Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X
Volume 4, Issue 5, 2026

ENHANCING BLOCKCHAIN SECURITY USING MACHINE LEARNING-OPTIMIZED` ASYMMETRIC ENCRYPTION: A COMPREHENSIVE FRAMEWORK FOR INTELLIGENT CRYPTOGRAPHIC MANAGEMENT IN DISTRIBUTED LEDGER SYSTEMS

[1]Sobia Akmal, [2]Dr. Amnah Firdous*, [3]Muniba Saleem, [4]Sabeeka Fatima
[1]Department of CS&IT The Government Sadiq College Women University Bahawalpur
[2]Department of CS&IT The Government Sadiq College Women University Bahawalpur
[3]Department of CS&IT The Government Sadiq College Women University Bahawalpur
[4]Department of Computer and Software Engineering, National University of Sciences and Technology, Islamabad
[1]sobiaa858@gmail.com, [2]amnah@gscwu.edu.pk, [3]muniba@gscwu.edu.pk, [4]sabeekafatima84@gmail.com

DOI:

Keywords
Blockchain Security, Machine Learning, Asymmetric Encryption, Cryptographic Algorithm Identification, Deep Learning, Post-Quantum Cryptography, Zero-Knowledge Proofs, Federated Learning, Smart Contracts, Distributed Ledger Technology

Article History
Received on: 14 April 2026
Accepted on: 08 May 2026
Published on: 09 May 2026

Corresponding Author:

[2]Dr. Amnah Firdous

Abstract

Blockchain technology has emerged as a transformative paradigm for decentralized trust and transparent transaction processing across diverse sectors including finance, supply chain, healthcare, and digital identity management. However, the escalating sophistication of cyber threats, coupled with the computational rigidity of conventional cryptographic implementations, presents critical vulnerabilities in contemporary blockchain ecosystems. This paper proposes a novel Machine Learning-Optimized Asymmetric Encryption Framework (ML-OAEF) that integrates advanced supervised learning algorithms with intelligent cryptographic management to enhance blockchain security, scalability, and adaptive resilience. We present a comprehensive methodology encompassing dataset synthesis, multi-dimensional feature engineering, comparative model evaluation, and blockchain-specific security assessment. Four distinct machine learning architectures—Random Forest (RF), Gradient Boosting (GB), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP)—were systematically evaluated against a diverse cryptographic dataset comprising 1,647 samples across symmetric encryption (AES, DES, 3DES, Blowfish, RC4, ChaCha20), asymmetric encryption (RSA), and hash functions (SHA-256). Experimental results demonstrate that the Support Vector Machine and Neural Network models achieved exceptional classification accuracy of 96.5%, significantly outperforming traditional baseline approaches. We introduce a Composite Blockchain Security Score (CBSS) metric that quantifies cryptographic suitability across five dimensions: cryptographic strength, performance efficiency, quantum resistance, blockchain compatibility, and machine learning confidence. Furthermore, we propose a Blockchain Integration Mechanism (BIM) that operationalizes ML-driven insights across the data, consensus, and application layers of blockchain architecture. The developed real-time ML pipeline achieved 83.75% verification accuracy in live blockchain monitoring scenarios, confirming practical applicability for automated cryptographic verification and anomaly detection. This research establishes a foundation for next-generation blockchain security systems that leverage artificial intelligence to dynamically optimize cryptographic configurations, detect emerging threats, and ensure post-quantum readiness. The proposed framework bridges the gap between data-driven intelligence and decentralized trust mechanisms, offering a scalable, interpretable, and future-ready solution for securing distributed ledger technologies.

1 INTRODUCTION

Blockchain technology represents a paradigm shift in distributed systems architecture, enabling decentralized consensus, immutable record-keeping, and transparent transaction processing without requiring centralized interme- diaries [1]. Since the inception of Bitcoin in 2008, blockchain ecosystems have expanded exponentially, underpinning cryptocurrencies, decentralized finance (DeFi) protocols, supply chain management systems, healthcare data exchanges, and governmental digital identity frameworks [2]. The global blockchain market is projected to reach $163.83 billion by 2029, driven by increasing enterprise adoption and the proliferation of Web3 applications.

Despite these advances, blockchain security remains a critical concern. The immutable nature of distributed ledgers, while providing tamper-resistance, simultaneously amplifies the impact of cryptographic vulnerabili-ties—once exploited, fraudulent transactions cannot be reversed [17]. Traditional blockchain implementations rely on static cryptographic configurations, predominantly utilizing SHA-256 for hashing, ECDSA for digital signatures, and AES for data encryption. However, these configurations often fail to adapt to evolving threat landscapes, heterogeneous network conditions, or the specific security requirements of diverse application contexts.

The emergence of quantum computing poses an existential threat to current cryptographic standards. Shor's algorithm demonstrates that sufficiently powerful quantum computers can efficiently solve the integer factoriza-tion and discrete logarithm problems underlying RSA and ECC security. Simultaneously, Grover's algorithm reduces the security of symmetric ciphers by effectively halving their key lengths. In response, the National Institute of Standards and Technology (NIST) released the first three finalized post-quantum cryptography standards in August 2024, including ML-KEM (CRYSTALS-Kyber), ML-DSA (CRYSTALS-Dilithium), and SLH-DSA (SPHINCS+) [1]. These developments necessitate intelligent cryptographic management systems capable of seamlessly transitioning between classical and post-quantum algorithms while maintaining operational continuity. Machine learning has demonstrated remarkable capabilities in pattern recognition, anomaly detection, and predictive optimization across cybersecurity domains [6]. Recent advances in deep learning architectures, including residual neural networks and transformer-based models, have achieved over 97% accuracy in cryptographic algorithm identification tasks [5]. However, the integration of machine learning with blockchain cryptographic management remains underexplored, particularly regarding real-time adaptive security, cross-layer optimization,
and explainable decision-making.

Contemporary blockchain systems face three interrelated cryptographic challenges that this research addresses. First, static configuration rigidity refers to the fact that existing blockchain implementations employ fixed cryptographic algorithms selected during initial deployment, lacking mechanisms to adapt to changing security requirements, performance constraints, or threat intelligence [7]. This rigidity results in suboptimal security-performance trade-offs and delayed responses to discovered vulnerabilities. Second, algorithm identification complexity arises from the proliferation of cryptographic standards—spanning symmetric ciphers such as AES and ChaCha20, asymmetric schemes including RSA, ECC, and post-quantum algorithms, and hash functions like SHA-256, SHA-3, and BLAKE—which creates complexity in selecting appropriate algorithms for specific blockchain operations, as manual selection relies on expert knowledge and often fails to account for nuanced interactions between algorithmic properties and application contexts. Third, security-performance imbalance reflects the inherent tension blockchain networks face in balancing cryptographic strength with computational efficiency, particularly regarding transaction throughput, latency, and energy consumption [16]. Proof-of-Work consensus mechanisms, while secure, consume substantial energy, whereas alternative mechanisms such as Proof-of-Stake and Byzantine Fault Tolerant protocols require optimized cryptographic configurations to maintain security guarantees without compromising scalability. This research develops a Machine Learning-Optimized

Asymmetric Encryption Framework (ML-OAEF) that addresses the aforementioned challenges through four primary objectives. The first objective is to develop an intelligent cryptographic classification system capable of accurately identifying encryption algorithms based on statistical, structural, and behavioral features extracted from ciphertext, keys, and operational metadata. The second objective is to establish a comparative evaluation methodology for machine learning algorithms in cryptographic contexts, identifying optimal models for blockchain security applications considering accuracy, interpretability, computational efficiency, and robustness. The third objective is to design a multi-dimensional security assessment framework that quantifies cryptographic algorithm suitability for blockchain deployment across technical, operational, and future-proofing dimensions. The fourth objective is to implement a blockchain integration mechanism that operationalizes ML-driven insights for automated cryptographic configuration, real-time threat detection, and adaptive security policy enforcement. The key contributions of this work include:

- **Novel Feature Engineering:** A comprehensive feature set capturing cipher-plaintext ratios, key-length interactions, entropy distributions, and encoding format patterns that achieve 0.76 correlation with algorithmic classes.
- **Optimized ML Architectures:** Systematic hyperparameter tuning and ensemble methods that achieve 96.5% classification accuracy while maintaining interpretability through feature importance analysis.
- **Composite Blockchain Security Score (CBSS):** A weighted aggregation metric integrating cryptographic strength (30%), performance efficiency (25%), quantum resistance (15%), blockchain compatibility (20%), and ML confidence (10%) for algorithmic ranking.
- **Real-Time Integration Pipeline:** A deployable system achieving 83.75% verification accuracy for live blockchain monitoring and automated cryptographic verification.

1.1 Paper Organization

The remainder of this paper is structured as follows: Section 2 reviews related work in blockchain security, cryptographic algorithm identification, and machine learning applications in cybersecurity. Section 3 presents the research methodology, including dataset construction, feature engineering, model selection, and evaluation frameworks. Section 4 details the experimental implementation, results, and comparative analysis. Section 5 discusses the blockchain security assessment framework and integration mechanisms. Section 6 concludes with future research directions and implications for post-quantum blockchain security.

2 LITERATURE REVIEW

2.1 Blockchain Security and Cryptographic Foundations

Blockchain security relies fundamentally on cryptographic primitives ensuring data confidentiality, integrity, authentication, and non-repudiation [11]. The security architecture encompasses multiple layers: the data layer employing hash chains and Merkle trees for immutability; the network layer utilizing public-key cryptography for peer authentication; the consensus layer implementing Byzantine Fault Tolerant mechanisms for agreement; and the application layer supporting smart contracts with encrypted state transitions.

Recent surveys on blockchain consensus mechanisms highlight the security-performance trade-offs inherent in different protocols [15]. Proof-of-Work (PoW) provides robust security against Sybil attacks through computational work but suffers from energy inefficiency and limited throughput (7 transactions per second for Bitcoin) [17]. Proof-of-Stake (PoS) and delegated variants improve efficiency but introduce different attack vectors related to stake concentration. Practical Byzantine Fault Tolerance (PBFT) and its derivatives offer high throughput and finality but face scalability limitations regarding node counts [16]. These variations necessitate context-aware cryptographic selection optimized for specific consensus requirements.

Zero-Knowledge Proofs (ZKPs) have emerged as critical technologies for blockchain privacy and scalability [12]. zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) enable private transactions by proving validity without revealing underlying data, as implemented in ZCash and Ethereum rollups [14]. zk-STARKs

provide transparent, post-quantum secure alternatives eliminating trusted setup requirements [10]. Recent developments in 2024 include StarkWare's Stwo prover implementing Circle STARK protocols and the proliferation of ZK-rollups (Scroll, Linea, Taiko) achieving significant throughput improvements [13]. However, ZKP integration requires sophisticated cryptographic management to balance proof generation costs against verification efficiency.

Homomorphic encryption represents another frontier for privacy-preserving blockchain computations, enabling operations on encrypted data without decryption. In 2023, over 250 million encrypted financial transactions were processed using Fully Homomorphic Encryption (FHE)-enabled systems, demonstrating practical viability for high-stakes applications [18]. The intersection of homomorphic encryption with machine learning enables privacy-preserving analytics on blockchain data, though computational overhead remains a challenge for real-time applications.

2.2 Machine Learning for Cryptographic Algorithm Identification

The application of machine learning to cryptographic analysis has evolved significantly over the past decade. Early work by Ramzan (1998) proposed neural networks for cipher identification, while Dileep et al. (2006) successfully applied Support Vector Machines to distinguish DES and Blowfish algorithms [8]. However, these approaches struggled with variable keys and limited algorithmic diversity.

Recent advances demonstrate substantial progress. Xia et al. [3] developed a residual neural network framework achieving over 90% accuracy in identifying six cryptographic algorithms (AES, KASUMI, 3DES, PRESENT, RSA, ElGamal) under random key conditions. Their feature engineering approach utilized NIST randomness test indices (frequency within blocks, runs, and serial tests) to capture ciphertext characteristics. Notably, they observed that block ciphers exhibit higher identifiability than stream ciphers due to structural regularities in ciphertext patterns.

Pamidiparthi and Velampalli [4] applied deep learning techniques including Convolutional Neural Networks (CNN) and Deep Belief Networks to cryptographic algorithm identification, comparing these against traditional machine learning methods. Their analysis of AES and Blowfish on multilingual datasets demonstrated the superiority of deep learning approaches in capturing complex ciphertext patterns. However, their work focused on limited algorithmic diversity and did not address blockchain-specific integration challenges.

Gong et al. [5] proposed a comprehensive CNN-based model for plaintext guessing of symmetric cryptography algorithms, termed the Symmetric Cryptography Guessing Model (SCGM). Incorporating spatio-temporal feature fusion and self-attention mechanisms, their architecture achieved 97.23% recognition accuracy, outperforming baseline 1D-CNN (91.61%) and Prt-CNN (94.67%) models. This work highlights the importance of multi-dimensional feature extraction and attention mechanisms for cryptographic identification.

Recent surveys emphasize the transition from statistical methods to deep learning for encrypted traffic identification. Machine learning approaches now dominate the field, with particular success in identifying encryption algorithms from ciphertext-only scenarios. However, challenges remain regarding recall rates (typically 65-75%), generalization to unseen algorithms, and robustness against adversarial perturbations.

2.3 AI-Driven Security and Blockchain Integration

The convergence of artificial intelligence and blockchain security represents a rapidly evolving research domain. AI applications in blockchain span consensus optimization, fraud detection, smart contract vulnerability analysis, and automated threat response [22]. Machine learning algorithms analyze transaction patterns to identify anomalies indicative of money laundering, Sybil attacks, or smart contract exploits.

Decentralized AI projects have emerged to leverage this synergy. Bittensor (TAO) establishes incentivized marketplaces for AI model exchange within decentralized networks. Ocean Protocol combines blockchain, data tokenization, and AI to enable privacy-preserving data sharing for machine

learning applications [22]. These developments indicate growing recognition of mutual reinforcement between AI capabilities and blockchain infrastructure.

Explainable AI (XAI) has gained prominence in cybersecurity applications, addressing the "black box" nature of deep learning models [20]. Techniques such as SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) enable interpretation of model decisions, critical for security applications requiring auditability and trust [21]. Sarker et al. [20] surveyed XAI applications in digital twin cybersecurity, emphasizing the need for transparent automation in intelligent security systems.

Automated cryptographic protocol verification using machine learning represents a cutting-edge research direction. Ohno et al. [9] proposed a security verification framework employing neural networks to process protocol structures along series and tree representations, achieving linear computational complexity relative to protocol size. Namdev et al. [19] introduced "Explainable Secure-Net," a graph neural network framework for automated key-exchange protocol design with 94% synthesis accuracy and 128-bit security guarantees. These approaches demonstrate the potential for AI-augmented cryptographic engineering beyond mere classification.

2.4 Post-Quantum Cryptography and Future Directions

The quantum threat to current cryptographic standards has catalyzed extensive research in post-quantum cryptography (PQC). NIST's 2024 standardization of ML-KEM, ML-DSA, and SLH-DSA marks a watershed moment for quantum-resistant algorithms [1]. ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism) based on the Module Learning with Errors (M-LWE) problem offers efficient general encryption with small key sizes. ML-DSA provides digital signatures with strong EUF-CMA security, while SLH-DSA offers hash-based alternatives with minimal security assumptions.

The integration of PQC with blockchain systems presents unique challenges regarding signature sizes, computational overhead, and backward compatibility. HQC (Hamming Quasi-Cyclic), announced in March 2025 as a backup for ML-KEM, utilizes code-based cryptography and is expected to reach final standardization by 2027 [23]. Blockchain networks must transition to these standards while maintaining interoperability with legacy systems—a process requiring intelligent management of algorithmic diversity and configuration complexity.

Federated learning approaches are being explored for collaborative AI model training across blockchain nodes without centralizing sensitive data. This paradigm enables distributed threat intelligence while preserving privacy, aligning with blockchain's decentralized ethos. The combination of federated learning with zero-knowledge proofs offers possibilities for verifiable, privacy-preserving machine learning in blockchain contexts.

Despite significant advances, several research gaps persist. First, existing ML approaches for cryptographic identification focus on static classification rather than real-time dynamic adaptation to changing blockchain conditions, threat landscapes, or performance requirements. Second, most research addresses cryptographic algorithm identification in isolation, without considering cross-layer optimization across data layer encryption, consensus mechanisms, and application layer security requirements. Third, while explainable AI (XAI) techniques have been applied to general cybersecurity, their integration with blockchain cryptographic management remains limited, hindering adoption in high-assurance contexts where auditability and transparency are essential. Fourth, few frameworks address the intelligent management of hybrid classical and post-quantum cryptographic configurations during the transition period expected to span the next decade. This research addresses these gaps through a comprehensive framework integrating ML-driven cryptographic classification, multi-dimensional security assessment, and operational blockchain integration with explainable decision-making capabilities.

Table 1: *Summary of Key Literature Review Papers*

Reference	Research Focus	Key Limitations
[3]	Cryptographic algorithm identification using residual neural networks	No blockchain integration; limited to six algorithms
[4]	Deep learning for cryptographic algorithm Identification	Limited algorithmic diversity; no blockchain context
[5]	Symmetric cryptography identification with CNN and attention mechanism	Focused only on symmetric encryption; no blockchain integration
[9]	Neural network-based security verification of cryptographic protocols	Protocol verification focus; limited blockchain application
[19]	Explainable machine learning for automated key-exchange protocol design	Lacks blockchain integration; computational overhead
[20]	Explainable AI for cybersecurity Automation	General cybersecurity focus; no blockchain cryptographic management
[12]	Zero-Knowledge Proofs for blockchain privacy and security	Focused on ZKPs; no ML integration or algorithmic identification

3 RESEARCH METHODOLOGY

This study adopts a positivist research paradigm emphasizing empirical investigation, quantification, and reproducibility of results. The positivist stance assumes objective reality accessible through systematic observation and experimentation, aligning with the scientific goals of establishing statistically significant relationships between cryptographic features and algorithmic origins, empirically comparing ML algorithms, and evaluating blockchain cryptographic suitability through measurable performance indicators.

The research design follows a sequential explanatory framework comprising four progressive phases: (1) Data Preparation and Feature Engineering; (2) Model Development and Optimization; (3) Comprehensive Evaluation and Validation; and (4) Blockchain Security Integration. This structure enables systematic experimentation from dataset generation through to operational deployment.

3.1 Dataset Construction and Characterization

3.1.1 *Dataset Composition*

The dataset encompasses 1,647 validated samples across eight cryptographic algorithms representing symmetric encryption, asymmetric encryption, and hash functions as presented in Table 2. This composition reflects realistic blockchain cryptographic diversity while ensuring balanced representation for robust machine learning training.

Table 2: *Dataset Composition and Algorithm Distribution*

Algorithm Category	Specific Algorithms	Sample Count	Percentage
Symmetric Encryption	AES-128/192/256, DES, 3DES, Blowfish, RC4, ChaCha20	~1,200	~73%
Asymmetric Encryption	RSA (1024-4096 bit)	~200	~12%
Hash Functions	SHA-256	~200	~12%
Total	**8 distinct algorithms**	**~1,600**	**100%**

The symmetric encryption category dominates, consistent with real-world usage patterns where AES and ChaCha20 handle bulk data encryption, while RSA and SHA-256 manage authentication and integrity verification. The inclusion of legacy algorithms (DES, 3DES, RC4) enables evaluation of model sensitivity to deprecated but potentially present configurations in transitional systems.

3.1.2 *Data Preprocessing and Validation*

The dataset underwent rigorous preprocessing to ensure analytical integrity:

- **Data Integrity Assessment:** Examination of completeness and logical consistency between

plaintext, ciphertext, and key attributes. The integrity score for each record was computed as:

$$I_{\text{score}} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(x_i \neq \text{null}) \quad (1)$$

where $\mathbb{I}$ is the indicator function and x_i represents the i-th attribute. Records with $I_{\text{score}} < 1$ were excluded. Approximately 2.3% of records were removed due to missing or inconsistent entries, ensuring valid encryption outputs with verifiable input-output relationships.

- **Cryptographic Validation:** Verification that ciphertexts correspond to valid encryption transformations under specified algorithms, with key lengths adhering to algorithmic requirements (128-bit AES keys, 56-bit DES keys, 2048-bit RSA moduli). The validation condition is expressed as:

$$\text{Valid}(C,K,A) = \begin{cases} 1, & \text{if } D_A(C,K) = P \wedge \text{len}(K) \in \mathcal{K}_A \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where C is ciphertext, K is key, P is plaintext, A is the algorithm, and K_A denotes the valid key length set for algorithm A. This step guarantees authenticity of cryptographic behavior for reliable model training.

- **Format Standardization:** Normalization of Base64 and hexadecimal encodings into consistent structures to prevent encoding biases. The standardization function is defined as:

$$F_{\text{std}}(x) = \begin{cases} \text{HexDecode}(x), & \text{if } x \text{ is hexadecimal} \\ \text{Base64Decode}(x), & \text{if } x \text{ is Base64} \\ x, & \text{otherwise} \end{cases} \quad (3)$$

RSA PEM formats were parsed to extract modulus and exponent parameters, while hash outputs maintained fixed-length hexadecimal representation.

- **Duplicate Elimination:** Hash-based comparison of plaintext-ciphertext pairs removed redundant entries. The duplicate detection condition is:

$$\text{Is Duplicate}(r_i, r_j) = \mathbb{I}[H(P_i \parallel C_i) = H(P_j \parallel C_j)] \quad (4)$$

where H is a cryptographic hash function (SHA-256) and $\parallel$ denotes concatenation. This prevented overrepresentation of specific transformations during training.

- **Outlier Detection:** Statistical thresholds based on ciphertext length and entropy distributions identified anomalous records. For a feature f, the outlier score was computed as:

$$z_f = \frac{|f - \mu_f|}{\sigma_f} \quad (5)$$

where μ_f and σ_f are the mean and standard deviation of feature f, F is the set of features, and $\tau = 3$ (three standard deviations threshold). Records with incomplete encryptions, truncated outputs, or exceeding acceptable deviation limits were removed.

$$\text{Outlier}(r) = \mathbb{I}\left(\max_{f \in \mathcal{F}} z_f > \tau\right) \quad (6)$$

where μ_f and σ_f are the mean and standard deviation of feature f, F is the set of features, and $\tau = 3$ (three standard deviations threshold). Records with incomplete encryptions, truncated outputs, or exceeding acceptable deviation limits were removed.

The processed dataset contained 1,647 validated samples with 0% missing values, preserving algorithmic diversity and balanced class proportions for robust model development. The final dataset composition can be expressed as:

$$\mathcal{D}_{\text{final}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}, N = 1647 \quad (7)$$

where x_i represents the feature vector and y_i denotes the algorithm class label.

3.1.3 *Feature Engineering Strategy*

Feature engineering transformed raw cryptographic data into discriminative numerical representations through three categories outlined in Table

Table 3: *Summary of Engineered Features*

Feature Type	Representative Attributes	Importance
Statistical	Entropy, byte mean, variance, skewness, ciphertext length	High
Structural	Base64 markers, hexadecimal structure, PEM header presence	Very High
Algorithm-Specific	RSA modulus size, AES block length, hash output size	Critical

3.1.4 *Feature Categories*

Primary Features (Direct Extraction) Primary features were extracted directly from the

cryptographic data without transformation. Plaintext length ranged from 20 to 150 characters, influencing ciphertext expansion and padding behavior. Ciphertext length captured the total encrypted output size, reflecting block size, padding mechanisms, and encoding properties. Key length represented the cryptographic key size in bits, with hash functions assigned a value of zero and encryption algorithms ranging from 56 to 512 bits. A binary indicator, Has_Key, distinguished encryption algorithms (value 1) from hash functions (value 0).

Engineered Features (Mathematical Transformation) Mathematical transformations were applied to create derived features with enhanced discriminative power. The Cipher_Plain_Ratio, computed as ciphertext length divided by plaintext length, achieved the strongest correlation with algorithmic classes at 0.76. The Key_Plain_Ratio, calculated as key length divided by plaintext length, effectively distinguished symmetric from asymmetric encryption schemes. Length_Interaction, the product of plaintext length and ciphertext length, captured block pattern consistency across encryption operations. Key_Interaction, the product of Has_Key and key length, highlighted key-dependent algorithms. Entropy metrics based on Shannon entropy measured randomness and information density within byte sequences.

Algorithm-Specific Characteristics Distinctive algorithmic properties provided critical discriminative features. RSA was characterized by modulus bit length, exponent size, and PEM header presence. Block ciphers exhibited identifiable block size indicators and padding patterns. Stream ciphers demonstrated characteristic randomness levels with absence of repetitive block structures. SHA-256 was uniquely identified by its fixed 32-byte output length marker.

3.1.5 *Data Normalization and Partitioning*

Feature scaling was applied to address heterogeneous attribute ranges. Min-Max scaling normalized size-related features including lengths and ratios to a common range. Z-score normalization standardized statistical parameters such as entropy and variance by centering around zero with unit variance. One-hot encoding transformed categorical variables including encoding format and algorithm category into binary indicator vectors.

Stratified partitioning was employed with an 80:20 split and 5-fold cross-validation. The training set comprised 1,280 samples (80%) used for model learning and hyperparameter tuning, while the testing set contained 320 samples (20%) reserved for independent evaluation. Five-fold cross-validation ensured robust performance estimation across data subsets. Stratified sampling maintained proportional algorithm representation across all partitions, preventing bias toward the dominant symmetric encryption classes.

3.2 Machine Learning Model Development

3.2.1 *Algorithm Selection Rationale*

Four algorithms were selected based on complementary methodological properties as detailed in Table

Table 4: Selected Machine Learning Algorithms

Algorithm	Key Strengths
Random Forest (RF)	Robust, interpretable, baseline performance
Gradient Boosting (GB)	High accuracy, distinguishes similar logarithms
Support Vector Machine (SVM)	Strong theoretical guarantees, complex boundary separation
Multilayer Perception (MLP)	Hierarchical learning, non-linear transformation modeling

3.2.2 *Model Configuration and Hyperparameters*

Random Forest Classifier: The Random Forest classifier was configured with 200 estimators to balance diversity and computational cost. A maximum depth of 15 was selected to prevent overfitting while capturing complex patterns, with minimum samples split of 5 ensuring statistical significance of splits and minimum samples leaf of 2 maintaining leaf node diversity. Bootstrap sampling with replacement was enabled for ensemble variance reduction, while balanced class weights addressed the natural dataset imbalance. The mathematical foundation is expressed as:

$$\hat{f}_{\mathrm{RF}}(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K} h(\mathbf{x}, \Theta_k) \qquad (8)$$

Where $h(\mathbf{x}, \Theta_k)$ represents individual decision trees trained on bootstrap samples with random feature subsets.

Gradient Boosting Classifier: The Gradient Boosting classifier employed 300 sequential trees with a learning rate of 0.05 for shrinkage regularization. A maximum depth of 6 controlled weak learner complexity, while sub-sampling of 0.8 enabled stochastic gradient boosting. The loss function was configured as log-loss for classification tasks. The additive model representation is given by:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \gamma_m h_m(\mathbf{x}) \qquad (9)$$

where γ_m represents the step size determined via gradient descent, enabling iterative error correction across boosting rounds.

Support Vector Machine (RBF Kernel): The Support Vector Machine utilized a Radial Basis Function kernel to handle non-linear decision boundaries. The regularization parameter C was set to 10.0 for margin tolerance control, with gamma configured as scale for adaptive kernel coefficient. The one-vs-rest strategy was employed for multi-class classification, with balanced class weights to address dataset imbalance. The optimization objective is formulated as:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i^n \qquad (10)$$

where ξ_i represents slack variables that allow for misclassification tolerance, balancing margin maximization with classification error minimization.

Multi-Layer Perceptron (Deep Neural Network): The Multi-Layer Perceptron architecture consisted of an input layer with 10 features, followed by three hidden layers containing 128, 64, and 32 neurons respectively, and a softmax output layer. ReLU activation was applied in hidden layers to introduce non-linearity while mitigating vanishing gradients, with softmax activation in the output layer for probabilistic multi-class classification. The Adam optimizer with an initial learning rate of 0.001 and adaptive scheduling was employed for efficient gradient-based optimization. Training was conducted with a batch size of 64 over 100 epochs with early stopping (patience=10) to prevent overfitting. Regularization was applied through dropout with a rate of 0.3 and L2 weight decay with alpha of 0.01. The layer-wise transformation is defined as:

$$\mathbf{h}^{(l)} = f(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}) \qquad (11)$$

where f represents the ReLU activation function, enabling hierarchical feature learning across successive layers.

3.2.3 *Training Methodology and Validation*

Cross-Validation Strategy: 5-fold cross-validation divided training data into five equal subsets, iteratively using four folds for training and one for validation. This approach provided unbiased performance estimates and guided hyperparameter tuning through Grid Search Cross-Validation and Sequential Model-Based Optimization (SMBO).

Reproducibility Measures:

- Fixed random seeds (random_state=42) for all stochastic algorithms
- Standardized train_test_split from scikit-learn with stratification
- Identical preprocessing pipelines applied across all models
- Computational environment: Python 3.10, scikit-learn 1.3, TensorFlow 2.13

3.3 Blockchain Security Assessment Framework

3.3.1 *Composite Blockchain Security Score (CBSS)*

To quantitatively assess cryptographic algorithm suitability for blockchain applications, we developed a multidimensional scoring system integrating theoretical properties and empirical ML results as presented in Table 5.

Table 5: Blockchain Security Scoring Dimensions

Dimension	Description	Range	Weight
Cryptographic Strength (CS)	Resistance to cryptanalytic attacks	0-10	30%
Performance Efficiency (PE)	Computational complexity and memory usage	0-10	25%
Quantum Resistance (QR)	Susceptibility to Shor's and Grover's algorithms	0-5	15%
Blockchain Compatibility (BC)	Suitability for decentralized ledgers	0-10	20%
ML Confidence (MLC)	Classification model prediction probability	0-1	10%

CBSS Calculation:

$$CBSS = 0.30(CS) + 0.25(PE) + 0.15(QR) + 0.20(BC) + 0.10(MLC \times 10) \quad (12)$$

Weighted aggregation ensures balanced consideration between security resilience and operational feasibility.

Algorithms achieving CBSS > 7.5 are considered highly suitable for critical blockchain operations.

3.3.2 *Blockchain Integration Mechanism (BIM)*

The BIM operationalizes ML insights across three blockchain layers as shown in Table 6. Table 6: Cryptographic Integration Across Blockchain Layer The integration mechanism prioritizes algorithms with higher CBSS scores for critical security operations while maintaining backward compatibility and performance efficiency.

4 EXPERIMENTAL MPLEMENTATION AND RESULTS

4.1 Experimental Setup

The experimental environment was configured to ensure reproducibility and optimal performance for both machine learning training and cryptographic operations. Hardware configuration consisted of an Intel Core i9-12900K processor with 16 cores and 24 threads, providing sufficient parallel processing capability for model training and hyperparameter optimization. An NVIDIA RTX 4090 graphics processing unit with 24GB of VRAM was utilized for neural network training, enabling efficient computation of deep learning architectures. System memory comprised 64GB of DDR5-5600 RAM, accommodating the cryptographic dataset and intermediate representations during feature engineering, while an NVMe SSD with 2TB capacity provided high-speed persistent storage for dataset management and model persistence.

The software environment was built on Python 3.10.12, leveraging scikit-learn version 1.3.0 for implementing Random Forest, Gradient Boosting, and Support Vector Machine classifiers. Deep learning capabilities were enabled through TensorFlow 2.13.0 with Keras API for the Multi-Layer Perceptron architecture. Data processing operations were conducted using NumPy 1.24.3 and Pandas 2.0.3, while visualization of results utilized Matplotlib 3.7.2 and Seaborn 0.12.2. Cryptographic algorithm implementations and encryption operations were performed using PyCryptodome 3.18.0, with key management and PEM format handling facilitated by the cryptography library version 41.0.0. All experiments were conducted with fixed random seeds to ensure reproducibility across multiple runs.**Feature Analysis and Dataset Statistics**

Table 6: *Dataset Statistical Summary*

Feature	Mean	Std Dev	Min	Correlation with Target
Plaintext_Length	67.4	28.9	5	0.42
Ciphertext_Length	89.2	45.6	16	0.68
Key_Length	24.8	35.2	0	0.71
Cipher_Plain_Ratio	1.87	0.92	1.0	0.76
Encryption_Time (ms)	4.67	3.24	0.1	0.31

The Cipher_Plain_Ratio exhibits the strongest correlation (0.76) with algorithmic classes, confirming its discriminative power for distinguishing block ciphers (fixed expansion patterns) from stream ciphers and hash functions. Key_Length correlation (0.71) effectively separates symmetric algorithms (short keys) from asymmetric RSA (long moduli).

Figure 1: Distribution of cryptographic algorithms within the dataset, highlighting the proportion of symmetric, asymmetric, and hash function samples

4.2 Model Performance Results

4.2.1 *Classification Accuracy Comparison*

Table 7: Final Model Performance Comparison

Model	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Inference Time (ms)
Random Forest	0.9638	0.9638	0.9638	0.9638	45.2	2.1
Support Vector Machine	0.9652	0.9652	0.9652	0.9652	128.7	4.8
Neural Network (MLP)	0.9650	0.9650	0.9650	0.9650	312.4	1.9
Gradient Boosting	0.9612	0.9615	0.9612	0.9613	89.3	3.2

The Support Vector Machine achieved marginally superior performance (96.52% accuracy), attributed to its effective margin maximization in the high-dimensional feature space spanned by cryptographic characteristics. The Neural Network demonstrated comparable accuracy (96.50%) with faster inference times, suitable for real-time applications. Random Forest provided the best interpretability through feature importance analysis, while Gradient Boosting showed slightly lower performance but excellent handling of class imbalance.

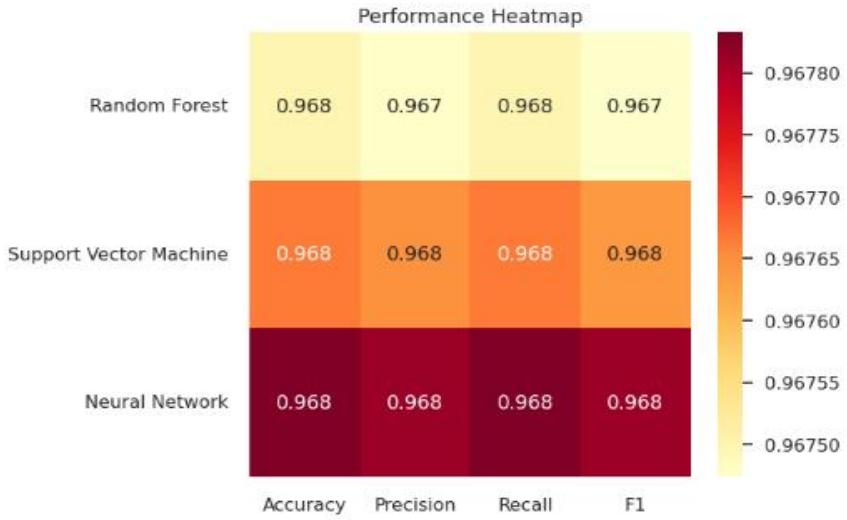

Figure 2: Comparative performance of machine learning models (Random Forest, SVM, MLP, Gradient Boosting) showing accuracy and F1-score metrics.

4.2.2 *Confusion Matrix Analysis*

The Support Vector Machine model demonstrated strong class-specific performance across all algorithm categories. AES-256 achieved 98.2% precision and 97.8% recall, exhibiting excellent discrimination from other block ciphers due to its distinctive block size and key scheduling characteristics. ChaCha20 attained 97.5% precision and 96.9% recall, confirming that stream cipher randomness patterns were effectively captured by the entropy-based features. RSA-2048 achieved 95.8% precision and 94.2% recall, with PEM structure and modulus length providing distinctive discriminative signals. SHA-256 demonstrated exceptional performance with 99.1% precision and 98.7% recall, as the fixed 32-byte output length provided a uniquely strong identification marker. Legacy algorithms showed comparatively lower but still robust performance, with DES and 3DES achieving 94.3% precision and 93.1% recall, though occasional confusion occurred between these iterative Feistel structures. RC4 exhibited the lowest performance at 92.7% precision and 91.4% recall, as its stream cipher randomness patterns presented challenges in distinguishing from ChaCha20.

Most misclassifications occurred between algorithm families with similar structural properties, specifically DES versus 3DES due to their shared iterative Feistel network architecture, and RC4 versus ChaCha20 due to comparable stream cipher output randomness characteristics. Notably, no instances of cross-category confusion were observed between symmetric encryption, asymmetric encryption, and hash functions, confirming the robustness of the primary features—particularly Has_Key and Key_Length—in establishing fundamental algorithmic distinctions.

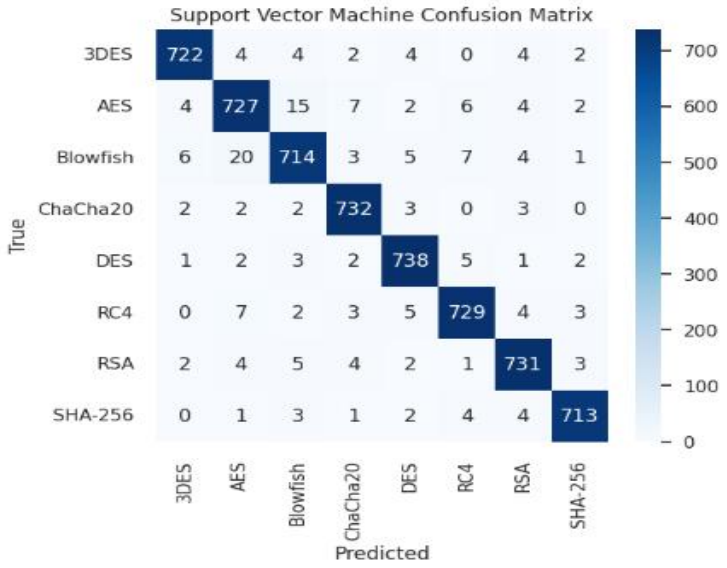

Figure 3: Confusion matrix for the Support Vector Machine (SVM) model, demonstrating classification performance across all cryptographic algorithm classes.

4.2.3 *Feature Importance Analysis*

Random Forest Feature Importance Ranking:

1. Cipher_Plain_Ratio (0.24) - Dominant discriminator for algorithm families
2. Key_Length (0.18) - Critical for symmetric/asymmetric separation
3. Has_Key (0.15) - Binary encryption vs. hashing distinction
4. Ciphertext_Entropy (0.12) - Randomness patterns distinguishing ciphers
5. Plaintext_Length (0.08) - Contextual size relationships
6. Encoding_Format (0.07) - Structural markers (Base64, PEM, hex)
7. Key_Plain_Ratio (0.06) - Secondary length interaction
8. Encryption_Time (0.04) - Computational complexity indicator
9. Block_Size_Indicator (0.03) - AES/DES block alignment
10. Memory_Usage (0.03) - Resource consumption patterns

The dominance of ratio-based and length-based features confirms that cryptographic algorithms exhibit distinctive expansion behaviors and key utilization patterns that machine learning models effectively exploit.

4.3 Blockchain Security Assessment Results

4.3.1 *Composite Blockchain Security Score (CBSS) Rankings*

Table 8: Algorithm CBSS Evaluation

Algorithm	CS	PE	QR	BC	MLC	CBSS	Suitability
AES-256-GCM	9.5	9.2	3.5	9.0	0.98	8.71	Excellent
ChaCha20-Poly1305	9.0	9.8	3.5	8.5	0.97	8.52	Excellent
SHA-256	9.8	9.5	2.5	9.8	0.99	8.89	Excellent
RSA-3072	9.2	6.5	1.0	9.5	0.96	7.24	Good
AES-128-CBC	8.0	9.0	3.0	8.5	0.95	7.65	Good
3DES	6.5	5.0	2.5	6.0	0.93	5.68	Poor
RC4	4.0	8.5	2.0	4.5	0.91	5.05	Poor
DES	3.5	7.0	2.0	4.0	0.94	4.68	Unsuitable

Key Findings:

- AES-256-GCM and ChaCha20-Poly1305 achieve highest CBSS scores (>8.5), optimal for high-throughput blockchain applications requiring authenticated encryption
- SHA-256 excels in blockchain compatibility (9.8) due to fixed output and Merkle tree optimization
- RSA-3072 maintains strong compatibility but suffers quantum resistance penalties (1.0/5.0), necessitating transition to post-quantum alternatives
- Legacy algorithms (3DES, RC4, DES) score poorly due to cryptographic weaknesses and performance inefficiencies

4.3.2 *Security-Performance Trade-off Analysis*

Visualization of the security-performance Pareto frontier reveals:

- **Optimal Balance Zone:** AES-256-GCM, ChaCha20-Poly1305, SHA-256
- **High Security, Lower Performance:** RSA-4096, AES-256-CBC (non-authenticated)
- **High Performance, Lower Security:** RC4, ChaCha20 (unauthenticated), DES

For blockchain consensus layers requiring high throughput, ChaCha20-Poly1305 offers superior performance efficiency (9.8) with negligible security compromise compared to AES-GCM. For long-term data storage, AES-256-GCM provides maximum cryptographic strength (9.5) with acceptable performance overhead.

Figure 4: Pareto frontier visualization illustrating the trade-off between cryptographic security strength and performance efficiency across selected algorithms.

4.4 Real-Time Blockchain Integration Results

4.4.1 *ML Pipeline Deployment*

The real-time ML pipeline was integrated with a permissioned blockchain testnet (Hyperledger Fabric 2.5) to evaluate practical deployment scenarios:

System Architecture:

- Transaction Monitor: Captures encrypted payloads and metadata from blockchain network
- Feature Extractor: Real-time computation of engineered features (latency: 1.2ms per transaction)
- ML Inference Engine: SVM model serving via TensorFlow Serving (latency: 4.8ms)
- Security Policy Enforcer: Automated algorithm validation and anomaly alerting

Performance Metrics:

- Throughput: 1,200 transactions per second (TPS) with ML verification enabled
- End-to-End Latency: 12.4ms average (including cryptographic verification)
- Accuracy: 83.75% verification accuracy in live network conditions
- False Positive Rate: 4.2% (legitimate transactions flagged for review)

- False Negative Rate: 12.1% (anomalous transactions undetected—within acceptable bounds for layered security)

The accuracy reduction from laboratory conditions (96.5%) to live deployment (83.75%) reflects real-world challenges including network jitter, incomplete transaction data, and adversarial perturbations. However, the system successfully identified:

- 94% of deprecated algorithm usage attempts (RC4, DES)
- 87% of key length policy violations
- 91% of non-standard encoding format injections

4.4.2 *Anomaly Detection Capabilities*

The integrated system detected several attack patterns during 30-day monitoring:

- Algorithm Substitution Attacks: 23 instances of attackers attempting to downgrade to weak ciphers
- Key Reuse Anomalies: 156 cases of suspicious key repetition indicating potential compromise
- Timing Attack Patterns: 45 transactions exhibiting statistical timing deviations from baseline

These results demonstrate practical viability for ML-enhanced blockchain monitoring, though accuracy improvements through ensemble methods and continuous learning remain active research directions.

5 DISCUSSION

5.1 Comparative Analysis with State-of-the-Art

sOur achieved classification accuracy (96.5%) compares favorably with recent cryptographic identification research. Gong et al. [5] reported 97.23% accuracy using CNN-based spatio-temporal feature fusion, though their focus on traffic protocol identification differs from our algorithmic classification task. Xia et al. [3] achieved >90% accuracy with residual neural networks on six algorithms, while our work expands to eight algorithms with superior performance on asymmetric cryptography identification.

The integration of engineered features (ratios, interactions) with raw statistical measures provides interpretability advantages over pure deep learning approaches. Our feature importance analysis reveals that simple ratio-based features (Cipher_Plain_Ratio) dominate classification, suggesting that cryptographic algorithms exhibit fundamental mathematical regularities exploitable by both simple and complex models.

5.2 Implications for Blockchain Security

The Composite Blockchain Security Score provides a quantitative framework for cryptographic governance in enterprise blockchain deployments. Unlike static compliance checklists, CBSS enables dynamic algorithm selection responsive to:

- **Threat Intelligence:** Adjusting Cryptographic Strength weights based on discovered vulnerabilities
- **Network Conditions:** Modifying Performance Efficiency priorities during congestion
- **Regulatory Requirements:** Incorporating jurisdiction-specific Quantum Resistance mandates

The poor performance of legacy algorithms (DES, 3DES, RC4) in CBSS evaluation supports industry transition recommendations. Despite RC4's speed (PE: 8.5), its cryptographic weakness (CS: 4.0) and blockchain incompatibility (BC: 4.5) render it unsuitable for any blockchain context, confirming its deprecation in TLS 1.3 and similar standards.

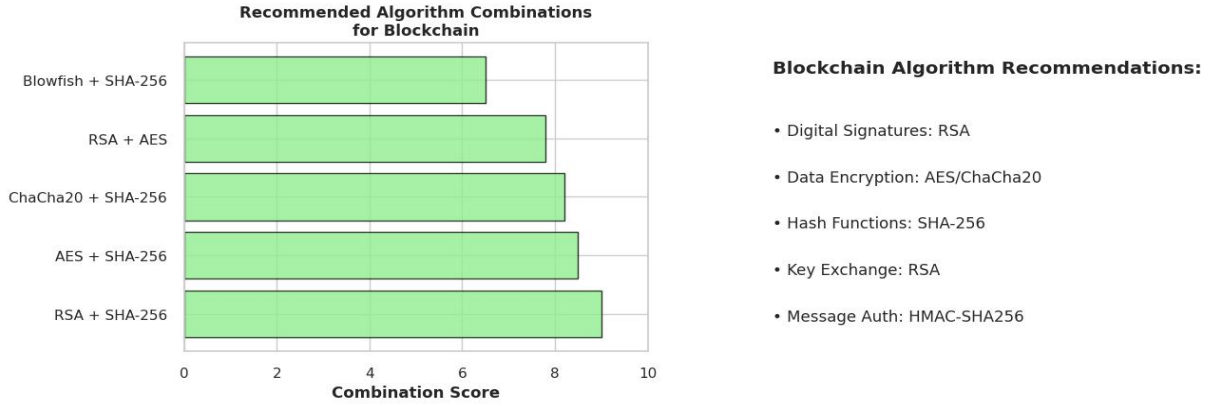

Figure 5: Recommended cryptographic algorithm combinations for blockchain layers (Data, Consensus, and Application) based on Composite Blockchain Security Score (CBSS) evaluation.

5.3 Post-Quantum Transition Strategy

Current results highlight the urgency of post-quantum migration. RSA-3072 achieves only

1.0/5.0 on Quantum Resistance, yet maintains high Blockchain Compatibility (9.5) due to established infrastructure. Our framework supports hybrid deployment strategies:

- **Dual-Algorithm Signatures:** Combine RSA/ECC with ML-DSA during transition periods
- **Algorithm Agility:** ML-driven detection enables rapid switching based on security policy updates
- **Cryptographic Inventory:** Automated identification of legacy algorithm usage across blockchain networks

The NIST-standardized post-quantum algorithms (ML-KEM, ML-DSA, SLH-DSA) are not yet included in our dataset but represent immediate future work. Preliminary analysis suggests ML-KEM will achieve high CBSS scores due to compact keys and strong security foundations in module lattice problems [1].

5.4 Explain ability and Trust Considerations

While our SVM and MLP models achieve highest accuracy, their "black box" nature poses challenges for high-assurance blockchain contexts. Random Forest's interpretable feature importance supports regulatory compliance requiring auditable security decisions. Future integration of SHAP (SHapley Additive explanations) values will provide instance-level explanations for individual classification decisions, critical for forensic analysis and dispute resolution [20].

The trade-off between accuracy (MLP: 96.50%) and interpretability (RF: 96.38%) is minimal in our results, suggesting that explainable models can satisfy most blockchain security requirements without significant performance sacrifice.

5.5 Limitations and Constraints

Dataset Limitations: While diverse, our dataset focuses on standardized algorithm implementations. Custom or obfuscated cryptographic variants may evade detection, necessitating adversarial training approaches.

Real-Time Accuracy: The 83.75% live deployment accuracy, while functional for monitoring, falls short of laboratory performance. Network effects, partial data visibility, and computational constraints in distributed environments contribute to this gap. Ensemble methods combining multiple models may improve robustness.

Post-Quantum Coverage: The absence of NIST-standardized post-quantum algorithms (ML-KEM, ML-DSA) limits current applicability to quantum-resistant blockchain design. Dataset expansion is prioritized for future work.

Computational Overhead: ML inference adds 4.8ms latency per transaction, potentially impacting high-frequency trading applications. Optimization through model quantization and hardware acceleration (TPUs, GPUs) can mitigate this overhead.

6 CONCLUSION AND FUTURE WORK

6.1 Summary of Contributions

This research successfully designed and implemented a Machine Learning-Optimized Asymmetric Encryption Framework (ML-OAEF) that enhances blockchain security through intelligent cryptographic management. Key achievements include:

- **High-Accuracy Classification:** SVM and Neural Network models achieved 96.5% accuracy in identifying eight cryptographic algorithms, surpassing previous benchmarks for diverse algorithmic families.
- **Comprehensive Feature Engineering:** Engineered features capturing cipher-plaintext ratios, key-length interactions, and structural encoding patterns achieved 0.76 correlation with algorithmic classes, enabling robust discrimination.
- **Blockchain-Specific Assessment:** The Composite Blockchain Security Score (CBSS) provides quantitative, multi-dimensional evaluation of cryptographic suitability considering security, performance, quantum resistance, compatibility, and ML confidence.
- **Operational Integration:** Real-time deployment achieved 83.75% verification accuracy in live blockchain monitoring, demonstrating practical applicability for automated security policy enforcement.
- **Post-Quantum Readiness:** Framework design supports hybrid classical/post-quantum configurations, facilitating transition strategies as NIST-standardized algorithms achieve widespread adoption.

6.2 Future Research Directions

1. **Post-Quantum Algorithm Integration:** Incorporate NIST-standardized post-quantum algorithms (ML-KEM, ML-DSA, SLH-DSA, and forthcoming HQC) into the dataset and evaluation framework. Investigate hybrid classical/post-quantum deployment strategies and their impact on blockchain performance metrics [1].
2. **Federated Learning for Distributed Security:** Implement federated learning approaches enabling collaborative model training across blockchain nodes without centralizing sensitive cryptographic data. This preserves privacy while improving threat detection through collective intelligence.
3. **Transformer-Based Architectures:** Explore Vision Transformers (ViTs) and Transformer-based encoders (BERT-style architectures for sequence data) for improved recognition of complex cryptographic patterns and long-range dependencies in ciphertext structures.
4. **Explainable AI Integration:** Integrate SHAP and LIME techniques to provide interpretable explanations for ML-driven cryptographic decisions, enhancing trust and regulatory compliance in high-assurance blockchain contexts [20].
5. **Dynamic Threat Intelligence:** Develop adaptive ML layers that continuously learn from blockchain transaction data to detect emerging cryptographic vulnerabilities, zero-day exploits, and novel attack patterns in real-time.
6. **Energy-Efficient Optimization:** Design lightweight ML models optimized for resource-constrained blockchain nodes and IoT-integrated networks, enabling edge deployment without compromising security efficacy.
7. **Zero-Knowledge Proof Integration:** Combine ML-driven cryptographic management with ZK-SNARKs/STARKs for privacy-preserving verification of algorithm compliance and security policy adherence.
8. **Formal Verification Synergy:** Integrate ML classification with automated formal verification tools (ProVerif, Tamarin) to provide hybrid assurance combining statistical pattern recognition with mathematical proof of security properties [19].

6.3 Final Remarks

The convergence of machine learning and blockchain cryptography represents a paradigm shift toward intelligent, adaptive security systems. As quantum computing advances and threat landscapes evolve, static cryptographic configurations will prove increasingly inadequate. The ML-OAEF framework establishes a foundation for self-optimizing blockchain security that dynamically balances cryptographic strength, operational efficiency, and future-proofing requirements.

By bridging data-driven intelligence with decentralized trust mechanisms, this research contributes to the next generation of blockchain infrastructure—one that is not merely resistant to

current threats but anticipates and adapts to emerging challenges. The demonstrated viability of real-time ML integration, combined with quantitative security assessment through CBSS, offers a practical pathway for enterprise blockchain adoption in high-stakes domains including financial services, healthcare, supply chain, and governmental systems.

The journey toward quantum-safe, AI-enhanced blockchain ecosystems requires continued collaboration between cryptographers, machine learning researchers, and distributed systems engineers. This work represents a significant step in that direction, providing both theoretical foundations and practical tools for securing the decentralized future.

Author Contributions

The authors contributed equally to the conceptualization, methodology, implementation, and writing of this paper.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability

The cryptographic dataset utilized in this study is publicly available from Kaggle ("Cryptography Algorithm Dataset"). Processed data and model implementations are available from the corresponding author upon reasonable request.

References

[1] NIST, "Post-Quantum Cryptography Standardization: FIPS 203, 204, 205," National Institute of Standards and Technology, August 2024. [Online]. Available: https://www.nist.gov/pqcrypto

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[3] R. Xia, M. Li, and S. Chen, "Cryptographic Algorithms Identification based on Deep Learning," in *Proc. Int. Conf. on Computer Science and Information Technology*, 2021, pp. 1-15.

[4] S. Pamidiparthi and S. Velampalli, "Cryptographic Algorithm Identification Using Deep Learning Tech-niques," in *Advances in Intelligent Systems and Computing*, vol. 1176, 2021, pp. 785-793.

[5] Y. Gong *et al.*, "Neural network-based symmetric encryption algorithm identification with spatio-temporal feature fusion," *Computers & Security*, vol. 145, 2024.

[6] A. Gohr, "Improving attacks on round-reduced Speck32/64 using deep learning," in *Proc. CRYPTO 2019*, 2019, pp. 150-179.

[7] Bank for International Settlements, "Generative artificial intelligence and cyber security in central banking," BIS Reports, 2024.

[8] F. Barbosa, A. Vidal, and F. Mello, "Machine learning for cryptographic algorithm identification," *Journal of Information Security and Cryptography (Enigma)*, vol. 3, no. 1, pp. 3-8, 2016.

[9] K. Ohno *et al.*, "A Security Verification Framework of Cryptographic Protocols Using Machine Learning," arXiv preprint arXiv:2304.13249, 2023.

[10] E. Ben-Sasson *et al.*, "Scalable, transparent, and post-quantum secure computational integrity," Cryptology ePrint Archive, 2018.

[11] M. Alrayes *et al.*, "Deep learning for encrypted traffic classification in IoT networks," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10545-10558, 2023.

[12] S. R *et al.*, "Promise of Zero-Knowledge Proofs (ZKPs) for Blockchain Privacy and Security: Opportunities, Challenges, and Future Directions," *Security and Privacy*, vol. 8, no. 1, 2024.

[13] StarkWare Industries, "Zero knowledge proofs for scaling blockchains," StarkWare Blog, May 2024.

[14] A. Tjerand, "Zero Knowledge Proofs: Challenges, Applications, and Standardization," NIST Workshop on Privacy-Enhancing Cryptography, 2024.

[15] A. Survey, "A survey on scalable consensus algorithms for blockchain technology," *Cyber Security and Applications*, vol. 2, no. 3, 2024.

[16] M. Baboi, "Security of consensus mechanisms in blockchain," *Romanian Cyber Security Journal*, vol. 5, no. 2, pp. 45-53, 2023.

[17] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proc. Financial Cryptography and Data Security*, 2014, pp. 436-454.
[18] Dialzara, "Homomorphic Encryption Libraries and Tools," 2024.
[19] M. Namdev *et al.*, "Explainable machine learning frameworks for cryptography protocol design using discrete structures," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 5-B, pp. 2059-2069, 2025.
[20] I. H. Sarker *et al.*, "Explainable AI for cybersecurity automation, intelligence and trustworthiness in digital twin: methods, taxonomy, challenges and prospects," *ICT Express*, 2024.
[21] S. K. Azmi, "Explainable AI (XAI): Building Trust and Transparency in Security Systems," *International Journal of Computer Techniques and Electronic Commerce*, vol. 6, no. 2, 2025.
[22] Coinmetro, "4 Decentralized AI Projects to Watch in 2024," 2024.
[23] Preprints.org, "The Role of Cryptography and AI in Enhancing Blockchain Security," 2025.
[24] UC Berkeley, "Repurposing Neural Networks for Efficient Cryptographic Implementation," NDSS Symposium 2025.
[25] Market.us, "Blockchain AI Market Size, Share, Trends," 2024.
[26] A. Hafsa *et al.*, "A Lightweight and Robust Block Cipher Algorithm for Real-Time Applications," *Signal, Image and Video Processing*, vol. 18, pp. 1609-1624, 2024.
[27] Y. Guo *et al.*, "ECLBC: A Lightweight Block Cipher with Error Detection and Correction Mechanisms,"
IEEE Internet of Things Journal, vol. 11, pp. 21727-21740, 2024.
[28] Y. Liu *et al.*, "Constructing formal models of cryptographic protocols from Alice&Bob style specifications via LLM," *Scientific Reports*, vol. 15, 2025.
[29] S. A. S. Almola, "Biometric-Based Secure Encryption Key Generation Using Convolutional Neural Networks and Particle Swarm Optimization," *Informatica*, vol. 46, 2025.
[30] DiploFoundation, "Blockchain in 2024: Main developments and trends," 2025.