.C.; Rath, S. Quantum Machine Learning Approaches for Coordinated Stealth Attack Detection in Distributed Generation Systems. *arXiv* **2025**, arXiv:2601.00873.
17. Liu, R.; Eren, M.; Nicholas, C. Can Feature Engineering Help Quantum Machine Learning for Malware Detection? *arXiv* **2023**, arXiv:2305.02396. [CrossRef]
18. Rahman, M.A.; Akter, M.S.; Miller, E.; Timofti, B.; Shahriar, H.; Masum, M.; Wu, F. Fine-Tuned Variational Quantum Classifiers for Cyber Attacks Detection Based on Parameterized Quantum Circuits and Optimizers. In *Proceedings of the 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC), Osaka, Japan, 2–4 July 2024*; IEEE: New York, NY, USA, 2024; pp. 1067–1072. [CrossRef]
19. Bellante, A.; Fioravanti, T.; Carminati, M.; Zanero, S.; Luongo, A. Evaluating the potential of quantum machine learning in cybersecurity: A case-study on PCA-based intrusion detection systems. *Comput. Secur.* **2025**, *154*, 104341. [CrossRef]
20. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017*; Curran Associates: Red Hook, NY, USA, 2017; pp. 6000–6010. [CrossRef]
21. Baevski, A.; Zhou, H.; Mohamed, A.; Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020*; Curran Associates: Red Hook, NY, USA, 2020; pp. 12449–12460. [CrossRef]
22. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; pp. 213–229. [CrossRef]
23. Chen, S.; Sun, P.; Song, Y.; Luo, P. DiffusionDet: Diffusion Model for Object Detection. In *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023*; IEEE: New York, NY, USA, 2023; pp. 19773–19786. [CrossRef]
24. Shen, X.; Wang, Y.; Ma, Y.; Li, L.; Niu, Y.; Yang, Z.; Shi, Y. A multi-expert diffusion model for surface defect detection of valve cores in special control valve equipment systems. *Mech. Syst. Signal Process.* **2025**, *237*, 113117. [CrossRef]
25. Kocher, P.; Jaffe, J.; Jun, B. Differential Power Analysis. In *Proceedings of the Advances in Cryptology—CRYPTO' 99, Santa Barbara, CA, USA, 15–19 August 1999*; Wiener, M., Ed.; Springer: Berlin/Heidelberg, Germany, 1999; pp. 388–397. [CrossRef]
26. Mangard, S.; Oswald, E.; Popp, T. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*; Springer: Berlin/Heidelberg, Germany, 2007. [CrossRef]
27. Quisquater, J.J.; Samyde, D. ElectroMagnetic Analysis (EMA): Measures and Counter-measures for Smart Cards. In *Proceedings of the Smart Card Programming and Security, Cannes, France, 19–21 September 2001*; Attali, I., Jensen, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2001; pp. 200–210. [CrossRef]
28. Lee, J.; Han, D.G. Security analysis on dummy based side-channel countermeasures—Case study: AES with dummy and shuffling. *Appl. Soft Comput.* **2020**, *93*, 106352. [CrossRef]
29. Havlíček, V.; Córcoles, A.; Temme, K.; Harrow, A.; Kandala, A.; Chow, J.; Gambetta, J. Supervised learning with quantum-enhanced feature spaces. *Nature* **2019**, *567*, 209–212. [CrossRef]
30. Schuld, M.; Sweke, R.; Meyer, J.J. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A* **2021**, *103*, 032430. [CrossRef]
31. McClean, J.; Boixo, S.; Smelyanskiy, V.; Babbush, R.; Neven, H. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* **2018**, *9*, 4812. [CrossRef]
32. Park, S.; Kuai, J.; Kim, H.; Ko, H.; Jung, C.; Son, Y. A Lightweight Degradation-Aware Framework for Robust Object Detection in Adverse Weather. *Electronics* **2026**, *15*, 146. [CrossRef]
33. Park, S.; Kim, J.; Son, Y. Quantum Neural Network Approach for Identifying Dummy Power Traces in Side-Channel Analysis. In Proceedings of the 26th International Symposium on Advanced Intelligent Systems (ISIS), Cheongju, Republic of Korea, 6–9 November 2025.
34. Hospodar, G.; Gierlichs, B.; Mulder, E.D.; Verbauwhede, I.; Vandewalle, J. Machine learning in side-channel analysis: A first study. *J. Cryptogr. Eng.* **2011**, *1*, 293–302. [CrossRef]
35. Picek, S.; Heuser, A.; Jovic, A.; Ludwig, S.A.; Guilley, S.; Jakobovic, D.; Mentens, N. Side-channel analysis and machine learning: A practical perspective. In *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017*; IEEE: New York, NY, USA, 2017; pp. 4095–4102. [CrossRef]
36. Picek, S.; Heuser, A.; Guilley, S. Template attack versus Bayes classifier. *J. Cryptogr. Eng.* **2017**, *7*, 343–351. [CrossRef]
37. Wu, L.; Perin, G.; Picek, S. The Best of Two Worlds: Deep Learning-assisted Template Attack. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2022**, *2022*, 413–437. [CrossRef]
38. Masure, L.; Dumas, C.; Prouff, E. A Comprehensive Study of Deep Learning for Side-Channel Analysis. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**, *2020*, 348–375. [CrossRef]

39. Cagli, E.; Dumas, C.; Prouff, E. Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures. In *Proceedings of the Cryptographic Hardware and Embedded Systems—CHES 2017, Taipei, Taiwan, 25–28 September 2017*; Fischer, W., Homma, N., Eds.; Springer: Cham, Switzerland, 2017; pp. 45–68. [CrossRef]
40. Zaid, G.; Bossuet, L.; Habrard, A.; Venelli, A. Methodology for Efficient CNN Architectures in Profiling Attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**, *2020*, 1–36. [CrossRef]
41. Park, J.; Xu, X.; Jin, Y.; Forte, D.; Tehranipoor, M. Power-based Side-Channel Instruction-level Disassembler. In *Proceedings of the 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 24–28 June 2018*; IEEE: New York, NY, USA, 2018; pp. 1–6. [CrossRef]
42. Park, J.; Tyagi, A. Using Power Clues to Hack IoT Devices: The power side channel provides for instruction-level disassembly. *IEEE Consum. Electron. Mag.* **2017**, *6*, 92–102. [CrossRef]
43. Strobel, D.; Bache, F.; Oswald, D.; Schellenberg, F.; Paar, C. SCANDALee: A side-ChANnel-based DisAssembLer using local electromagnetic emanations. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 9–13 March 2015*; IEEE: New York, NY, USA, 2015; pp. 139–144. [CrossRef]
44. Eisenbarth, T.; Paar, C.; Weghenkel, B. Building a Side Channel Based Disassembler. In *Transactions on Computational Science X: Special Issue on Security in Computing, Part I*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 78–99. [CrossRef]
45. Krishnankutty, D.; Li, Z.; Robucci, R.; Banerjee, N.; Patel, C. Instruction Sequence Identification and Disassembly Using Power Supply Side-Channel Analysis. *IEEE Trans. Comput.* **2020**, *69*, 1639–1653. [CrossRef]
46. Narimani, P.; Akhaee, M.A.; Habibi, S.A. Side-Channel based Disassembler for AVR Micro-Controllers using Convolutional Neural Networks. In *Proceedings of the 2021 18th International ISC Conference on Information Security and Cryptology (ISCISC), Isfahan, Iran, 1–2 September 2021*; IEEE: New York, NY, USA, 2021; pp. 75–80. [CrossRef]
47. Cristiani, V.; Lecomte, M.; Hiscock, T. A Bit-Level Approach to Side Channel Based Disassembling. In *Proceedings of the Smart Card Research and Advanced Applications, Prague, Czech Republic, 11–13 November 2019*; Belaïd, S., Güneysu, T., Eds.; Springer: Cham, Switzerland, 2020; pp. 143–158. [CrossRef]
48. van Geest, J.; Buhan, I. A Side-Channel Based Disassembler for the ARM-Cortex M0. In *Proceedings of the Applied Cryptography and Network Security Workshops, Rome, Italy, 20–23 June 2022*; Zhou, J., Adepu, S., Alcaraz, C., Batina, L., Casalicchio, E., Chattopadhyay, S., Jin, C., Lin, J., Losiouk, E., Majumdar, S., et al., Eds.; Springer: Cham, Switzerland, 2022; pp. 183–199. [CrossRef]
49. Fendri, H.; Macchetti, M.; Perrine, J.; Stojilović, M. A deep-learning approach to side-channel based CPU disassembly at design time. In *Proceedings of the 2022 Conference & Exhibition on Design, Automation & Test in Europe, Leuven, BEL, Antwerp, Belgium, 14–23 March 2022*; IEEE: New York, NY, USA, 2022; pp. 670–675. [CrossRef]
50. Maillard, J.; Hiscock, T.; Lecomte, M.; Clavier, C. Side-channel disassembly on a System-on-Chip: A practical feasibility study. *Microprocess. Microsystems* **2023**, *101*, 104904. [CrossRef]
51. Alabdulwahab, S.; Kim, J.; Kim, Y.T.; Son, Y. Advanced Side-Channel Evaluation Using Contextual Deep Learning-Based Leakage Modeling. *ACM Trans. Softw. Eng. Methodol.* **2026**, *35*, 39. [CrossRef]
52. Arguello, C.N.; Searle, H.; Rampazzi, S.; Butler, K.R.B. A Practical Methodology for ML-Based EM Side Channel Disassemblers. *arXiv* **2022**, arXiv:2206.10746. [CrossRef]
53. Lee, J.; Han, D.G. DLDDO: Deep Learning to Detect Dummy Operations. In *Proceedings of the Information Security Applications, Jeju Island, Republic of Korea, 26–28 August 2020*; You, I., Ed.; Springer: Cham, Switzerland, 2020; pp. 73–85. [CrossRef]
54. Park, S.; Seo, A.; Cheong, M.; Kim, H.; Kim, J.; Son, Y. Evaluating the Vulnerability of Hiding Techniques in Cyber-Physical Systems Against Deep Learning-Based Side-Channel Attacks. *Appl. Sci.* **2025**, *15*, 6981. [CrossRef]
55. Shor, P.W. Fault-tolerant quantum computation. *arXiv* **1997**, arXiv:quant-ph/9605011.
56. Larasati, H.T.; Choi, B.S. Towards fault-tolerant distributed quantum computation (FT-DQC): Taxonomy, recent progress, and challenges. *ICT Express* **2025**, *11*, 417–435. [CrossRef]
57. Preskill, J. Fault-tolerant quantum computation. *arXiv* **1997**, arXiv:quant-ph/9712048. [PubMed]
58. Shor, P.W. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A* **1995**, *52*, R2493–R2496. [CrossRef] [PubMed]
59. Steane, A.M. Error Correcting Codes in Quantum Theory. *Phys. Rev. Lett.* **1996**, *77*, 793–797. [CrossRef] [PubMed]
60. Kitaev, A. Fault-tolerant quantum computation by anyons. *Ann. Phys.* **2003**, *303*, 2–30. [CrossRef]
61. Mitarai, K.; Negoro, M.; Kitagawa, M.; Fujii, K. Quantum circuit learning. *Phys. Rev. A* **2018**, *98*, 032309. [CrossRef]
62. Senokosov, A.; Sedykh, A.; Sagingalieva, A.; Kyriacou, B.; Melnikov, A. Quantum machine learning for image classification. *Mach. Learn. Sci. Technol.* **2024**, *5*, 015040. [CrossRef]
63. Noh, D.I.; Jeong, S.G.; Hwang, W.J. Hybrid Quantum ResNet for Time Series Classification. *IEEE Trans. Emerg. Top. Comput.* **2025**, *13*, 1083–1098. [CrossRef]
64. Hur, T.; Kim, L.; Park, D.K. Quantum convolutional neural network for classical data classification. *Quantum Mach. Intell.* **2022**, *4*, 3. [CrossRef]

65. Gong, L.H.; Pei, J.J.; Zhang, T.F.; Zhou, N.R. Quantum convolutional neural network based on variational quantum circuits. *Opt. Commun.* **2024**, *550*, 129993. [CrossRef]
66. Feng, Y.Y.; Li, Y.; Li, J.; Zhou, J.; Shi, J.J. Variational Shadow Quantum Circuits Assisted Quantum Convolutional Neural Network. *Adv. Quantum Technol.* **2025**, *8*, 2400510. [CrossRef]
67. Park, J.; Rahman, F.; Vassilev, A.; Forte, D.; Tehranipoor, M. Leveraging Side-Channel Information for Disassembly and Security. *ACM J. Emerg. Technol. Comput. Syst.* **2019**, *16*, 6. [CrossRef]
68. Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
69. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. In *Neurocomputing: Foundations of Research*; MIT Press: Cambridge, MA, USA, 1988; pp. 696–699. [CrossRef]
70. Elman, J.L. Finding Structure in Time. *Cogn. Sci.* **1990**, *14*, 179–211. [CrossRef]
71. Schuster, M.; Paliwal, K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]