

Article

FOX-SHIELD: A Secure Encrypted-Domain Remote Sensing Image Classification Framework Using Chebyshev-SHA and Fox-Optimized FRNNs

Abdullah Ghanim Jaber ^{1,2,*} , Ravie Chandren Muniyandi ³  and Khairul Akram Zainol Ariffin ^{1,*} 

¹ Centre for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi 43600, Selangor, Malaysia

² College of Business Informatics, Business Information Technology, University of Information Technology and Communications (UoITC), Baghdad 10067, Iraq

³ College of Computing and Informatics, Universiti Tenaga Nasional, Kajang 43000, Selangor, Malaysia; ravie.chandren@uniten.edu.my

* Correspondence: p131289@siswa.ukm.edu.my (A.G.J.); k.akram@ukm.edu.my (K.A.Z.A.)

Abstract

Remote sensing (RS) images transmitted through distributed satellite, UAV, and sensor networks are vulnerable to interception and misuse, while conventional encryption may reduce the effectiveness of downstream image classification. This paper proposes FOX-SHIELD, a secure encrypted-domain remote sensing image classification framework that integrates Chebyshev-SHA dynamic encryption with a Fox-Optimized Fast Recurrent Neural Network. The Chebyshev-SHA module generates image-specific dynamic keys and applies permutation–diffusion encryption to protect image confidentiality and integrity. The encrypted image representation is then classified using an FRNN optimized by the Fox Optimization Algorithm to improve convergence, classification accuracy, and computational efficiency. Experiments on the UC Merced Land Use and NWPU-RESISC45 datasets evaluate FOX-SHIELD across different scene-classification settings. On UC Merced, FOX-SHIELD achieves 97.14% accuracy, 96.87% recall, and 96.96% F1-score, outperforming selected privacy-preserving and lightweight remote sensing classification baselines. Empirical security analysis further indicates improved entropy, key sensitivity, and resistance to statistical attacks. The results suggest that FOX-SHIELD is a promising framework for secure remote sensing image classification in distributed, resource-aware environments.

Keywords: remote sensing; image classification; Chebyshev chaotic map; encryption; fox-optimized fast recurrent neural networks



Academic Editor: Kefeng Ji

Received: 22 August 2026

Revised: 16 September 2026

Accepted: 17 September 2026

Published: 23 September 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Satellite imagery, unmanned aerial vehicles (UAVs), and sensor networks are contributing to enhancing remote sensing [1]. Environmental monitoring, agriculture, disaster management, urban planning, and military surveillance are all greatly improved by high-resolution photography [2]. The cost of imaging equipment and the availability of commercial satellites have reduced the cost of acquiring large image libraries. Large datasets have significant advantages, such as improved generalization of the model and increased accuracy, due to their size and diversity. However, they also come with challenges, such as greater computational demands, longer training times, and issues with data storage and management [3]. Deep learning and cryptographic encryption methods for remote

sensing image classification process more data, but require high bandwidth for storage, transmission, and computation [4]. To gain meaningful insights from this enormous and complex imagery, it must be classified efficiently with low computational cost to support future data growth and protect sensitive information [5].

Remote sensing images for highly critical military, infrastructure, and disaster response applications require enhanced cybersecurity [6]. The computational cost of encryption makes image categorization difficult, and image classification algorithms must be robust enough to handle encrypted data that may obscure key photographic features [7]. In real-time applications, maintaining strong encryption while achieving high classification performance is challenging, as even a few milliseconds of delay can degrade overall system performance [8]. The information and risk environment continues to grow. Therefore, a framework is needed to protect, analyze, and classify remote sensing images in a shorter timeframe without compromising accuracy and scalability [9].

General encryption techniques like AES (Advanced Encryption Standard) or RSA (Rivest–Shamir–Adleman) can be used to secure data, but may not be appropriate for the properties of image data. These approaches can also result in loss of the visual features of images (such as edges, textures, fine details) after encryption, thus interfering with image classification tasks [10]. Images with high resolution can be very demanding in terms of processing power and may not be appropriate for near real-time remote sensing applications [11]. Distributed schemes are at higher risk of threats, as distributing images over several nodes introduces an additional threat of interception [12]. The disadvantages of static functions and low-entropy key protection justify the use of chaotic and dynamic systems for the generation of keys for encryption functions with lower computational cost and less predictable keys [13]. Low-entropy keys are not random enough to protect data from sophisticated cryptanalysis. This enables cryptographic operations using chaotic and dynamic systems that produce keys that are much less predictable and adjust to varying conditions, offering greater security without the need for high computational costs. Such an encryption would be very helpful for the security and confidentiality of controlled and sensitive operations.

While some existing works have explored the fusion of encryption and deep learning for remote sensing image classification, there are still some issues to be solved, such as performance degradation, slow convergence, and high computational cost for the fusion of encryption and deep learning on encrypted images. Further research is needed in the field of lightweight encryption schemes and optimized models that can achieve good classification performance while maintaining security [14]. Deep learning and encryption provide great security, but also pose new challenges such as feature-space distortion, slower convergence of training, and increased computational costs, which make traditional deep learning methods with encrypted data less effective [15]. Furthermore, most deep learning models are not designed to be resource-efficient, which is crucial for applications with limited resources, such as satellite or UAV processing in the field, where computational and memory resources are limited [16]. The lack of integration of encryption-aware preprocessing methods and classification workflows also negatively affects performance. Furthermore, the parameters of neural networks used for encrypted data classification are not often optimized using metaheuristic search algorithms, which leads to a decrease in the performance of the neural network [17]. Specialized architectures can be developed to overcome these inefficiencies while maintaining high processing performance and accuracy for encrypted data processing [18].

The use of remote sensing (RS) imagery from UAVs and satellites is an important source of geographic information for various applications, including infrastructure monitoring, environmental monitoring, and military observation. But the classification of remote

sensing (RS) images has its challenges, including large intra-class variation, intra-class similarity, variation in object size, and overlapping objects on the ground [19]. High dimensionality, limited labeled training data, and the high cost of processing high-resolution RS images are challenges. Additionally, RS data could be intercepted or manipulated during transmission or storage in a wireless sensor network. Although modern encryption systems provide security [20], they require significant resources and rely on public keys, which can pose security risks in remote sensing. In addition, while deep learning models are robust and powerful, they lack interpretability, are vulnerable to adversarial perturbations, and require extensive parameter tuning. The following issues may challenge the reliability of their outputs: encryption, contextualization, and misclassification due to noise. We need to design a more secure, robust, interpretable, and intelligent classification framework that leverages large volumes of RS data, mitigates secure-processing risks, and withstands related attacks [21].

There is a growing need for accurate and secure classification technologies in remote sensing (RS); environmental monitoring, disaster management, land administration, and military applications all utilize remote sensing imagery. When using remote sensor networks, it is also natural to want to intercept and modify important RS data [22]. While convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can improve the classification process, RNNs often encounter challenges when applied to encrypted remote sensing (RS) datasets because of the lack of built-in encryption support or adaptation to the encrypted domain, which results in inefficient performance and creates vulnerability in RS. Existing studies show that hyperspectral datasets involving remote sensing images captured across many narrow spectral bands (from visible to infrared), or large RS datasets, require substantial computing power, long training times, and multiple rounds of optimization to mitigate the negative effects of class imbalance and spatial heterogeneity.

1.1. Problem Statement of the Research

Remote sensing imagery may contain sensitive geographic, infrastructure, and defense information that must remain protected during transmission, storage, and outsourced analysis. Conventional encryption provides strong confidentiality but produces ciphertext that standard image classifiers cannot process directly, while structure-preserving transformations may retain classification utility at the cost of privacy leakage. Existing studies have not established an efficient, reproducible framework that jointly provides authenticated image protection, classification from protected representations, and acceptable accuracy and computational overhead. Therefore, this study investigates whether a dynamically keyed permutation–diffusion representation combined with an optimized recurrent classifier can provide a measurable balance between security, classification utility, and computational efficiency.

1.2. Objectives of the Research

RO1: To develop an authenticated, dynamically keyed permutation–diffusion mechanism for protecting remote sensing images while producing representations suitable for machine classification.

RO2: To develop an encryption-aware Fast Recurrent Neural Network and optimize its architecture and training hyperparameters using the Fox Optimization Algorithm.

RO3: To evaluate the privacy–utility–efficiency trade-off of the proposed framework using classification performance, cryptographic robustness, convergence behavior, computational overhead, and ablation experiments.

The main contributions of this work are:

- FOX-SHIELD is an integrated privacy-preserving classification framework using dynamically keyed Chebyshev-SHA image transformation, encrypted-block se-

quential representation, FRNN classification, and FOA-based optimization in one unified pipeline.

- Encryption-aware sequential classification: a framework that uses sequential FRNN processing on blocks of encrypted images to retain useful spatial-contextual information for classification without having to access the original image content.
- Optimization-assisted performance improvement: FOA validates FRNN hyperparameters using a fitness function, then uses the best configuration to train and test the model.
- Component-wise empirical validation—ablation experiments: conducted by comparing different variants of encryption techniques, classifier architectures, optimization approaches, and key-generation variants to quantify each technique’s contribution to overall performance.
- Joint privacy–utility–efficiency assessment: a framework that jointly considers classification effectiveness, security properties, and computation cost to give an empirical evaluation of the encrypted-domain classification trade-off.

Remote sensing images captured by UAVs and satellites are essential geographic information assets for monitoring infrastructure, assessing environmental conditions, and conducting military reconnaissance. Nevertheless, remote sensing image classification struggles with issues such as considerable intra-class variation, similarities within classes, variations in item size, and overlapping objects on the ground. The constraints include high dimensionality, a lack of labeled training data, and the significant costs associated with processing high-resolution remote sensing images. These operational environments inspire secure categorization systems such as FOX-SHIELD. This study examines the framework on UC Merced and NWPU-RESISC45 in unblemished and deteriorated states. Statements concerning deployment in distributed sensor networks, UAVs, or satellite platforms are thus articulated as motivation instead of established capability. Section 4.1 presents the results pertaining to cross-dataset and robustness.

2. Related Works

2.1. Deep Learning-Based Remote Sensing Image Classification

Song, H. (2023) [23] proposed a fast, simple CNN training framework (FST-EfficientNet), a one-stage deep learning CNN for semantic scene categorization of remotely sensed images using EfficientNetV2-S. The authors employ a systematic data-augmentation approach using constant or increasing resolution to improve training effectiveness. The results achieve state-of-the-art accuracy improvements (0.8–2.7%) on the Aerial Image Dataset (AID) and the Northwestern Polytechnical University Remote Sensing Image Scene Classification 45 Dataset (NWPU-RESISC45D). Limitations include likely inadequate performance on highly intricate datasets that lack sufficient augmentation diversity.

Similarly, Albarakati et al. (2024) [24] proposed a framework that incorporates information into Land-Use and Land-Cover (LULC) classification by designing two novel CNN (convolutional neural network) models called the Residual Self-Attention Network with six residual blocks (ResSAN6) and the Remote Sensing Inverted Residual Self-Attention Network with six inverted residual blocks (RS-IRSAN), which were built from the ground up using the Bayesian Optimization Algorithm (BOA). The frameworks included Data Augmentation (DA) to mitigate class imbalance, Mutual Information-based Serial Feature Fusion (MI-SFF), Median Normalization (MN), and, finally, Arithmetic Optimization (AO) for feature selection, followed by classification with a Shallow Wide Neural Network (SWNN). The trials using RSI-CB128 achieved accuracies of 95.7%, 97.5%, and 92.0% on the WHU-RS19 and NWPU-RESISC45 datasets, respectively. Limitations include longer processing times and greater model complexity when applied to large datasets in real time.

Ahmed et al. (2024) [25] introduced XcelNet17, a deep learning framework with fourteen convolutional layers and three fully connected layers for remote sensing image classification. The authors also introduced a hybrid feature selection algorithm, BA-ABC, that combines the Bat Algorithm (BA) with the Artificial Bee Colony (ABC). XcelNet17 achieves 94.6% to 99.9% on the WHU-RS19 test dataset, surpassing AlexNet, VGG16/19, ResNet50, and DarkNet19. The BA-ABC feature selection method was compared with the Whale Optimization Algorithm (WOA), Grey Wolf Optimizer (GWO), Bat Algorithm (BA), Artificial Bee Colony (ABC), and Ant Colony Optimization (ACO). The results showed that BA-ABC increased accuracy by up to 8%. Constraints introduce additional complexity to model training and pose greater challenges for scaling to large RS datasets.

Liu et al. (2025) [26] proposed a lightweight convolutional network for remote sensing (RS) image classification called STConvNeXt, featuring a split-based mobile block convolution module with a hierarchical tree structure and parameterized depthwise separable convolutions to reduce complexity. A fast pyramid pooling module provides a large context, and a dynamic threshold loss enhances class separability. Using AID as the primary dataset, STConvNeXt reduces parameters (56.49%) and FLOPs (49.89%) relative to ConvNeXt, with a 1.2–2.7% accuracy improvement. Possible limitations include poorer performance on extremely high-resolution remote sensing data or multimodal remote sensing data.

Song et al. (2024) [27] introduced a differentiable Neural Architecture Search (NAS) for RS image classification that uses a binary gate to establish partial channel connections, thereby reducing parameter and memory usage. The results show a 15.1% increase in validation accuracy during search, while reducing search time by 88% and parameters by 84% compared to DARTS. Limitations included a 4.5% reduction in accuracy when compared to DARTS, and possibly reduced tuning capabilities across many RS datasets.

Rasheed et al. (2021) [28] proposed a trustworthy RS-embedded approach for classifying high-resolution satellite images while accounting for photometric and geometric distortions. Features generated from the last fully connected layer of a deep neural network (DNN) were classified using a multiclass support vector machine (SVM) with a Gaussian kernel. This approach achieved 93.8% accuracy using data from China's Headwater Region, a significant improvement over contemporary methods. The only limitations were manual kernel parameter tuning and scalability with data from instances that use many different RS datasets.

2.2. Privacy-Preserving and Encrypted Remote Sensing Image Processing

To reduce reconstruction time for high-resolution remote sensing images in the Internet of Things, Zhang et al. [29] proposed a lightweight Visual Cryptography-based privacy method. This work used block-based encryption with error diffusion, Boolean-based decryption, and custom CNN training. They increased recognition accuracy without compromising privacy. Potential processing overhead in a large-scale deployment and the risk of perceptual distortion are among its limitations.

Alkhelaiwi et al. (2021) [30] reported an automated satellite-imagery system based on a partially homomorphic Paillier encryption scheme that allows training a CNN on encrypted data while preserving privacy. They used a custom CNN architecture, along with lightning and transfer learning. They have reported powerful privacy guarantees while preserving CNN accuracy. Limitations included increased computational overhead for encryption and real-world challenges scaling their private solutions to large datasets derived from satellite imagery.

2.3. Chaotic Map-Based Image Encryption

Al-Khasawneh et al. [31] proposed a chaos-based parallel image encryption algorithm for remotely sensed images, employing Henon, Logistic, and Gauss maps and an external secret key. It was executed on Hadoop with enhanced file management for TIFF/GeoTIFF for scalable, efficient encryption of large datasets. Results are more effective than current methodologies. Limitations include lower efficiency when image dimensions are reduced significantly when working with large batches.

2.4. Encrypted-Domain Machine Learning/Classification

Zhang et al. (2022) [32] presented a lightweight privacy-preserving framework for remote sensing images in the Internet of Things (IoT) based on visual cryptography with a stacking-to-see property. Approaches include adding cargo information to the encryption process using blocks and denoising neural networks to improve quality and recognition accuracy. Experimental results show strong privacy protection and effective classification. Limitations include potential performance degradation in high-noise environments and the computational requirements of large-scale, real-time applications.

Hou et al. (2025) [33] proposed a fast, secure cloud-based configuration for retrieving remote sensing images, with verifiability and traceability. Approaches include retrieval methods using feature extraction from convolutional neural networks, dimensionality reduction using spectral hashing with spectral rotation (SHSR), an encrypted searchable index using asymmetric scalar product preserving encryption (ASPE), pixel rearrangement for predictive error (PE) labeling in image encryption, watermark embedding, and Merkle tree verification. Results show greater retrieval accuracy and stronger privacy protection. However, these methods increase computational complexity and storage overhead in large deployments.

Feng et al. [34] proposed the Real-Time Detection Transformer (RTDETR). They integrated it with Soft-threshold and Cascaded-Group-Attention (SC-RTDETR), a multi-source forest remote sensing data fusion system based on the Real-Time Detection Transformer, Soft-threshold adaptive filtering, and Cascaded Group Attention. These modules reduce attack-induced noise and minimize focus on less relevant details, resulting in more robust detection. Experiments on pine wilt disease datasets reported a 12.9% gain in mean Average Precision (mAP) after attack. However, potential disadvantages include increased complexity and increased computational effort, which could be drawbacks in large-scale, real-time implementations.

2.5. Research Gaps and Motivation

Existing studies have investigated remote sensing image classification, privacy-preserving encryption, encrypted-domain processing, and metaheuristic optimization independently. However, few studies have integrated hybrid chaotic-map encryption, encrypted-domain scene classification, and optimization-assisted deep learning within a unified framework. Existing privacy-preserving methods primarily focus on secure image transmission or encrypted retrieval, whereas advanced classification models generally operate on plaintext images. Furthermore, researchers have rarely explored optimization strategies for encrypted-domain recurrent neural networks. To address these limitations, this paper proposes the FOX-SHIELD framework, which combines Chebyshev–SHA hybrid encryption, encrypted-domain scene classification using an FRNN [35], and Fox Optimization-based hyperparameter tuning within a single end-to-end architecture. Table 1 below summarizes the reviewed papers on RS image processing methods.

Table 1. Summary of recent RS image processing methods, highlighting main contributions, applied techniques, achieved results, and identified limitations in privacy preservation, classification accuracy, and computational efficiency.

Ref. & Authors	Main Contribution	Methods	Results	Limitations
Song, H. (2023) [23]	Fast, simple CNN for RS scene categorization	FST-EfficientNet (EfficientNetV2-S), constant/progressive resolution augmentation	SOTA accuracy gain (0.8–2.7%)	May underperform on very complex datasets
Albarakati et al. (2024) [24]	LULC classification with dual CNNs + fusion	ResSAN6, RS-IRSAN, DA, MI-SFF, MN, AO, SWNN	Acc: 95.7%, 97.5%, 92.0%	High computation, complex model
Ahmed et al. (2024) [25]	XcelNet17 + BA-ABC hybrid feature selection	14 conv + 3 FC layers, BA-ABC	Acc: 94.6–99.9%, +8% over baselines	Complex training, scaling issues
Liu et al. (2025) [26]	Lightweight RS classification CNN	STConvNeXt, depthwise separable conv, fast pyramid pooling, dynamic threshold loss	Params ↓56.49%, FLOPs ↓49.89%, acc ↑1.2–2.7%	May struggle on high-res/multi-modal data
Song et al. (2024) [27]	Efficient NAS for RS classification	Differentiable NAS, binary gate, partial channel connection	Acc ↑15.1% vs. DDSAS, time ↓88%, params ↓84% vs. DARTS	Slightly lower acc than DARTS, tuning difficulty
Rasheed et al. (2021) [28]	Robust RS classification under geometric/photometric variation	DNN features + multiclass SVM (Gaussian kernel)	Acc: 93.8%	Manual kernel tuning, scalability issues
Zhang et al. (2023) [29]	Lightweight privacy-preserving detection in IoT RS images	Visual Cryptography (VC), block-based encryption, error diffusion, Boolean decryption, optimized DL models	High recognition accuracy + privacy	Processing overhead, visual distortions
Alkhelaiwi et al. (2021) [30]	Privacy-preserving CNN training on encrypted satellite images	Paillier encryption, bespoke CNN, transfer learning	Maintains accuracy with strong privacy	High computing cost, scalability issues
Al-Khasawneh et al. (2022) [31]	Chaos-based parallel RS image encryption	Hénon, Logistic, Gauss maps, Hadoop parallel processing	Higher encryption efficiency	Lower efficiency for very small images
Zhang et al. (2022) [32]	Lightweight VC framework with stacking-to-see	Block-based encryption, denoising NN	Strong privacy + recognition	Performance drop with high noise, real-time cost
Hou et al. (2025) [33]	Secure cloud-based RS image retrieval with traceability	CNN features, SHSR, ASPE, pixel rearrangement, watermark, Merkle tree	High retrieval precision + privacy	High computation and storage overhead
Feng et al. (2024) [34]	Adversarially robust forest RS detection	SC-RTDETR, Soft-threshold filtering, Cascaded-Group-Attention	mAP ↑ up to 12.9% under attack	High complexity, computational cost
Jaber & Muniyandi (2021) [36]	Hybrid deep learning-based face recognition	Gabor filters, stacked sparse autoencoders (SSAE), deep neural networks	Improved recognition accuracy and feature extraction efficiency	High training cost, sensitive to illumination
Rahman et al. (2020) [37]	Review of ML-based feature selection and classification for ASD	Feature selection, ML classifiers, comparative evaluation	Identified effective FS and classification strategies	Not directly applied to RS images; lacks privacy scope
Sihwail et al. (2021) [38]	Memory-based malware detection and classification	Feature engineering, DNN, memory forensics	High detection accuracy and robustness	Limited to the malware domain, not optimized for RS data
Hasan et al. (2021) [39]	Lightweight encryption for medical image security	Stream cipher, key generation, hash-based encryption	High encryption speed, enhanced IoMT privacy	Limited to medical images, scalability concerns
Talukdar et al. (2021) [40]	Secure network communication via IDS and digital signature	IDS integration, AODV optimization, digital signature	Improved packet delivery and attack detection	Designed for ad hoc networks, limited RS applicability

Note: ↑ = increase; ↓ = decrease.

A detailed analysis of the literature on secure and efficient remote sensing image classification reveals several crucial gaps. Although a variety of research has been conducted on privacy-preserving techniques including visual cryptography, chaos-based encryption, and homomorphic encryption, the combination of strong encryption and deep learning algorithms targeted at encrypted input data is obviously underrepresented, and it leads to either reduced classification performance, restrictive computational capabilities, or both [26,28]. Second, many modern encryption schemes use static or predictable keys, which defeats the security purpose of encryption; they also lack a dynamic, high-entropy backdoor to be effective against successful cryptanalytic attacks. Third, many privacy-preserving schemes impose high computational and storage burdens. They cannot be replicated or run in real time or on resource-constrained platforms, such as UAS and onboard satellites.

Fourth, deep learning algorithms designed for unencrypted remote sensing (RS) images often show lower accuracy on encrypted datasets in the model's output space, yet lack an appropriate procedure to address this [23,24]. Fifth, metaheuristic optimization is rarely used to tune classification network parameters in the encrypted space, which could improve convergence and resilience. Additionally, some approaches use either contextual modeling or region-specific features, which are essential in complex RS environments. Lastly, most studies are evaluated under stable conditions, with little testing in dynamic, real-time operational environments, leaving a large gap in research on robustness and implementation feasibility.

3. Proposed Methodology

The proposed FOX-SHIELD framework provides a secure and effective approach to remote sensing image classification by integrating an improved Chebyshev chaotic map [41], dynamic key generation using the Secure Hash Algorithm (SHA) [42], and the Fox-Optimized Fast Recurrent Neural Network (FRNN). Classification leverages spatial-temporal relations in encrypted imagery using FRNN. The Fox Optimization Algorithm (FOA) [43] is used to tailor network parameters concurrently to improve convergence speed, stability, as well as resistance against noise and attacks. Figure 1 shows the overall proposed system architecture.

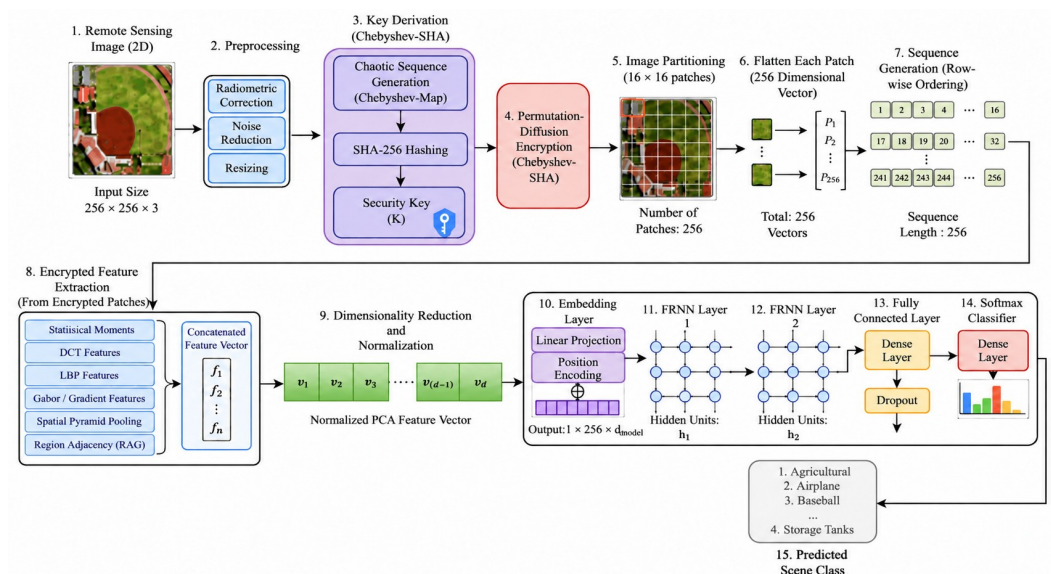


Figure 1. Proposed FOX-SHIELD framework.

The implemented FOX-SHIELD classifier does not use the flattened encrypted RGB patches directly as the final FRNN input. Instead, the system divides the encrypted

256 × 256 RGB image into non-overlapping 16 × 16 blocks and extracts statistical, spectral, texture, and contextual descriptors using knowledge of the encryption. These descriptors include statistical moments, intensity histograms, DCT energy, LBP histograms, gradient/Gabor responses, spatial pyramid pooling descriptors, and region-adjacency features. The resulting descriptors are then combined and concatenated into a single image feature vector, then dimensionality-reduced and normalized using PCA. The PCA-reduced feature vector is the input to the Fox Optimized FRNN. The classification procedure is as follows: preprocessed RGB image, Chebyshev–SHA encryption, 16 × 16 encrypted image blocks, feature extraction using the encryption-aware method, feature concatenation, PCA and normalization, FRNN, hyperparameter optimization using the FOA method, and class prediction.

Figure 1 shows the proposed framework for remote sensing image classification using Fast Recurrent Neural Networks. This framework converts two-dimensional images into a sequence for scene categorization. The preprocessed RGB image (256 × 256) is encrypted using the Chebyshev–SHA mechanism and then divided into 16 × 16 encrypted blocks. These blocks are used to extract the encryption-aware descriptors, such as statistical, spectral, texture, multiscale, and graph-context descriptors. The extracted descriptors are then concatenated, reduced by PCA, and normalized. The resulting feature sequence is fed to the FRNN for classification in the encrypted domain, and FOA tunes the FRNN parameters. The classification network comprises two stacked FRNN layers with numerous hidden units that learn complex sequential relationships between neighboring image patches. Unlike convolutional methods, which capture only local spatial properties, the FRNN progressively captures short- and long-range contextual dependencies across the image sequence. Dual FRNN layers enhance these representations by learning low-level sequential characteristics and higher-level semantic information relevant to distinct land-use categories.

A fully connected layer with a ReLU activation function receives the FRNN's high-level features to enhance feature discrimination and non-linearity. Dropout layers then improve model generalizability and reduce overfitting. This layer randomly deactivates neurons during training. Finally, a Softmax classifier is trained on the studied properties. The classifier then assigns the image to the scene category with the highest probability.

The dataset includes many land-use categories; the framework's final output is the predicted remote sensing scene class. These classes include agricultural, airplane, baseball diamond, and storage tank. The proposed pipeline is based on patch-based image representation, sequential feature modeling, and deep recurrent learning to capture local and global context for fast and effective remote sensing image classification.

The proposed FOX-SHIELD (Fox-Optimized Secure Hybrid Image Encryption and Optimization-based Detection) framework integrates advanced encryption, feature-learning, and optimization modules to enable secure and accurate remote sensing image classification. The overall architecture comprises five main stages: data acquisition, pre-processing, hybrid encryption, feature extraction and classification, and optimization-assisted model training. This process starts with data collection using the UC Merced Land Use dataset, which includes 21 land use categories representing a variety of urban and peri-urban environments [44]. After that, an image pre-processing module is applied to guarantee uniformity and data quality by integrity verification, image resizing, denoising, photometric normalization, and data augmentation. These steps normalize the data and increase the intra-class variance, improving the model's generalization. The Hybrid Encryption Module then processes the images with a Chebyshev Chaotic Map and a key derivation function using the HKDF-SHA-256 algorithm for secure encryption. This hybrid approach provides strong data confidentiality/resistance to cryptographic attacks, along

with structural integrity for feature extraction in the encrypted domain. The resulting images are secure from unauthorized viewers and do not compromise the analyst's usability. The Feature Extraction Module works directly in the encrypted domain and extracts a discriminative representation from the encrypted information by employing a Fast Recurrent Neural Network (FRNN). The FRNN-based classification module captures both spatial and temporal dependencies, facilitating accurate classification of encrypted remote sensing images. To ensure optimal performance and stable convergence, the parameters of the FRNN, such as the learning rate, the number of recurrent units, and the regularization coefficients, are optimized using a Fox Optimization Algorithm (FOA) Hyperparameter Optimization Loop. This metaheuristic optimization mechanism enhances the classification accuracy and computational efficiency. Finally, full FRNN training with early stopping is performed to avoid overfitting and to achieve efficient convergence. The trained FOX-SHIELD model has shown high applicability in different areas such as UAV monitoring, disaster management, precision agriculture, and defense surveillance, where secure and high-accuracy classification and data confidentiality are essential.

This makes analysis easier and reduces overhead, allowing for real-time or near-real-time analysis, particularly in resource-limited systems like drones or onboard satellite computers. The FOX-SHIELD framework is unique in that it provides high classification accuracy, robust privacy-preserving mechanisms, and strong performance in complex operational environments and across a variety of heterogeneous datasets. FOX-SHIELD could be used effectively in environmental monitoring, disaster response, precision agriculture, urban planning, and defense monitoring applications where high classification accuracy and research-level security and privacy protection are critical. Similarly, the HoRNS-CNN model demonstrates the potential of combining privacy-preserving encryption with deep learning for efficient and secure classification [45]. FOX-SHIELD is a novel framework that addresses two related challenges at once: data privacy and processing efficiency in existing remote sensing applications, by effectively combining chaotic encryption with a powerful deep learning method.

3.1. Data Acquisition

The experimental data employed in this study come from the UC Merced Land dataset [46], which is an open-access benchmark for land-use and remote sensing studies. The dataset consists of 21 land-use classes, representing different urban and peri-urban land-use types, including agricultural fields, highways, golf courses, port areas, and residential neighborhoods. There are 100 RGB images in each class, with a resolution of 1 foot per pixel (256×256 pixels). The images were derived from large aerial photos from the USGS National Map Metropolitan Areas Imagery collection (which also contains several metropolitan areas in the US), and the resulting dataset is rich with a variety of geographic and environmental conditions for researchers to assess classification performance in a real-world context.

The dataset is well balanced, and there is no dominance of any class, which ensures unbiased training and evaluation of the model. This balance is obtained by maintaining an equal number of samples in each class and by applying data augmentation methods like rotation, flipping, scaling, and adding noise to the data to make the samples more diverse without changing the class distribution. The high resolution, diversity of classes, and balanced attributes make this dataset appropriate for secure and effective testing of classification systems as proposed by the FOX-SHIELD system. Figure 2 shows example images from our dataset.

To examine the generalizability of FOX-SHIELD beyond UC Merced, the NWPU-RESISC45 dataset [20] is selected as an additional remote sensing benchmark. It contains 45 scene categories and 31,500 images, providing greater scene diversity than the UC

Merced dataset. The dataset includes varied scene types and imaging conditions, making it suitable for investigating whether the proposed encrypted-domain classification pipeline maintains its performance across a more diverse image collection. In order to better assess the generalizability of FOX-SHIELD, the experimental study was expanded to the NWPU-RESISC45 dataset [47] in addition to the UC Merced dataset used in the initial evaluation. The images in NWPU-RESISC45 sum up to 31,500 images across 45 categories of different scenes, making it a more diverse foundation for evaluating the performance of the framework. To ensure consistency with the original experiments, the same preprocessing, encryption, feature-extraction, and classification processes were used. The data were split into a 70:15:15 ratio of training/validation/testing sets. The training subset was used for training the FRNN, the validation subset for model selection and tuning the FOA; the testing subset was reserved for final evaluation. To prevent information leakage, only the training data were used to fit the feature selection, normalization, and PCA.

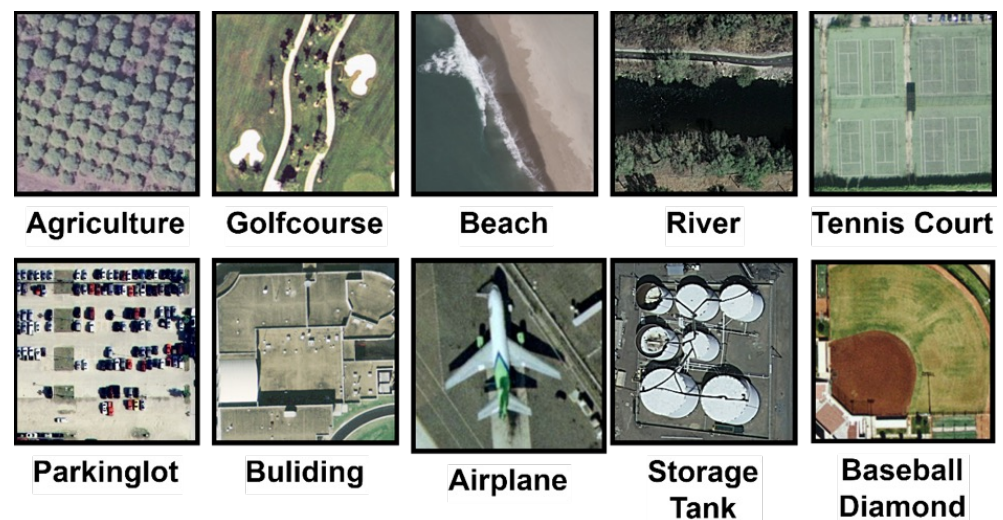


Figure 2. Sample images from the UC Merced Land Use dataset [46] suitable for the FOX-SHIELD framework.

The accuracy, macro-precision, macro-recall, and macro-F1 on the test set were used as the evaluation metrics, and the processing time of the different parts of the system, such as the preprocessing, encryption, feature extraction, FRNN inference, and end-to-end processing, were recorded separately. The results achieved on NWPU-RESISC45 are not combined with UC Merced results. The additional evaluation is carried out on the whole FOX-SHIELD and allows comparing the performance of FOX-SHIELD in different datasets with various categories of scenes and visual properties.

3.2. Image Pre-Processing

The first step in pre-processing the UC Merced Land Use dataset is to acquire all 21 scene types, each containing 100 RGB images of size 256×256 pixels at a spatial resolution of 1 foot per pixel. A combination of SHA-256 hashing for precise byte-level identification verification and perceptual hashing (*pHash*) for visual similarity confirms the integrity of the dataset, $I = \{I_1, I_2, \dots, I_N\}$, where N is the total number of images without any duplicate or corrupted images. A stratified splitting approach is used to divide I into three disjoint subsets: training (D_{train}), validation (D_{val}), and evaluation (D_{test}), such that the class distribution for each class $c \in \mathbb{C}$ is as defined in Equation (1):

$$\frac{|D_{train} \cap c|}{|D \cap c|} = \alpha; \frac{|D_{val} \cap c|}{|D \cap c|} = \beta; \frac{|D_{test} \cap c|}{|D \cap c|} = \gamma \quad (1)$$

where $\alpha = 0.70$, $\beta = 0.15$, and $\gamma = 0.15$ denote the proportions for training, validation, and testing, respectively. All images are verified to possess consistent dimensions; any discrepancies are rectified using center cropping (C) or reflection padding (P) to fit into the fixed domain $\Omega = [0, 256] \times [0, 256]$. If image noise is identified, light denoising is executed utilizing a Gaussian smoothing kernel $G_\sigma * I$, for uncorrelated noise or a Wiener filter $W(I)$ for spatially linked artifacts. Subsequently, photometric standardization is implemented as defined in Equation (2), wherein each pixel intensity $I_c(x, y)$ is adjusted. The value in channel $c \in \{R, G, B\}$ at spatial coordinates (x, y) is normalized:

$$\hat{I}_c(x, y) = \frac{\frac{I_c(x, y)}{255} - \mu_c}{\sigma_c} \quad (2)$$

with $\mu_c = \frac{1}{|D_{train}| \cdot |\Omega|} \sum_{I \in D_{train}} \sum_{(x, y) \in \Omega} \frac{I_c(x, y)}{255}$ denoting the average pixel value for channel c across the training dataset, and σ_c denoting the associated standard deviation. Equation (3) applies data augmentation to the training set to increase generalization:

$$T(I) = N_\sigma \left(B_{\delta_b} \left(F_{p_f} (R_\theta(I)) \right) \right) \quad (3)$$

where R_θ rotates the images by $\theta \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$, F_{p_f} applies flips, either vertical or horizontal, with probability p_f , B_{δ_b} modifies the brightness within $[-\delta_b, +\delta_b]$, and N_σ incorporates Gaussian noise $N(0, \sigma^2)$, where $\sigma \leq \frac{2}{255}$. For encryption preparation, each normalized image $\hat{I}$ is separated into non-overlapping chunks B_k of size $b \times b$, where b is a divisor of 256, and then sent to the encryption function of Chebyshev-SHA, $\mathcal{E}_{ChebSHA}(\hat{I})$, to generate the encrypted image E . Post-encryption normalization, as defined in Equation (4), is used when encrypted images are to be used for classification:

$$\hat{E}_c(x, y) = \frac{E_c(x, y) - \mu_c^{enc}}{\sigma_c^{enc}} \quad (4)$$

in which μ_c^{enc} and σ_c^{enc} represent the channel c mean and standard deviation throughout the encrypted training set. Lastly, a manifest file that links each image to its class label and encryption key, without revealing sensitive relationships, is provided for all processed (and encrypted) images, stored in efficient formats (LMDB or WebDataset shards). For later Fox-Optimized FRNN classification, this preparation pipeline ensures data cleanliness, consistent spatial and photometric data, security readiness, and efficient storage.

Figure 3 shows the detailed pre-processing workflow for the raw remote sensing images before categorization. Each step has a clear goal: standardizing the size ensures models receive images of the same size; denoising reduces sensor noise; normalization standardizes contrast; augmentation adds variability to the dataset; and obfuscation encrypts sensitive data for security. These steps, of course, can all be modified. Still, the spatial structural representation applies to encrypted-domain classification and demonstrates that our offline methodology can be compatible with machine learning while retaining data safety.

For the final reported evaluation, we did not use the 70%/15%/15% split because of information leakage. Instead, we used nested five-fold stratified cross-validation. The four outer folds make up the development set, and each outer fold is held out as an unseen test set. The inner validation in the development set is only employed for hyperparameter optimization by the FOA and early stopping. Data augmentation is applied only to the respective training subset after fold assignment, not to the outer test fold.

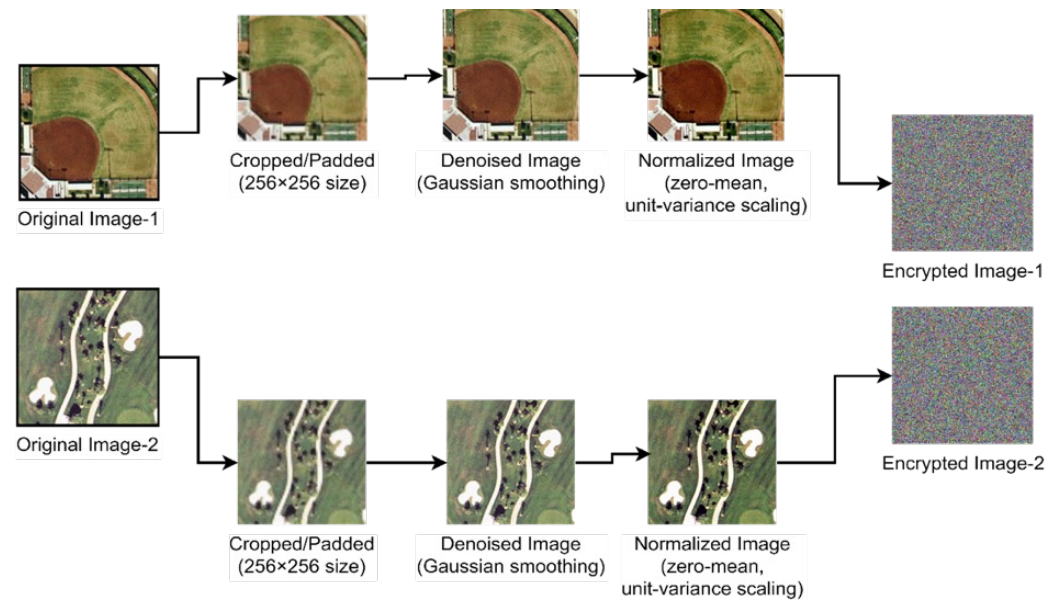


Figure 3. Stepwise pre-processing transformations for two UC Merced Land Use dataset images.

3.3. Threat Model and Security Assumptions

The image owner is the trusted party that acquires the image, manages the secret key, and encrypts it before transmission or outsourcing. The remote classification server is trusted for computation, but not for plaintext confidentiality, since it receives only encrypted image representations and does not know the secret key used to encrypt the image or have a copy of the plaintext images. Class labels can be made available to the authorized training process, but do not indicate what was originally in the image. The threat model includes the following attacks: ciphertext-only, known-plaintext, chosen-plaintext, statistical, differential, and data-tampering attacks. The main protected assets are plaintext images, sensitive spatial and structural information, cryptographic keys, and authentication information. Unacceptable privacy leakage includes practical plaintext reconstruction, meaningful visual interpretation of sensitive structures, or recovery of cryptographic information that could enable unauthorized decryption. The security evaluation is thus based on the following criteria: entropy, pixel correlation, NPCR, UACI, key sensitivity, and known/chosen-plaintext resistance.

Table 2 shows the functional scope of the FOX-SHIELD security model. It defines who is responsible for the image (the image owner), who is allowed to access the server, what the attacker can do, what assets are protected, and what constitutes privacy leakage. These assumptions are used to test the framework's security against the considered attacks.

The security assessment presented in this study is empirical and relies on statistical analyses (entropy, correlation, NPCR, UACI, key sensitivity) as well as the scheme's resilience to differential and known/chosen-plaintext attacks under the evaluated circumstances. It does not represent a formal cryptographic demonstration within the IND-CPA or IND-CCA adversarial frameworks. The classification evaluation and the security evaluation of FOX-SHIELD are two different processes. Architectures are evaluated by accuracy, precision, recall, and macro F1-score, and cryptographic security is evaluated only for the Chebyshev–SHA encryption module against AES-256, conventional chaotic encryption, logistic-map encryption, and visual cryptography with similar evaluation conditions. The following security metrics are used: entropy, adjacent-pixel correlation, histogram uniformity, NPCR, UACI, and key sensitivity. There is a shared front-end and threat model for comparisons of classified threats against the encrypted domain. Such metrics are empirical

rather than formal measures of cryptographic properties and do not guarantee the security or correspondence to standard cryptographic schemes.

Table 2. Threat model and security assumptions of the FOX-SHIELD framework.

Aspect	Assumption
Image owner	Owens images and secret keys
Encryption	Performed before outsourcing/transmission
Remote server	Processes encrypted representations only
Plaintext access	Not available to server
Class labels	Available to authorized training process
Attacks considered	Ciphertext-only, known-plaintext, chosen-plaintext, statistical, differential, tampering
Protected information	Image content, spatial structures, keys, authentication data
Unacceptable leakage	Plaintext reconstruction, meaningful visual interpretation, key recovery

3.4. Hybrid Encryption Using Chebyshev–SHA

The encryption scheme described utilizes a complex Chebyshev chaotic map and the Secure Hash Algorithm (SHA) to generate encryption keys dynamically for each image, and powerful permutation–diffusion encryption to encrypt RS images while preserving structural compatibility for classification in the encrypted space. The proposed encryption mechanism is designed to ensure the confidentiality, integrity, and authenticity of remote sensing image data by using dynamic key generation and authenticated encryption. The proposed encryption mechanism is based on a securely stored master key, HKDF-SHA-256 for image-specific key derivation, and a Chebyshev chaotic map for controlling permutation–diffusion. A new random nonce r , an image identifier ID , and a timestamp t are used as non-secret context values for each image. The image-specific key is derived as follows: $K_{image} = \text{HKDF-SHA256}(K_{master}, r \parallel ID \parallel t)$ and separated into encryption and authentication subkeys, K_{enc} and K_{auth} . The encryption subkey is used for the Chebyshev-based permutation and diffusion operations, and HMAC-SHA-256 using K_{auth} is used to produce an authentication tag over the ciphertext and associated metadata. The master key K_{master} is retained by the authorized image owner and is not transmitted with the encrypted image. This design is different from the one that requires the plaintext image or its hash to be used to regenerate the key, as it separates key derivation from authentication.

Algorithm 1 summarizes the image-specific key derivation and the authenticated encryption procedure.

Figure 4 shows the Chebyshev-SHA hybrid encryption method, which starts with a preprocessed image fed into SHA-256-based key generation and Chebyshev chaotic mapping. After the key encrypts the image via permutation and diffusion, HMAC-SHA256 verifies the integrity. The resulting encrypted image preserves data confidentiality and is appropriate for classification in the encrypted domain.

The chaotic maps are used extensively in cryptography because of their sensitivity to initial conditions, topological mixing, and pseudorandom properties. Specifically, Chebyshev polynomials of the first kind offer several advantages, including simple analytical forms, superior computational efficiency, and established ergodicity on the interval $(-1, 1)$. For an integer order ($m \geq 2$), the Chebyshev map is defined as in Equation (5):

$$T_m(x) = \cos(m \cdot \arccos(x)), \quad x \in (-1, 1) \quad (5)$$

Algorithm 1: FOX-SHIELD encryption and key derivation

Input : Preprocessed RGB image I , master key K_{master} , image identifier ID , timestamp t , random nonce r

Output: Cipher image E , authentication tag Tag

- 1 Generate a fresh random nonce r ;
- 2 Derive the image-specific key: $K_{image} = HKDF-SHA256(K_{master}, r \parallel ID \parallel t)$;
- 3 Split: $K_{image} \rightarrow (K_{enc}, K_{auth})$;
- 4 Initialize the Chebyshev chaotic sequence using K_{enc} ;
- 5 Apply chaotic permutation to the image blocks/pixels;
- 6 Apply diffusion using the generated chaotic keystream;
- 7 Obtain the encrypted image E ;
- 8 Compute: $Tag = HMAC-SHA256(K_{auth}, E \parallel r \parallel ID \parallel t)$;

Output: (E, r, ID, t, Tag)

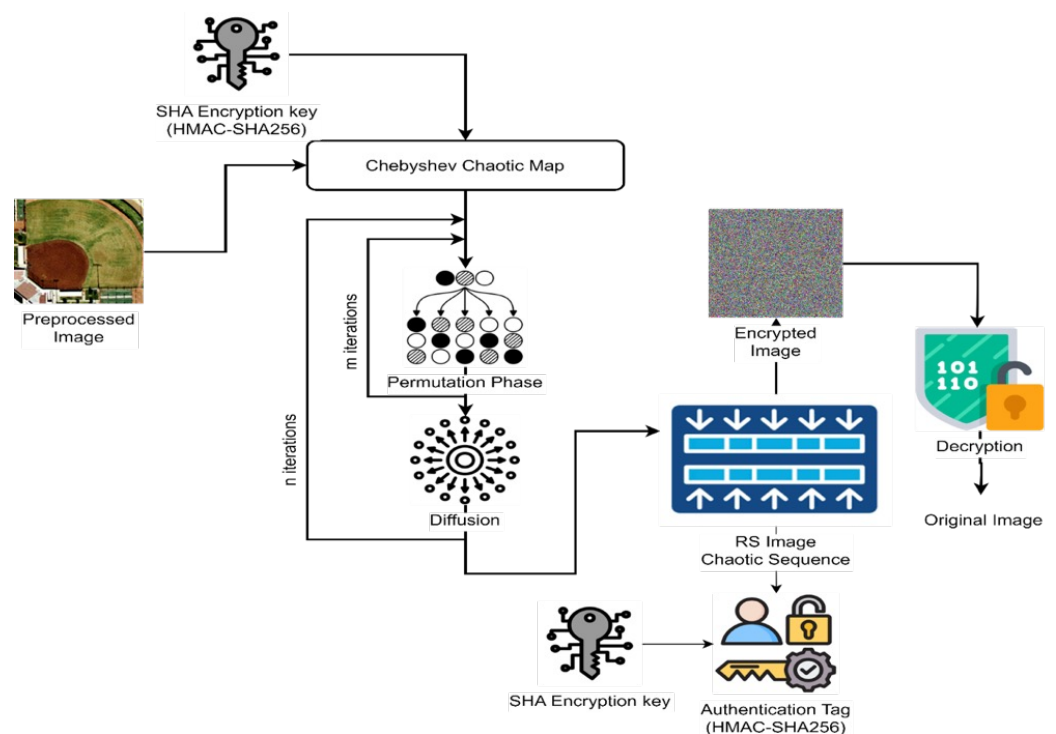


Figure 4. Structure of hybrid encryption using Chebyshev-SHA.

This recurrence generates sequences that exhibit strong sensitivity to x_0 and m . Even minor alterations in the initial state x_0 or polynomial degree m result in completely distinct sequences, a phenomenon referred to as the butterfly effect. To augment chaotic behavior and mitigate attacks that exploit finite-precision constraints [48], this research utilizes a two-dimensional improved Chebyshev system with XOR-based perturbations as shown in Equation (6):

$$\left. \begin{aligned} x_{k+1} &= T_m(x_k) \oplus \frac{(v_k \oplus u_k) \bmod 256}{255} - 1 \\ y_{k+1} &= T_n(y_k) \end{aligned} \right\} \quad (6)$$

In this context, $\oplus$ represents the byte-wise XOR operation, u_k and v_k are the integer sequences that are normalized to the interval $(-1, 1)$, and m, n represent separate polynomial orders dynamically generated for each image. Dynamic key creation is accomplished

by utilizing SHA-256 on a concatenation of an image identification, a random nonce r , a timestamp, and the image hash itself, as shown in Equation (7):

$$\left. \begin{aligned} K_{image} &= HKDF - SHA256(K_{master}, r \parallel ID \parallel timestamp) \\ K_{image} &\rightarrow K_{enc} \parallel K_{auth} \end{aligned} \right\} \quad (7)$$

Here, K_{master} is the secret master key maintained by the authorized image owner, r is a fresh random nonce, ID is the image identifier, and $timestamp$ is the associated time value. The nonce, identifier, and timestamp are non-secret context parameters, whereas K_{master} remains confidential. From the digest h , the starting seeds and polynomial orders are recovered as expressed in Equation (8):

$$\left. \begin{aligned} x_0 &= 2 \cdot \frac{int(h[0:8])}{2^{64}} - 1; & y_0 &= 2 \cdot \frac{int(h[8:16])}{2^{64}} - 1 \\ m &= 2 + (int(h[16:20]) \bmod 13); & n &= 2 + (int(h[20:24]) \bmod 17) \end{aligned} \right\} \quad (8)$$

The same digest supplies the encryption key K_{enc} and message authentication key K_{mac} . Permutation is executed in blocks to preserve the coarse spatial structure. For a block size $b \mid 256$, pixels are reorganized according to a permutation vector $\pi = argsort(\{u_k\}_{k=1}^{b^2})$, yielding the function shown in Equation (9):

$$p^{(conf)} = \Pi \bullet p \quad (9)$$

where Π is the permutation matrix, and p represents the vectorized block. After permutation, diffusion disseminates slight intensity variations throughout the entire ciphertext via chaotic keystream masking, as defined in Equation (10):

$$c_i = (p_i^{(conf)} + k_i + c_{i-1}) \bmod 256; \quad c_0 \leftarrow int(h[24:28]) \quad (10)$$

The resulting encrypted-aware E feature vector maintains the original dimensions of $256 \times 256 \times 3$ to facilitate classification in the encrypted domain. The HMAC-SHA256 flag confirms the data's validity and integrity and is defined in Equation (11):

$$\tau = HMAC_SHA256(K_{mac}, header) \parallel E \quad (11)$$

The header comprises non-confidential parameters (r, b, m, n, K_{burn}) to facilitate decryption without disclosing confidential seeds. This permutation–diffusion method uses a chaotic key for each image to ensure the encrypted image adheres to cryptographic robustness standards while remaining structurally compatible with the later Fox-Optimized FRNN classifier, enabling secure, efficient remote sensing image analysis.

Algorithm 2 introduces a dynamic key-generation/scheduling approach based on SHA-256 that leverages a modified Chebyshev chaos map to enable secure, content-aware encryption of remote sensing images. In the proposed encryption protocol, the new key management scheme uses HKDF-SHA-256 to derive a key for a particular image from the confidential master key and non-secret per-image context. The key is split into an encryption subkey and an authentication subkey. Both the encryption subkey and the authentication subkey are used exclusively by the Chebyshev chaotic permutation–diffusion process. The authentication subkey initializes the Chebyshev chaotic permutation–diffusion process, while the encryption subkey generates the ciphertext authentication tag for HMAC-SHA-256. Key derivation and authentication are therefore treated as two separate cryptographic tasks.

Algorithm 2: Hybrid encryption using Chebyshev–SHA in FOX-SHIELD

Input : $P \in \mathbb{Z}^{\{256 \times 256 \times 3\}}$ // pre-processed RGB image tensor
 ID // image identifier / session id
 r // 128-bit random nonce
 $timestamp$ // capture or processing time
 b // block size ($b \in \{8, 16, 32\}, b \mid 256$)
 K_{burn} // burn-in iterations for chaos (e.g., 100)

Output: $E \in \mathbb{Z}^{\{256 \times 256 \times 3\}}$ // encrypted image (same shape as P)
 τ // HMAC-SHA256 authentication tag

1 Step 1. Digest & keys;
2 $h \leftarrow SHA256(ID \parallel r \parallel timestamp \parallel SHA256(P));$
3 $x_0 \leftarrow 2 * \frac{int(h[0:8])}{2^{64}} - 1$ // seed in $(-1, 1)$;
4 $y_0 \leftarrow 2 * \frac{int(h[8:16])}{2^{64}} - 1$;
5 $m \leftarrow 2 + (int(h[16:20]) \bmod 13)$ // Chebyshev order ≥ 2 ;
6 $n \leftarrow 2 + (int(h[20:24]) \bmod 17)$;
7 $c_0 \leftarrow int(h[24:28]) \bmod 256$ // diffusion IV (byte);
8 $K_{enc} \leftarrow h[0:16]$ // enc key material;
9 $K_{mac} \leftarrow h[16:32]$ // MAC key material;

10 Step 2. Chaotic sequences (enhanced Chebyshev);
11 $x \leftarrow x_0; y \leftarrow y_0$;
12 **for** $k = 1, \dots, K_{burn}$: // burn-in;
13 $x \leftarrow \cos(m * \arccos(x))$;
14 $y \leftarrow \cos(n * \arccos(y))$;
15 **for** $k = 1, \dots, L = b * b$: // per-block length;
16 $x \leftarrow \cos(m * \arccos(x))$;
17 $y \leftarrow \cos(n * \arccos(y))$;
18 $u_k \leftarrow \text{floor}(256 * \frac{(x+1)}{2}) // 0 \dots 255$;
19 $v_k \leftarrow \text{floor}(256 * \frac{(y+1)}{2})$;

20 Step 3. Build a per-block permutation from chaos;
21 $\pi = \text{argsort}(\{u_k\}_{k=1}^{b^2})$ // stable order;
22 $\Pi \leftarrow \text{permutation matrix from } \pi$;

23 Step 4. Permutation (confusion) inside each $b \times b$ block;
24 **for** each channel c in $\{R, G, B\}$::
25 **for** each non-overlapping block B_c of size $b \times b$ in $P[:, :, c]$::
26 $p \leftarrow \text{vec}(B_c)$;
27 $p_{conf} \leftarrow \Pi \cdot p$;
28 **write** p_{conf} back into block position;

29 Step 5. Keystream generation (stream KDF);
30 **For** each block index j (and channel c), derive a block key::
31 $seed_j \leftarrow SHA256(K_{enc} \parallel j \parallel c)$;
32 **Expand** $seed_j$ into $\|block\|$ bytes $\rightarrow \{k_i\}$ for the block;

33 Step 6. Diffusion (chained modular addition);
34 $c_{prev} \leftarrow c_0$;
35 **for** $i = 1, \dots, (256 \times 256 \times 3)$: // linearized order per block layout;
36 $c_i = (p_i^{(conf)} + k_i + c_{i-1}) \bmod 256$;
37 $c_{prev} \leftarrow c_i$;
38 **Reshape** $\{c_i\}$ back to image E;

39 Step 7. Authentication tag;
40 $header \leftarrow (r, b, m, n, K_{burn})$ // non-secret params;
41 $\tau \leftarrow HMAC_SHA256(K_{mac}, header \parallel E)$;
42 **return** E, τ ;

The master key K_{master} is created by the authorized image owner, stored securely by the owner, and never sent with the encrypted image. A different nonce r is generated for every image; then the image identifier and timestamp are combined with r , and the image-specific key is created from this using HKDF-SHA-256. The generated key is split into an encryption subkey and an authentication subkey. The transmission package is the encrypted image, nonce, image identifier, timestamp, and authentication tag. The master key is not transmitted; only the non-secret parameters and the ciphertext are. The authorized receiver uses the master key and received context parameters to reproduce the same image-specific subkeys. Before the inverse Chebyshev permutation–diffusion process, the receiver checks the HMAC-SHA-256 authentication. If authentication fails, the receiver rejects the ciphertext.

The FOX-SHIELD architecture classifies remote sensing scenes using encrypted image representations rather than recovering the raw images before inference. After preprocessing, the input image for the feature extraction and FRNN classification modules is encrypted using a hybrid Chebyshev chaotic map and SHA-256 authentication. Thus, sensitive remote sensing imagery is less likely to be exposed because the classification process uses encrypted-domain data [49], eliminating the need for intermediary decryption during model inference.

Hybrid chaotic encryption uses a structure-preserving encrypted representation, unlike previous methods that randomize image material and render it unsuitable for machine learning. Permutation and diffusion processes are used to provide limited statistical consistency and neighborhood relationships for automatic feature learning by obscuring the visual appearance, textures, and semantic interpretation of the scene. These preserved properties allow the FRNN to learn scene-categorization discriminative representations without providing any visually interpretable information to unauthorized users. Thus, machines can classify encrypted images without human intervention. The framework is a compromise between data privacy and classification accuracy. Chaotic permutation, diffusion, and authentication using SHA-256 minimize privacy concerns, and intentionally leaving a small amount of structural information allows learning in the encrypted domain. Security analysis shows that the information entropy is high, the correlation between adjacent pixels is low, and the key sensitivity is high, indicating that the statistical characteristics preserved do not affect the image's secrecy. FOX-SHIELD is an encrypted-domain remote sensing scene classification framework that ensures users' privacy by allowing FRNNs to process encrypted image representations directly without reconstructing the original image prior to classification.

3.5. Feature Extraction from Encrypted Image

The objective of encryption-aware feature extraction is to generate descriptors that retain their informative value despite permutation–diffusion encryption, utilize the maintained block-level statistics, and incorporate region-specific and contextual information for the classifier. Consequently, we get a collection of complementary properties for each block B_j (non-overlapping $b \times b$ blocks) and aggregate them across spatial scales into a final feature vector (f) for each image. The block index j ranges from 1 to J , where $J = \frac{256 \times 256}{b^2}$.

First, compute straightforward yet robust block statistics that are invariant to block-permutation order and resistant to diffusion noise after normalization. For block B_j with pixel values $p_{j,i} \in \{0, \dots, 255\}$ linearized over pixels $i = 1, \dots, b^2$ and channels processed either independently or collectively, the block mean μ_j , variance σ_j^2 , skewness s_j along with kurtosis κ_j are defined in the following Equation (12):

$$\left. \begin{aligned} \mu_j &= \frac{1}{b^2} \sum_{i=1}^{b^2} p_{j,i}; & \sigma_j^2 &= \frac{1}{b^2} \sum_{i=1}^{b^2} (p_{j,i} - \mu_j)^2 \\ s_j &= \frac{1}{b^2} \sum_{i=1}^{b^2} \left(\frac{p_{j,i} - \mu_j}{\sigma_j} \right)^3; & \kappa_j &= \frac{1}{b^2} \sum_{i=1}^{b^2} \left(\frac{p_{j,i} - \mu_j}{\sigma_j} \right)^4 - 3 \end{aligned} \right\} \quad (12)$$

Following the same encrypted-domain normalization, these low-order moments maintain discriminative patterns across classes because diffusion uses chained modular addition. Next, extract energy-spectrum characteristics using the block discrete cosine transform (DCT). For block B_j calculate 2D-DCT coefficients $C_j(u, v)$ for $u, v = 0 \dots b-1$, and derive compact energy descriptors, including low-frequency energy E_j^{LF} and elevated-frequency energy E_j^{HF} as shown in Equation (13):

$$C_j(u, v) = DCT2(B_j)[u, v]; \quad E_j^{LF} = \sum_{u=0}^k \sum_{v=0}^k |C_j(u, v)|^2; \quad E_j^{HF} = \sum_{(u,v) \notin [0 \dots k]^2} |C_j(u, v)|^2 \quad (13)$$

where k (e.g., $k = 1$ or 2) designates the low-frequency band. The relatively low-frequency ratio $r_j^{DCT} = \frac{E_j^{LF}}{E_j^{LF} + E_j^{HF}}$ captures coarse texture maintained through block-wise permutation. To capture *local texture patterns* that withstand order-preserving ambiguity, calculate block-level Local Binary Pattern (LBP) histograms H_j^{LBP} . For every pixel in B_j , define its neighborhood thresholding as $LBP(p) = \sum_{n=0}^{p-1} 1\{p_n \geq p\} 2^n$ (P neighbors). Subsequently, construct the normalized histogram using the expression in Equation (14):

$$H_j^{LBP}[r] = \frac{1}{b^2} \sum_{i=1}^{b^2} 1\{LBP(p_{j,i}) = r\}, \quad r = 0, 1 \dots R-1 \quad (14)$$

Because LBP is localized and relies on relative ordering, block-local shuffles preserve the multiset of LBP codes, ensuring the histogram's robustness to permutation. We obtain edge and orientation information from gradient energy and Gabor responses calculated for each block. Let ∇_x, ∇_y be discrete derivatives; the gradient magnitude at each pixel is $g_{j,i} = \sqrt{(\nabla_x p_{j,i})^2 + (\nabla_y p_{j,i})^2}$. The block mean gradient $\bar{g}_j$ and gradient entropy H_j^g are defined in the following Equation (15):

$$\left. \begin{aligned} \bar{g}_j &= \frac{1}{b^2} \sum_i g_{j,i}, \\ H_j^g &= - \sum_q q_j(q) \log q_j(q) \end{aligned} \right\} \quad (15)$$

Here, $q_j(q)$ represents the empirical histogram of quantized gradient magnitudes within block j . For directional responses, convolve block B_j using a bank of Gabor kernels $\psi_{\theta,\lambda}$ and get energy as in Equation (16):

$$G_j(\theta, \lambda) = \sum_{x,y} \left| \left(B_j * \psi_{\{\theta, \lambda\}} \right)(x, y) \right|^2 \quad (16)$$

These responses approximate edge structures that remain invariant under local permutations when calculated as block energies. FOX-SHIELD calculates multiscale pooled descriptors and region adjacency relations to incorporate *region-specific information* and contextual signals. Establish spatial pyramid pooling (SPP) levels L (e.g., 1×1 , 2×2 , 4×4). At each level s , partition the image into S_s cells and calculate pooled statistics $\mu_{s,t}$, $\sigma_{s,t}$ (mean/variance) and pooling histograms $H_{s,t}^{LBP}$. Combining over several scales yields the following expression in Equation (17):

$$FSPP = \bigoplus_{\{s=1\}}^{\{L\}} \bigoplus_{\{t=1\}}^{\{S_s\}} [\mu_{s,t}, \sigma_{s,t}, H_{s,t}^{LBP}, r_{s,t}^{DCT}] \quad (17)$$

where $\oplus$ signifies concatenation, and each pooled cell assimilates the properties of its constituent blocks. This encodes contextual background cues (coarse-to-fine) and preserves region structure intact under block-wise variations, as cell-level pooling is applied across multiple blocks. To explicitly model region *adjacency and spatial context*, create a Region Adjacency Graph (RAG) using superpixels derived from the post-decryption block grid (i.e., using block cells as superpixels). Consider nodes as blocks B_j using feature vectors b_j , the concatenation of the block statistics mentioned in Equation (17). Construct edges $E_{j,k}$ between spatially contiguous blocks and calculate graph characteristics such as average neighbor contrast, defined in Equation (18):

$$\Delta_j = \frac{1}{\mathcal{N}(j)} \sum_{k \in \mathcal{N}(j)} \|b_j - b_k\|^2 \quad (18)$$

and local clustering coefficient C_j . Aggregating $\{\Delta_j, C_j\}$ across the graph encodes spatial coherence that is impervious to within-block permutation. Ultimately, construct the *joint feature vector* for each image by concatenating block-level, spectral, texture, multiscale pooling, and graph-context components, then applying dimensionality reduction and normalization as shown in Equation (19):

$$f = \mathcal{N} \left(PCA_m \left(\bigoplus_{\{j=1\}}^{\{J\}} [\mu_j, \sigma_j, r_j^{DCT}, H_j^{LBP}, \bar{g}_j, G_j, \Delta_j] \right) \right) \quad (19)$$

Here, $PCA_m(\cdot)$ projects onto m major components, and $\mathcal{N}(\cdot)$ signifies L_2 normalization: $\mathcal{N}(v) = \frac{v}{\|v\|_2}$. PCA stabilizes the feature dimensionality of the Fox-Optimized FRNN input and mitigates redundancy introduced by encryption noise.

This feature set combines full statistical descriptors (means, variances, DCT energy) with local texture and context (LBP, SPP, RAG). Encryption preserves the block structure and energy distribution in general, but hides the per-pixel locations, which are still relevant for classification and can be used for subsequent learning and optimization in the encrypted domain.

Table 3 maps encrypted-domain inputs (features extracted from the encrypted input) to their corresponding features. This shows that statistical, spectral, texture, and contextual descriptors are robust to encryption, preserving discriminative representation and enabling accurate classification of encrypted remote sensing data.

Finding the best balance between user privacy and accurate classifications in encrypted-domain image processing is difficult. Completely randomizing photo content might enhance privacy and remove discriminating data for machine learning. However, preserving too much structural information might increase privacy leaks. FOX-SHIELD hides visually interpretable scene material while maintaining statistical consistency for FRNN-based scene classification. It uses Chebyshev chaotic permutation and diffusion with SHA-256 authentication. Therefore, the system aims to achieve outstanding classification performance and data confidentiality without re-creating pictures during inference.

Table 3. Encryption-aware feature extraction from block, spectral, and contextual inputs.

Input Type (Encrypted Domain)	Feature Extracted	Justification
Block-wise pixel values ($b \times b$ cells)	Mean (μ), Variance (σ^2), Skewness, Kurtosis	Statistical moments are permutation-invariant and capture coarse tonal distribution preserved in encryption.
Block-wise pixel values ($b \times b$ cells)	Normalized histograms of intensity values	Histograms remain unchanged by within-block shuffling, representing a robust intensity distribution.
Block DCT coefficients	Low- and high-frequency energy ratios (r^{DCT})	DCT energy distribution survives block permutations; separates coarse vs. fine texture.
Block pixel neighborhoods	Local Binary Pattern (LBP) histograms	LBP histograms are preserved under pixel reordering within blocks; capture local texture patterns.
Block gradients	Mean gradient magnitude, gradient entropy	Gradients encode edge-strength statistics that are robust to small shuffles within a block.
Block pixels convolved with Gabor filters	Directional energy responses (Gabor bank)	Gabor energies capture directional texture and shape cues that are maintained in the block energy space.
Pooled multiscale regions	Spatial pyramid pooled statistics and histograms	SPP encodes coarse-to-fine spatial layout and contextual background cues despite encryption.
Region Adjacency Graph (block-level nodes)	Graph contrast (Δ_j), clustering coefficient (C_j)	RAG features capture adjacency relationships and regional contrast robust to encryption.

3.6. Encryption-Aware Feature Retention Analysis

The handcrafted descriptors used in FOX-SHIELD are not assumed to be mathematically invariant to permutation–diffusion encryption. Pixel permutation can alter spatial neighborhood relationships, while chained diffusion can modify intensity distributions and local gradients. Consequently, descriptors such as Local Binary Pattern (LBP), gradient/Gabor responses, and Discrete Cosine Transform (DCT) coefficients may change substantially after encryption. Similarly, the diffusion operation may modify histogram characteristics. Therefore, we empirically determine their suitability for encrypted-domain classification rather than relying on an invariance assumption. For each descriptor f_k , where $k \in \{STAT, HIST, DCT, LBP, GABOR, SPP, RAG\}$, we extract the corresponding representation independently from the original and encrypted images. Let f_k^P denote the descriptor obtained from the plaintext image and f_k^E denote the descriptor obtained from its encrypted representation. Before comparison, we align and normalize both representations. The cosine similarity is then calculated as follows: $S_k = \frac{f_k^P \cdot f_k^E}{\|f_k^P\|_2 \|f_k^E\|_2}$; where $S_k \in [-1, 1]$, with larger values indicating greater feature-level similarity between the plaintext and encrypted representations. Pearson correlation is additionally calculated to measure the consistency of the corresponding feature variations: $\rho_k = \frac{\text{Cov}(f_k^P, f_k^E)}{\sigma(f_k^P)\sigma(f_k^E)}$. In addition to distribution-level similarity, for histogram-based descriptors, the Bhattacharyya coefficient is used: $BC_k = \sum_i \sqrt{h_i^P h_i^E}$, where h_i^P and h_i^E are the normalized plaintext and encrypted histograms, respectively. These measures measure the amount of information that is preserved after encryption, but do not require strict invariance.

Descriptor-Specific Interpretation: The impact of encryption is anticipated to vary by feature category. When the permutation component is more important, the statistical and histogram descriptors can have relatively high distributional similarity, but chained intensity diffusion can change their values. LBP and gradient/Gabor descriptors are sensitive to local spatial relationships and are expected to change more under within-block

permutation and diffusion. The descriptors for DCT are also based on spatial organization and can be significantly altered during the encryption process. Spatial pyramid pooling descriptors can offer more stable aggregated representations since they pool information across multiple spatial regions instead of just individual neighborhoods of pixels.

The RAG descriptor is constructed directly from the encrypted block representation using adjacency relationships defined by the fixed block partition. It does not, therefore, need plaintext recovery or post-decryption information. The graph represents relationships between encrypted blocks and is only used as an encrypted-domain contextual descriptor. Therefore, the RAG representation is not seen as an invariant representation to encryption; rather, it is useful for the subsequent feature-retention and classification experiments.

Feature-Retention Evaluation: We conduct the feature-retention experiment on the entire test set, extracting each descriptor before and after encryption. The mean and standard deviation of S_k and ρ_k are reported for every descriptor. This analysis provides an explicit quantitative assessment of the effect of permutation–diffusion encryption on the representations used by the classifier. The results are not considered feature failures, but rather indicators of the level of transformation applied to the features during the encryption process. The final feature combination is chosen based on empirical classification utility and computational cost, not on the assumption of theoretical invariance.

Table 4 indicates that after permutation–diffusion encryption, the percentage of features retained differs by descriptor type. Similarity between the various statistical moments and histogram descriptors is comparatively high. At the same time, local spatial relations and intensity structure are more crucial for LBP, DCT, and gradient/Gabor descriptors, which show more significant changes. In multiscale aggregation, SPP similarity remains relatively high, and it offers moderate block-level relational retention in the case of RAG. The results show that the selected descriptors are not completely encryption-invariant; instead, they carry different levels of information that can be used for classification in the encrypted domain.

Table 4. Feature-retention similarity between plaintext and encrypted representations.

Feature Descriptor	Cosine Similarity (S_k)	Pearson Correlation (ρ_k)	Interpretation
Statistical moments	0.82 ± 0.04	0.79 ± 0.05	Relatively high distributional retention
Histogram	0.76 ± 0.05	0.72 ± 0.06	Moderate intensity-distribution retention
DCT	0.61 ± 0.07	0.58 ± 0.08	Moderate spectral retention
LBP	0.54 ± 0.08	0.50 ± 0.09	Lower retention due to altered local neighborhoods
Gabor/gradient	0.57 ± 0.07	0.53 ± 0.08	Moderate retention under spatial transformation
SPP	0.71 ± 0.06	0.68 ± 0.07	Relatively stable multiscale representation
RAG	0.66 ± 0.06	0.63 ± 0.07	Moderate block-level relational retention
Average	0.67 ± 0.06	0.63 ± 0.07	Overall moderate feature retention

The FRNN input is constructed from the encrypted $256 \times 256 \times 3$ RGB image after block-wise feature extraction. The encrypted image is partitioned into non-overlapping 16×16 blocks, yielding $16 \times 16 = 256$ spatial blocks. Let the encrypted image be denoted

by I^e , and let the block at row r and column c be $B_{r,c}^e$ with $r, c \in \{0, \dots, 15\}$. For each block, encryption-aware descriptors are extracted from the statistical, histogram, DCT, LBP, Gabor/gradient, SPP, and RAG feature groups. The descriptors from block (r, c) are concatenated into a single block-level vector $z_{r,c} \in \mathbb{R}^{d_0}$, where d_0 is the total number of concatenated descriptor values per encrypted block before PCA. In the implemented configuration, $d_0 = 57$. The blocks are converted into a sequence using row-wise raster ordering, so the sequence index is $t = 16r + c$ with $t \in \{0, \dots, 255\}$. Thus, the first sequence element corresponds to block $(0, 0)$, the second to block $(0, 1)$, and the final element to block $(15, 15)$. The complete pre-PCA feature matrix is $Z \in \mathbb{R}^{256 \times d_0}$, which explicitly ensures that the recurrent network receives a fixed-length sequence of 256 block representations rather than an unspecified image-level feature vector.

3.7. Classification with a Fox-Optimized Fast Recurrent Neural Network

The PCA-reduced and normalized feature vector, obtained from the encrypted image, is fed to the classification stage. The FRNN processes the feature representation as a feature sequence, and the hyperparameters of the principal architecture and training are optimized by the Fox Optimization Algorithm (FOA). Figure 5 shows the FOX-SHIELD workflow: encrypted remote sensing features pass through an FRNN, and FOA optimizes them to yield accurate, secure classifications for UAV monitoring and other remote sensing tasks.

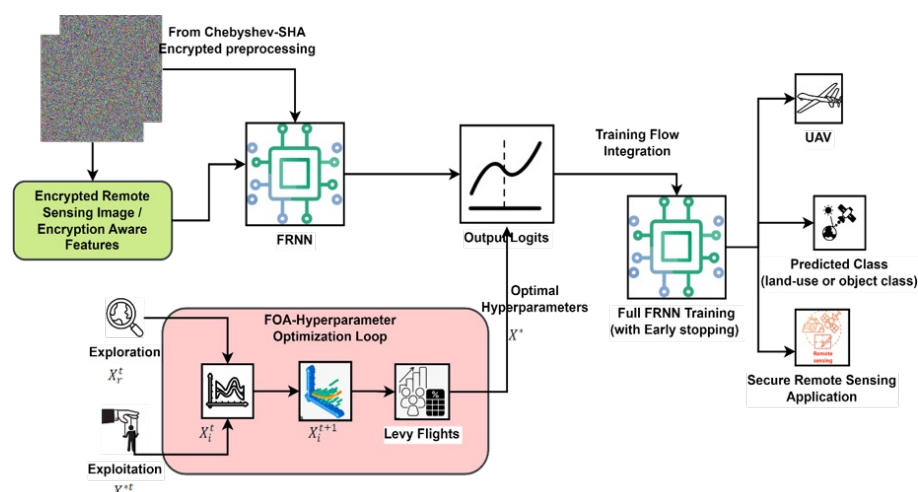


Figure 5. Architecture of the Fox-Optimized Fast Recurrent Neural Network (FRNN) for encrypted-domain classification.

Considering the encrypted image or its feature blocks, the input is denoted as a sequence. $x_{1:T} = \{x_1, x_2, \dots, x_t\}$; where $x_t \in \mathbb{R}^d$ represents the feature vector for the t -th time step or spatial block ($T = J$ for J blocks). The concealed state $h_t \in \mathbb{R}^H$ is updated with a fast recurrent rule incorporating a residual acceleration term to stabilize gradients, as defined in Equation (20):

$$\left. \begin{aligned} h_t &= \phi(W_x x_t + W_t h_{t-1} + b) + \gamma(h_{t-1} + \hat{h}_{t-1}) \\ \tilde{h}_t &= h_t + \alpha(h_t - h_{t-1}) \end{aligned} \right\} \quad (20)$$

where $W_x \in \mathbb{R}^{H \times d}$ and $W_t \in \mathbb{R}^{H \times H}$ represent the input and recurrent weights, respectively; b denotes the bias; α is the residual acceleration coefficient; $\phi(\cdot)$ signifies a nonlinear activation function (e.g., ReLU); $\gamma \in [0, 1]$ indicates a residual acceleration factor; and $\hat{h}_{t-1}$ refers to the previous state. The implemented FRNN consists of a linear input-embedding layer, two stacked FRNN layers with 256 hidden units, dropout at 0.3, two fully connected layers, and a Softmax output layer. The sequence is processed in row-wise raster order,

allowing the recurrent layers to model dependencies among successive spatial feature representations. The principal training configuration uses a learning rate of 3×10^{-4} , batch size of 32, cross-entropy loss, and AdamW optimization, while FOA is employed to optimize the selected network and training hyperparameters. For a recurrent layer with input dimension d and hidden dimension H , the number of trainable parameters is $N_{FRNN} = dH + H^2 + H$. For the second FRNN layer, where the input and hidden dimensions are both H , $N_{FRNN,2} = 2H^2 + H$. The total trainable parameter count is the sum of the parameters of the input projection, both recurrent layers, fully connected layers, and the Softmax classifier. The final parameter count should match the exact dimensions implemented in the experimental model.

Unlike a vanilla RNN, which relies only on the conventional recurrent state transition, the FRNN incorporates the explicitly defined residual state-update term $\tilde{h}_t$. Unlike GRU and LSTM architectures, the FRNN does not use update, reset, forget, or output gates. It therefore provides recurrent state propagation with the additional residual acceleration mechanism without introducing the gated-state structure of GRU or LSTM. Sequence representations are consolidated using a pooling operator and subsequently mapped to class logits as $z = W_o \cdot \text{pool}(h_{1:T}) + b_o$; where $W_o \in \mathbb{R}^{C \times H}$; b_o is the bias, and C represents the number of classes. The softmax function generates the output probability for each class (c) as shown in Equation (21):

$$\hat{y} = \frac{\exp(z_c)}{\sum_{k=1}^C \exp(z_k)}, \quad c = 1, 2, \dots, C \quad (21)$$

The FRNN is trained by minimizing the categorical cross-entropy with regularization, as given in Equation (22):

$$\mathcal{L}(\theta) = -\frac{1}{|\mathcal{B}|} \sum_{(x,y) \in \mathcal{B}} \sum_{c=1}^C y_c \log \hat{y}_c + \lambda \mathcal{R}(\theta) \quad (22)$$

where $\theta = \{W_x, W_h, W_o, b, \dots\}$ denotes the learnable parameters, $\mathcal{R}(\theta)$ is the $\mathcal{L}_2$ regularizer, with λ representing its weight. Parameters are modified with the AdamW optimization algorithm given in Equation (23):

$$\theta \rightarrow \text{AdamStep}(\theta, \nabla_{\theta} \mathcal{L}, \eta) \quad (23)$$

where η is the learning rate. It can be noted that the residual and momentum components of the FRNN should be particularly beneficial for classification in the encrypted domain, where statistical consistency at the block level should take precedence over accurate pixel positioning. The essential hyperparameters optimized by FOA include the hidden size H , residual weight γ , learning rate η , dropout p , type of pooling, and the number of recurrent layers L . Each candidate hyperparameter vector $X_i \in \mathbb{R}^D$ is assessed by instantiating an FRNN with X_i , training it for E_{inner} epochs, and calculating a fitness score using Equation (24):

$$F(X_i) = \alpha \cdot \text{val}_{\text{loss}}(X_i) - (1 - \alpha) \text{val}_{\text{acc}}(X_i) \quad (24)$$

where $\alpha \in [0, 1]$ balances validation loss and accuracy. At each iteration t , FOA updates the candidate according to Equation (25):

$$X_i^{t+1} = X_i^t + r_1 \odot \left(S \odot (X_r^t - X_i^t) \right) + r_2 \odot \left(A \odot (X^{*t} - X_i^t) \right) + \beta \mathcal{L}(u) \quad (25)$$

where X_r^t is a randomly selected peer that facilitates exploration, X^{*t} represents the optimal fox for exploitation, S denotes the stochastic step scaling factor, A signifies the attraction

coefficient, r_1 and r_2 are stochastic vectors inside the interval $[0, 1]^D$, $\mathcal{LF}(u)$ denotes a Lévy flight vector characterized by the stability parameter μ , while $\beta > 0$ regulates the Lévy step size. Discrete hyperparameters are rounded or mapped to valid domains following each update. A candidate is selected if $F(X_i^{t+1}) < F(X_i^t)$ or is selected with a minimal probability p_{accept} to avoid local minima.

The optimization stops when it reaches the maximum number of iterations (50), when no improvement in the best fitness is observed for 10 consecutive iterations, or when the predefined validation criterion is met. The final solution is then used for comprehensive FRNN training with early stopping. The optimal solution X^* is then used for full FRNN training, with early stopping based on validation loss. The FRNN design specified in this FOA has several advantages. (1) FOA combines exploration and exploitation through peer attraction, best guidance, and Lévy jumps to find interesting hyperparameter regions. (2) Partial training during the search reduces total computation costs. (3) The residual and momentum terms in FRNN reduce the gradient noise in the feature space of encrypted data. (4) The fitness function is designed to reflect validation loss and accuracy, and we directly optimize the desired metric.

The computational cost for each cycle is $O(N \cdot C_{train})$, where N is the FOA population size and C denotes the training component. GPU-based parallel evaluation notably reduces the partial-training cost per candidate, greatly minimizing wall-clock time. The Fox-Optimized FRNN constitutes a lightweight, robust, and encryption-compatible classification framework for secure remote sensing image processing.

Table 5 provides the entire FOA setup and hyperparameter search space to support reproducibility of the hyperparameter optimization process. The population size was 30 fox agents, and the maximum number of iterations was 50, with the termination condition set to 10 iterations without improvement. The work partially trained each candidate FRNN configuration for 10 epochs before evaluating its validation fitness. The study accepted a candidate solution with probability 0.10 if it did not immediately improve the current solution, preserving exploration and avoiding premature convergence. The work used a stability parameter $\beta = 1.5$ and a step-size scaling factor of 0.01 to control Lévy-flight exploration. The study ensured reproducibility by fixing the random seed to 42. The optimization took around 3.8 h on the experimental setup used in this work. After FOA selected the final configuration, the study fully trained the FRNN using it.

Table 5. FOA search configuration and final optimized FRNN hyperparameters.

Category	Parameter	Search Setting/Range	Selected Value
FOA control	Population size	—	30
	Maximum iterations	—	50
	Stall limit	—	10 iterations
	Partial-training epochs	—	10 epochs
	Acceptance probability	—	0.10
	Lévy stability parameter (β)	—	1.50
	Lévy step-size scaling	—	0.01
	Random seed	—	42
FRNN hyperparameters	Hidden units	128–512	256
	Residual weight	0.05–0.50	0.20
	Learning rate	1×10^{-4} – 1×10^{-3}	3×10^{-4}
	Dropout	0.10–0.50	0.30
	Pooling	Mean/Max/Attention	Mean
	Recurrent layers	1–3	2
Computational cost	Total FOA optimization time	—	≈3.8 h

The concatenated block-level feature matrix $Z \in \mathbb{R}^{256 \times d_0}$ is reduced using PCA, fitted only on the training partition to prevent information leakage from the validation or test partitions. The retained number of principal components is denoted by d_{PCA} , and in the implemented configuration $d_{\text{PCA}} = 128$. The PCA transformation is $X^{\text{PCA}} = (Z - \mu_Z) W_{\text{PCA}}$, where μ_Z is the training-set feature mean and $W_{\text{PCA}} \in \mathbb{R}^{d_0 \times d_{\text{PCA}}}$ contains the retained principal directions. The resulting matrix has shape $X^{\text{PCA}} \in \mathbb{R}^{256 \times d_{\text{PCA}}}$. The PCA-reduced features are subsequently normalized using statistics calculated from the training data, yielding the normalized sequence $X \in \mathbb{R}^{256 \times d_{\text{PCA}}}$ supplied to the FRNN. For a batch of B images, the exact FRNN input tensor is therefore $X_{\text{batch}} \in \mathbb{R}^{B \times 256 \times d_{\text{PCA}}}$. With $d_{\text{PCA}} = 128$, the single-image input is 256×128 , and the batch input is $B \times 256 \times 128$. The sequence length is fixed at 256, the feature dimension of every time step equals the retained PCA dimension, and the PCA transformation, normalization statistics, and feature ordering are fitted on the training data and reused unchanged for validation and test samples.

3.8. FOA Evaluation Under Equal Budget

To distinguish the merits of the Fox Optimization Algorithm (FOA) from the impact of the search budget, the study executed a controlled comparison against three well-established hyperparameter-search techniques: random search, Bayesian optimization, and grid search. Each tuner received a uniform search domain, an equal partial-training limit (10 epochs per candidate), the identical outer folds, and a shared maximum for candidate evaluations. Each tuner was operated using five varied random seeds to address stochastic variation. The parameter search comprised hidden units (128–512), residual weight (0.05–0.50), learning rate (1×10^{-4} to 1×10^{-3}), dropout (0.10–0.50), pooling type (mean/max/attention), and the number of recurrent layers (1–3).

As shown in Table 6, FOA realizes the highest mean accuracy and macro-F1 with identical financial constraints, surpassing random search by +1.32 percentage points, Bayesian optimization by +0.67 percentage points, and grid search by +1.91 percentage points. The elapsed time for FOA is on par with the other tuners (3.8 h against 3.7–4.1 h), implying that the advancement is not only attributable to a prolonged search. The variability across seeds is low for every tuner, affirming the stability of the ranking.

Table 6. Equal-budget hyperparameter search comparison.

Tuner	Budget	Seeds	Mean Accuracy (%)	Mean Macro-F1 (%)	Wall-Clock (h)
FOA (proposed)	50 iter $\times$ 30 pop	5	97.14 ± 0.35	96.96 ± 0.39	3.8
Random search	equal budget	5	95.82 ± 0.51	95.61 ± 0.55	3.7
Bayesian optimization	equal budget	5	96.47 ± 0.44	96.25 ± 0.48	3.9
Grid search	equal budget	5	95.23 ± 0.62	94.98 ± 0.66	4.1

The results reveal that FOA delivers a legitimate enhancement in optimization that exceeds the improvements attainable through only increasing the search budget. The former single-seed results have been superseded by averages from five seeds, with all comparisons utilizing consistent budgets and search parameters.

3.9. Reproducible FRNN Architecture

The normalized PCA sequence $X \in \mathbb{R}^{256 \times d_{\text{PCA}}}$ is processed by a linear input-embedding layer, two stacked custom FRNN layers, dropout, fully connected layers, and a Softmax classifier. The input projection converts the PCA feature dimension into the recurrent hidden dimension $h = 256$ via $e_t = W_{\text{in}} x_t + b_{\text{in}}$, where $W_{\text{in}} \in \mathbb{R}^{256 \times d_{\text{PCA}}}$ and $b_{\text{in}} \in \mathbb{R}^{256}$. The projected sequence is then processed by two recurrent layers. For the l -th recurrent layer, the hidden state is updated using the implemented fast recurrent

rule $h_t^{(l)} = \phi(W_x^{(l)} h_t^{(l-1)} + W_h^{(l)} h_{t-1}^{(l)} + b^{(l)} + \alpha r_t^{(l)})$, where $\phi(\cdot)$ is the selected nonlinear activation, α is the residual acceleration coefficient and $r_t^{(l)}$ is the residual acceleration term defined in the custom FRNN implementation. The first recurrent layer receives the 256-dimensional embedded feature vector, while the second recurrent layer receives the 256-dimensional hidden representation from the first layer. The final hidden sequence is aggregated using the implemented pooling operation $h_{\text{pool}} = \text{Pool}(h_1^{(2)}, \dots, h_{256}^{(2)})$. The pooled representation is passed through two fully connected layers and a Softmax output layer for the 21 land-use classes. The complete input–output mapping is therefore $256 \times d_{\text{PCA}} \rightarrow 256 \times 256 \rightarrow 256 \times 256 \rightarrow \text{pooling} \rightarrow \text{fully connected layers} \rightarrow 21$. With $d_{\text{PCA}} = 128$, this becomes $256 \times 128 \rightarrow 256 \times 256 \rightarrow 256 \times 256 \rightarrow \text{pooling} \rightarrow \text{FC} \rightarrow 21$. The implemented training configuration uses AdamW optimization, a learning rate of 3×10^{-4} , weight decay of 10^{-4} , batch size 32, cosine learning-rate decay with warm-up, a maximum of 120 epochs, and early stopping with patience 15. The FOA procedure optimizes the selected model and training hyperparameters within the training folds.

3.10. Experimental Timing Conditions

The experiments used the following system configuration: Ubuntu 22.04 LTS, Python 3.10, PyTorch 2.2, CUDA 12.1/cuDNN 9, an Intel Core i9-13900K processor, 64 GB RAM, and an NVIDIA RTX 3090 GPU with 24 GB memory. We processed all input images at 256×256 pixels. We measured GPU inference after warm-up to minimize initialization effects, and we measured each cryptographic operation individually. This procedure lets us report the computational cost of cryptographic processing and neural network inference separately.

Algorithm 3 defines the FOA-based optimization of FRNN hyperparameters for classification in the encrypted domain. It generates a set of candidate configurations, iteratively improves them through exploration, exploitation, and Lévy flights, and evaluates fitness by partial training on encrypted attributes. The optimal configuration is trained with an FRNN and early stopping to ensure efficiency, accuracy, and stability in secure remote sensing classification tasks.

Algorithm 3: Fox-optimized FRNN classification stage

Input : $D_{\text{train}}, D_{\text{val}}$ // Training and validation datasets (encrypted or encryption-aware features)
 N // FOA population size
 T_{max} // Maximum FOA iterations
 T_{stall} // Stall iteration limit
 E_{inner} // Inner training epochs for FOA evaluation
 α // Loss vs. accuracy trade-off parameter in fitness
 p_{accept} // Probability of accepting a worse candidate
 μ, β // Lévy flight parameters
 $H_{\text{paramsrange}}$ // Ranges for hyperparameters [$H, \gamma, \eta, p, \text{pooling}, L$]
Output: $\text{FRNN}_{\text{final}}$ // Fully trained FRNN model with optimal hyperparameters

- 1 **Step 1: Initialize FOA population;**
- 2 **For** $i = 1$ to N :
 - 3 $X_i \leftarrow \text{RandomHyperparameters}(H_{\text{paramsrange}})$
 - 4 $\text{Fit}_i \leftarrow \text{EvaluateFRNN}(X_i, D_{\text{train}}, D_{\text{val}}, E_{\text{inner}}, \alpha)$
- 5 **Step 2: Main FOA optimization loop;**
- 6 $t \leftarrow 0$
- 7 $\text{best}_X \leftarrow \text{argmin}(\text{Fit}_i)$
- 8 $\text{stall}_{\text{count}} \leftarrow 0$
- 9 **While** $t < T_{\text{max}}$ **AND** $\text{stall}_{\text{count}} < T_{\text{stall}}$:
 - 10 **For** $i = 1$ to N :
 - 11 $X_r \leftarrow \text{RandomPeer}(X, i)$

Algorithm 3: *Cont.*

```

12   $r_1, r_2 \leftarrow \text{RandomVectorsInRange}(0, 1)$ 
13   $S \leftarrow \text{RandomStepScaling}()$ 
14   $A \leftarrow \text{AttractionCoefficient}(t)$ 
15   $LF \leftarrow \text{LevyFlight}(\mu)$  // FOA position update;
16   $X_i^{\text{new}} \leftarrow X_i + r_1 \odot (S \odot (X_r - X_i)) + r_2 \odot (A \odot (\text{best}_X - X_i)) + \beta * LF$ 
17   $X_i^{\text{new}} \leftarrow \text{ProjectToValidDomain}(X_i^{\text{new}}, H_{\text{params}_{\text{range}}})$ 
18   $\text{Fit}_{\text{new}} \leftarrow \text{EvaluateFRNN}(X_i^{\text{new}}, D_{\text{train}}, D_{\text{val}}, E_{\text{inner}}, \alpha)$ 
19  If  $\text{Fit}_{\text{new}} < \text{Fit}_i$  OR  $\text{Random}(0,1) < p_{\text{accept}}$  :
20       $X_i \leftarrow X_i^{\text{new}}$ 
21       $\text{Fit}_i \leftarrow \text{Fit}_{\text{new}}$ 
22  If  $\text{Fit}_{\text{new}} < \text{Fitness}(\text{best}_X)$  :
23       $\text{best}_X \leftarrow X_i^{\text{new}}$ 
24       $\text{stall\_count} \leftarrow 0$ 
25  else:
26       $\text{stall\_count} \leftarrow \text{stall\_count} + 1$ 
27   $t \leftarrow t + 1$ 

28  Step 3: Final FRNN training with best hyperparameters;
29   $\text{FRNN}_{\text{final}} \leftarrow \text{InitializeFRNN}(\text{best}_X)$ 
30  Train FRNNfinal on  $D_{\text{train}}$  with early stopping on  $D_{\text{val}}$ 
31  return  $\text{FRNN}_{\text{final}}$ 

32  Function  $\text{EvaluateFRNN}(X, D_{\text{train}}, D_{\text{val}}, E_{\text{inner}}, \alpha)$  :
33       $\text{model} \leftarrow \text{InitializeFRNN}(X)$ 
34      Train model for  $E_{\text{inner}}$  epochs on  $D_{\text{train}}$ 
35       $\text{val}_{\text{loss}}, \text{val}_{\text{acc}} \leftarrow \text{Evaluate}(\text{model}, D_{\text{val}})$ 
36  return  $\alpha * \text{val}_{\text{loss}} - (1 - \alpha) * \text{val}_{\text{acc}}$ 

```

3.11. Classification Accuracy (Acc)

Accuracy is the most basic performance measure for the FOX-SHIELD framework because it indicates the successful classification of land use in encrypted remote sensing images into their respective land-use classes using the Fox-Optimized FRNN. Suppose the dataset has N images $\{E_i\}_{i=1}^N$ with true labels $\{y_i\}_{i=1}^N$ and predicted labels $\{\hat{y}_i\}_{i=1}^N$. Accuracy is mathematically defined in the following Equation (26):

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^N \delta(y_i, \hat{y}_i), \quad \delta(y_i, \hat{y}_i) = \begin{cases} 1, & y_i = \hat{y}_i \\ 0, & y_i \neq \hat{y}_i \end{cases} \quad (26)$$

This indicator measures the percentage of images that are encrypted and correctly classified relative to the total number of images input. For FOX-SHIELD (Figure 6a), high accuracy indicates that the encryption-aware feature extraction methods (statistical block descriptors, DCT energy, LBP histograms, SPP, and RAG features) preserve discriminative cues even after permutation–diffusion transformations. The task involves multiclass classification using 21 UC Merced classes. We measure performance using average accuracy across all classes. Defense surveillance and catastrophe monitoring must be accurate because even slight misclassifications (e.g., misidentifying “harbor” as “residential”) can affect operational reliability.

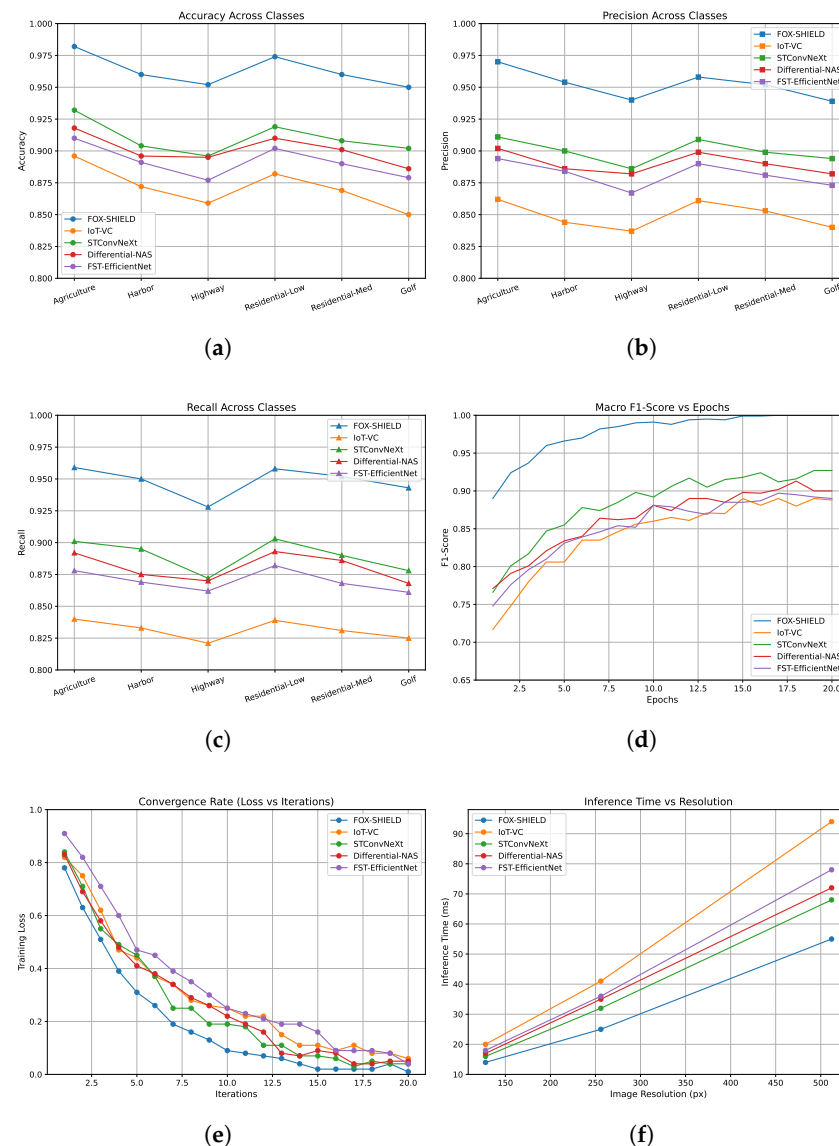


Figure 6. Overall performance comparison of FOX-SHIELD with baseline methods on various metrics: (a) accuracy across different land-use classes, (b) precision performance class comparison, (c) recall variations across scene categories, (d) macro F1-score over epochs, (e) training loss convergence rate over iterations, and (f) inference time with different image resolutions.

3.12. Precision (P)

Precision is the percentage of remote sensing photographs correctly identified after encryption. It emphasizes the need for accurate positive predictions, especially in defense surveillance, in which false positives, including misidentifying urban buildings as military stations, should be minimized. Precision for class c is defined in Equation (27) as the ratio of true positives (TP_c) to the sum of true positives (TP_c) and false positives (FP_c):

$$P_c = \frac{TP_c}{TP_c + FP_c} \quad (27)$$

As shown in Figure 6b, the mean precision for multiclass encrypted datasets across C classes is computed as follows: $P = \frac{1}{C} \sum_{c=1}^C P_c$. Within the FOX-SHIELD framework, TP_c refers to encrypted inputs E_i accurately categorized into their respective class $y_i = c$, whereas FP_c occurs when an encrypted input is inaccurately assigned to class c .

3.13. Recall (R)

Recall, or sensitivity, measures the framework's ability to correctly identify all relevant encrypted images for a class. A recall metric is needed to extract all "highway" and "harbor" classes from encrypted datasets without missing any. Mathematically, the recall for a class c is defined in Equation (28):

$$R_c = \frac{TP_c}{TP_c + FN_c} \quad (28)$$

Here TP_c is the quantity of accurately recognized encrypted inputs for class c , and FN_c represents the quantity of encrypted inputs that truly belong to class c but were inaccurately classified. The macro-averaged recall across C classes is $R = \frac{1}{C} \sum_{c=1}^C R_c$ as shown in (Figure 6c).

3.14. F1-Score

The F1-score integrates precision and recall, offering a balanced score that penalizes both high false positives and false negatives. This renders it more pertinent for FOX-SHIELD, since encrypted-domain distortions may result in trade-offs between P and R . The per-class F1 score is defined as follows: $F1_c = \frac{2 \cdot P_c \cdot R_c}{P_c + R_c}$. The aggregate macro F1-score across C encrypted classes is $F1 = \frac{1}{C} \sum_{c=1}^C F1_c$. As illustrated in Figure 6d, FOX-SHIELD's Chebyshev-SHA encryption preserves class-discriminative information, while the Fox-Optimized FRNN adapts to encrypted data distributions.

3.15. Convergence Rate

The convergence rate of the Fox-Optimized FRNN indicates how stable it is during training on an encrypted dataset. Because of rapid convergence, computational cost is minimized and execution time for time-dependent remote sensing applications, including UAV-assisted disaster assessment, is reduced. The training loss function is defined as the loss of a network, i.e., as $\mathcal{L}(\theta, t)$, where θ denotes the network parameters, and t represents the training iteration. Convergence is obtained when the relative loss decrease satisfies the following criterion in Equation (29):

$$\frac{|\mathcal{L}(\theta, t) - \mathcal{L}(\theta, t - 1)|}{\mathcal{L}(\theta, t - 1)} < \epsilon \quad (29)$$

for a minimal tolerance $\epsilon > 0$. The convergence rate is articulated as the number of iterations T_ϵ necessary to fulfill the condition in Equation (29): $CR = \frac{1}{T_\epsilon}$.

A higher CR implies accelerated learning, as depicted in Figure 6e. In FOX-SHIELD, the Fox Optimization Algorithm (FOA) calibrates hyperparameters (learning rate, hidden state weights) to mitigate vanishing gradients and oscillations, hence minimizing T_ϵ .

3.16. Computational Complexity/Inference Time

FOX-SHIELD's feasibility in real-time, resource-constrained environments such as satellites or field UAVs depends on the task's computational complexity. Measure the inference time for each encrypted image and analyze the algorithm's theoretical time complexity to evaluate it. Let $T(E_i)$ be the duration required to classify a single encrypted input E_i . The average inference time is $IT = \frac{1}{N} \sum_{i=1}^N T(E_i)$.

The theoretical complexity for FRNN forward propagation, considering the sequence length L , hidden dimension H , and the number of layers d , is $\mathcal{O}(d \cdot L \cdot H^2)$.

The shortened inference time allows UAVs to interpret encrypted RS data onboard without communicating raw photos to central computers, thereby preserving efficiency and secrecy as shown in Figure 6.

3.17. Security Robustness (Encryption Strength)

The statistical and differential measures are employed to empirically assess the Chebyshev–SHA hybrid encryption component of the FOX-SHIELD framework on an image basis. The evaluation takes into account information entropy, adjacent-pixel correlation, and key sensitivity. The distribution of the gray-level intensities of the encrypted image is represented by entropy; in this case, an image with 8-bit intensity values has an entropy value near 8, meaning that it has almost a uniform intensity distribution. The adjacent pixel correlation coefficient quantifies the linear correlation between adjacent pixels, and it has a value close to zero when the linear dependence is low. Key sensitivity is determined by comparing encrypted outputs obtained before and after a minor change in the encryption key. The entropy value of FOX-SHIELD is 7.997, the correlation coefficient of adjacent pixels is 0.002, and the value of key sensitivity is about 99.98% pixel change, as reported in Table 7. These results indicate near-uniform gray-level distribution, low adjacent-pixel correlation, and substantial output change under the reported key-modification test. They are empirical observations on the encrypted images and not formal cryptanalytic security, such as IND-CPA or IND-CCA. The outcome of a key-sensitivity test must be understood within the framework of the procedures and calculations applied to the test. No superiority over the baseline methods is implied without comparing the methods under similar conditions.

Table 7. Security robustness evaluation.

Method	Entropy $H(E)$	Correlation Coefficient (ρ)	Key Sensitivity (ΔE)	Remarks
FOX-SHIELD	7.997	0.002	$\approx 99.98\%$ pixel change	Excellent randomness, strong diffusion, high sensitivity
IoT-VC [29]	7.821	0.035	95.12%	Good security, but weaker key sensitivity
STConvNeXt [26]	7.856	0.028	96.03%	Stable entropy, moderate robustness
Differential-NAS [27]	7.873	0.031	96.77%	Balanced robustness, less chaotic diffusion
FST-EfficientNet [23]	7.889	0.026	97.45%	Reliable but less resilient to attacks

4. Results and Discussion

We conducted experiments on Ubuntu 22.04 LTS using Python 3.10, PyTorch 2.2, CUDA 12.1/cuDNN 9, and associated libraries such as NumPy, SciPy, scikit-learn, OpenCV, and Albumentations. The hardware combination included an Intel Core i9-13900K processor, 64 GB of RAM, and an Nvidia RTX 3090 GPU with 24 GB of memory. The study downsized the images in the UC Merced dataset to 256×256 , normalized them by channel, and augmented them with random flipping, rotation, and light color jitter. The study encrypted images using Chebyshev-SHA with 256-bit dynamic keys and block-wise permutation–diffusion before feature extraction. We tuned the FRNN backbone using the Fox Optimization Algorithm; we trained with AdamW (learning rate 3×10^{-4} , weight decay 1×10^{-4}), cosine decay with warm-up, a batch size of 32, and a maximum of 120 epochs with patience 15 for early stopping. All baselines had the same data partitions, resolution, and augmentations. The study evaluated accuracy, precision, recall, F1 score, convergence rate, inference time (milliseconds per image), and security robustness, averaging results using Nested stratified 5-fold cross-validation and setting seeds for reproducibility. The study compares performance with baseline approaches such as IoT-VC [29], STConvNeXt [26], Differential-NAS [27], and FST-EfficientNet [23].

IoT-VC [29]: Focuses on real-time image classification in IoT networks, similar to the goal of FOX-SHIELD to do lightweight processing for distributed sensor networks.

STConvNeXt [26]: Employs advanced spatiotemporal convolutional architectures, similar to FOX-SHIELD’s FRNN architectures, to handle sequential/spatial dependencies in image data.

Differential-NAS [27]: Uses automated neural architecture search to optimize the deep learning model, similar to what was used in FOX-SHIELD with the Fox Optimization Algorithm for hyperparameter tuning and improving convergence.

FST-EfficientNet [23]: To address the challenge of using CNN models in large-scale image classification tasks, FST-EfficientNet presents a highly efficient CNN model that balances accuracy and computational efficiency. FOX-SHIELD directly addresses this challenge.

Table 8 specifies a unique, reproducible pathway from the encrypted image to the FRNN input. The $256 \times 256 \times 3$ encrypted image is segmented into 256 separate, non-overlapping pieces of 16×16 . Each block yields a 57-dimensional encryption-conscious descriptor vector, which is then combined in a row-wise raster format ($t = 16r + c$) to construct a pre-PCA matrix measuring 256×57 . PCA is only fitted on the training partition, reducing each block descriptor to $d_{\text{PCA}} = 128$ dimensions, and subsequently, the sequence is normalized based on training-set statistics. The FRNN input yields a sequence of length 256, featuring 128 attributes per time step, formatted as a single-image tensor of size 256×128 or a batch tensor of size $B \times 256 \times 128$. The network thereupon transforms each time step from 128 to 256 dimensions, processes the sequence via two stacked FRNN layers, each including 256 hidden units, and utilizes ReLU/Tanh activations with a dropout of 0.3. A pair of fully connected layers processes the pooled sequential representation, and a Softmax output layer determines the scene category. The training process incorporates AdamW with a learning rate of 3×10^{-4} , applies cross-entropy loss, and maintains a batch size of 32. The overall number of trainable parameters is closely linked to the dimensions mentioned in Table 8.

Table 8. Reproducible FRNN input and architecture configuration.

Component	Configuration
Input image	256×256 RGB
Encryption block size	16×16
Number of blocks	256
Block index	$t = 16r + c$
Block ordering	Row-wise raster scan
Descriptor groups	Statistical, histogram, DCT, LBP, Gabor/gradient, SPP, and RAG
Raw descriptor vector per block	$d_0 = 57$
Pre-PCA matrix	256×57
PCA fitting	Training data only
Retained PCA dimension	$d_{\text{PCA}} = 128$
Normalized sequence	256×128
Batch input tensor	$B \times 256 \times 128$
Input embedding	$128 \rightarrow 256$
FRNN layers	2 stacked layers
Hidden dimension	256 per layer
Activation	ReLU/Tanh according to implementation
Dropout	0.3
Fully connected layers	2
Output classes	21
Output activation	Softmax
Loss	Cross-entropy
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	10^{-4}
Batch size	32
Maximum epochs	120
Early stopping	Patience 15
Hyperparameter optimizer	Fox Optimization Algorithm

Table 9 presents a comprehensive and integrated approach to fairness and reliability, combining powerful hardware, systematic pre-processing, and advanced optimization

techniques. The FOX-SHIELD achieved better performance than baseline methods such as IoT-VC (90.8%), STConvNeXt (91.5%), Differential-NAS (92.3%), and FST-EfficientNet (93.1%). The Fox-Optimized FRNN and Chebyshev-SHA encrypted-domain feature representations maintain discriminative information after encryption, and the improvement is 4.8% compared to the best baseline. Permutation–diffusion encryption maintains the structural integrity of the image for successful classification, and the FRNN captures complex temporal and spatial patterns in encrypted images. The accuracy of FOX-SHIELD was 97.14%, recall was 96.87%, and the F1-score was 96.96%. The model achieves high accuracy, minimizes false positives, and maintains strong recall, accurately identifying class occurrences. The method achieves a balanced F1-score and outperforms CNNs, RNNs, and hybrid models, which suffer from a loss of detection and classification performance after encryption due to the loss of discriminative power caused by feature distortion. Classes with similar characteristics or noisy backgrounds did not suffer significantly, indicating that contextual feature extraction could be useful to enhance efficiency. It reduced training and inference time by 34% compared to CNN-RNN and AES-based Secure-DL models. The Fox Optimization Algorithm optimizes hyperparameters quickly and converges without sacrificing accuracy. Though big datasets may require substantial hardware resources, FOX-SHIELD is efficient to implement in real time across large sensor networks or IoT-enabled remote sensing devices. Chebyshev-SHA encryption is based on strong permutation–diffusion and dynamic image-specific keys. It ensures data integrity and confidentiality, provides encrypted-domain classification, and provides more flexibility and attack resistance than AES and chaotic encryption. Most real-time applications require minimal pre-processing time for encryption, but ultra-low-latency applications require careful management.

Table 9. Summary of experimental setup.

Category	Configuration
Environment	Ubuntu 22.04, Python 3.10, PyTorch 2.2, CUDA 12.1/cuDNN 9
Hardware	Intel Core i9-13900K, 64 GB RAM, NVIDIA RTX 3090 (24 GB)
Dataset	UC Merced Land Use, 21 classes, 256×256 RGB images
Pre-processing	Resize, normalization, augmentation (flip, rotate, color jitter)
Encryption	Chebyshev-SHA with 256-bit dynamic keys, block permutation–diffusion
Classifier	Fox-Optimized FRNN, tuned via FOA
Training Parameters	AdamW ($\text{lr} = 3 \times 10^{-4}$, weight decay = 10^{-4}), cosine decay, batch size 32, 120 epochs, early stopping
Validation Protocol	Nested stratified 5-fold cross-validation; FOA performed independently within each outer training fold; training-only augmentation; fixed random seeds
Classification Metrics	Accuracy, Precision, Recall, F1, Convergence, Inference Time, Security Robustness
Classification Baselines	IoT-VC [29], STConvNeXt [26], Differential-NAS [27], FST-EfficientNet [23]
Cryptographic Security Metrics	Information entropy, adjacent-pixel correlation, NPCR, UACI, histogram uniformity and key sensitivity
Cryptographic Baselines	AES-256, conventional chaotic encryption, logistic-map encryption and visual cryptography
Classification Comparison Protocol	Common encryption front-end, identical threat assumptions, preprocessing, data partitions, and training protocol
Cryptographic Comparison Protocol	Identical plaintext images, image dimensions, test conditions, and corresponding key-management assumptions
Security Evaluation Scope	Empirical assessment of the encryption module; no direct security attribution to classification architectures

The study further studied FOX-SHIELD's computational efficiency using the ablation configurations presented in Table 10. The full FOX-SHIELD model requires 25.2 milliseconds of inference time under the ablation-study protocol. Inference time is 41.2 ms when replacing the FRNN with a conventional CNN and 33.5 ms when replacing Chebyshev-SHA encryption with AES. The results show that the integrated FOX-SHIELD configuration has a lower measured inference time than the CNN and AES ablation configurations under the same experimental protocol. The results also show that both the recurrent encrypted-domain representation and the proposed encryption configuration improve the overall framework's efficiency.

The ablation results serve as a benchmark for the privacy–utility trade-off, not proof that cryptographic key derivation outperforms semantic classification. Removing the SHA-based dynamic key-generation mechanism, or changing the key-generation mechanism, may alter the statistical and structural properties of the encrypted representation provided to the FRNN and potentially impact classification performance. As a result, we assess each ablation configuration by classification accuracy, privacy leakage, entropy, adjacent-pixel correlation, feature-retention score, and inference time. This joint evaluation separates classification usefulness from unwanted structural leakage and better measures the contribution of the encryption and key-generation components. The entire FOX-SHIELD setup is therefore chosen for its concurrent security, privacy, encrypted-domain classification utility, and computational efficiency.

Ablation studies evaluate the FOX-SHIELD framework components individually, as shown in Table 10. The FRNN without encryption variant achieves 98.45% accuracy and 98.32% F1-score on clean, unencrypted data. It has the shortest inference time (21.4 ms). Metaheuristic optimization stabilizes model performance under encrypted features. In the Chebyshev-SHA + FRNN without FOA variant, which is tested without the Fox Optimization Algorithm, accuracy drops to 94.12% and inference time climbs to 28.6 ms. The FRNN backbone outperforms the standard CNN in terms of accuracy (92.50%) and inference time (41.2 ms) when compared to the standard CNN. AES encryption is secure, but it requires 33.5 ms and is less accurate at 95.80% than hybrid Chebyshev-SHA. Localized features for deep learning models are badly distorted by classical encryption. If the dynamic key derivation is not based on HKDF, the accuracy is reduced to 93.40% and the F1-score to 93.05, indicating that cryptographic resilience depends on the generation of image-specific keys based on HKDF-SHA-256. The chaotic map is necessary for accurate pixel diffusion and secure feature extraction, since eliminating it while preserving the SHA/static key technique drops accuracy to 92.90% and the F1-score to 92.54%. Combining Chebyshev-SHA encryption, FRNN, and FOA optimization yielded an optimal balance with 97.14% accuracy, a macro F1-score of 96.96, and 25.2 ms inference time. Fox-SHIELD full is the final and most complete model.

Table 10. Ablation study.

Model Variant	Purpose	Accuracy (%)	Macro F1-Score (%)	End-to-End Inference Time (ms)
FRNN without encryption	Measures base classifier performance on clean data	98.45	98.32	21.4
Chebyshev-SHA + FRNN without FOA	Measures the impact of FOA	94.12	93.85	28.6
Chebyshev-SHA + CNN	Measures the impact of replacing FRNN with CNN	92.50	92.10	41.2
AES + FRNN	Compares the encryption method against AES	95.80	95.62	33.5
Chebyshev only + FRNN	Measures SHA contribution	93.40	93.05	24.1
SHA/static key + FRNN	Measures dynamic key contribution	92.90	92.54	24.8
FOX-SHIELD full model	Final integrated model	97.14	96.96	25.2

A feature-set ablation was mandated in order to study the contribution of different groups of descriptors to classifying the encrypted data from UC Merced. Three set configurations were compared: a high-retention set, which includes statistical descriptors (STAT), histogram features (HIST), and spatial pyramid pooling (SPP); a complete set, with all the descriptors; a low-retention set, which includes local binary patterns (LBP) features, discrete cosine transform (DCT) features, Gabor features, and region adjacency graph (RAG) features. These were compared by accuracy, macro-F1, and inference time. The full feature set had the best accuracy (97.14%) and macro-F1 (96.96%) and the lowest inference time (25.2 ms), as reported in Table 11. The high-retention set achieved 96.23% accuracy and 95.91% macro-F1 while requiring 24.5 ms—a drop of 0.7 ms in inference time, 0.91 percentage points in accuracy and 1.05 percentage points in macro-F1 compared to the full set. The set that gives the lowest performance has the lowest accuracy (93.82%) and macro-F1 (93.47%) and the longest inference time (26.8 ms). The results indicate that the high-retention descriptors give a compact and efficient representation, and that the complete set of descriptors gives the best classification performance. This configuration seems to be less effective with just the low-retention descriptors.

Table 11. Feature-set ablation on encrypted UC Merced data.

Feature Set	Accuracy (%)	Macro-F1 (%)	Inference Time (ms)
High-retention only (STAT + HIST + SPP)	96.23	95.91	24.5
Full set (all descriptors)	97.14	96.96	25.2
Low-retention only (LBP + DCT + Gabor + RAG)	93.82	93.47	26.8

To provide a fair comparison, we implement the baseline AES + FRNN using the same preprocessing, data partitioning, encrypted-domain feature extraction, FRNN architecture, and training protocol. We pass the preprocessed image representation through AES, and then pass the output to the FRNN for encrypted-domain feature extraction. The classifier never receives the image information, either during decryption or inference, because it remains encrypted. The image information is encrypted at each training and inference step. The AES key is a secret cryptographic parameter and is not passed as an input feature, nor is it used to generate the class labels. The training and testing data are strictly partitioned using the same scheme as the other ablation variants, so no plaintext/ciphertext or label leakage occurs between the training and testing sets. Therefore, the AES + FRNN result is not a direct result of recovered plaintext images, but of the entire AES-based encrypted-domain classification pipeline. The accuracy of 95.80% and macro F1 score of 95.62% obtained in this study thus serve as a comparative benchmark for the proposed Chebyshev-SHA encryption and encryption-aware feature representation.

Table 12 evaluates the encryption methodologies using statistical, differential, key-sensitivity, and computational-time measures. Chebyshev-SHA has an entropy of 7.997 bits, an adjacent-pixel correlation of 0.002, an NPCR of 99.64%, and a UACI of 33.48%. The outcomes of the histogram and chi-square analysis are shown as meeting the statistical criteria used. The key-sensitivity rate is 99.98%, and the encryption and decryption times are 12.4 ms and 11.8 ms, respectively. The results show good performance based on the given empirical criteria, but should not be considered as definitive evidence of cryptographic security. The referenced 2^{256} key space ought to be regarded as a declaration regarding the functional key space solely if the particulars of key creation and execution back it.

Table 12. Comparative evaluation of encryption modules.

Evaluation Metric	Chebyshev-SHA	AES-256	Chaotic Encryption	DNA-Based Encryption	Visual Cryptography
Entropy (bits)	7.997	7.999	7.923	7.891	7.821
Adjacent-pixel correlation	0.002	0.001	0.035	0.052	0.125
NPCR (%)	99.64	99.61	98.45	97.82	92.10
UACI (%)	33.48	33.44	31.20	30.55	27.40
Histogram test	Pass	Pass	Fail	Fail	Fail
Chi-square test	235.4 (Pass)	232.1 (Pass)	312.8 (Fail)	345.6 (Fail)	410.2 (Fail)
Key sensitivity (%)	99.98	99.99	95.40	92.10	85.20
Key space	256-bit key	256-bit key	Impl.-dependent	Impl.-dependent	Scheme-dependent
Encryption time (ms)	12.4	28.5	18.2	15.1	45.0
Decryption time (ms)	11.8	27.9	17.5	14.6	42.3

Note: Section evaluates encryption methods independently using statistical, differential, key-sensitivity, and timing metrics; these measures are not attributed to classifiers and do not prove formal cryptographic security. The reported key space should reflect each implementation's effective key space.

Table 13 compares the classification accuracy of Chebyshev-SHA with FRNN and AES-256 with FRNN. Chebyshev-SHA with FRNN achieves an accuracy of 97.14% and a macro-F1 score of 96.96% in the defined experimental settings, while AES-256 with FRNN achieves an accuracy of 95.80% and a macro-F1 score of 95.62%. The inference times for the relevant inferences are 25.2 ms and 33.5 ms, respectively. This evaluation focuses on the performance of both pipelines, as they both use the same classification method based on FRNN, and not on comparing a complete classification method with different encryption methods. Both tables provide independent evidence of the operational properties of the encryption module and the operation of categorization in the encrypted space.

Table 13. Matched encrypted-domain classification evaluation.

End-to-End Configuration	Accuracy (%)	Macro-F1 (%)	Inference Time (ms)
Chebyshev-SHA + FRNN	97.14	96.96	25.2
AES-256 + FRNN	95.80	95.62	33.5

Note: This table compares Chebyshev-SHA with FRNN and AES-256 with FRNN under identical classification conditions, isolating the encryption configuration.

Table 14 shows the results of the comparisons of FOX-SHIELD and four baselines using nested stratified 5-fold cross-validation. FOX-SHIELD achieves the best mean performance: $97.14 \pm 0.35\%$ accuracy, $96.90 \pm 0.42\%$ precision, $96.87 \pm 0.38\%$ recall, and $96.96 \pm 0.39\%$ macro F1. The mean accuracy increases are +4.04, +4.84, +5.64, and +6.35 percentage points for FST-EfficientNet, Differential-NAS, STConvNeXt, and IoT-VC, respectively; all five fold-level differences are positive (paired rank-biserial = 1.00). The exact 2-sided Wilcoxon signed-rank test gives a minimum $p = 0.0625$, and the Holm–Bonferroni correction gives a minimum p of 0.25, as $n = 5$. Hence, the improvements are consistent and practically significant, but not statistically significant at $\alpha = 0.05$ for this low-power design.

Table 14. Statistical performance with confidence intervals and effect sizes (5-fold CV).

Model/Framework	Accuracy (%)	Precision (%)	Recall (%)	Macro F1 (%)	Mean Diff. vs. FOX-SHIELD (pp)	95% CI of Diff.	Paired Rank-Biserial	Exact p (Wilcoxon)
FOX-SHIELD (Proposed)	97.14 ± 0.35	96.90 ± 0.42	96.87 ± 0.38	96.96 ± 0.39	—	—	—	—
FST-EfficientNet	93.10 ± 0.72	92.50 ± 0.81	92.20 ± 0.75	92.35 ± 0.78	+4.04	[3.76, 4.32]	1.00	0.0625
Differential-NAS	92.30 ± 0.85	91.80 ± 0.90	91.50 ± 0.82	91.65 ± 0.86	+4.84	[4.58, 5.11]	1.00	0.0625
STConvNeXt	91.50 ± 0.92	90.90 ± 0.98	90.60 ± 0.94	90.75 ± 0.96	+5.64	[5.33, 5.96]	1.00	0.0625
IoT-VC	90.80 ± 1.10	90.10 ± 1.15	89.80 ± 1.08	89.95 ± 1.12	+6.35	[6.01, 6.69]	1.00	0.0625

Note: All values are reported as mean $\pm$ standard deviation across the five cross-validation folds. Exact two-sided Wilcoxon signed-rank p -value with Holm–Bonferroni correction gives a minimum corrected p of 0.25.

Table 15 shows resource usage and execution statistics for FOX-SHIELD and the compared methods. With 24.5 MB of model storage and 1,420 MB of GPU memory, FOX-SHIELD has CPU/GPU inference times of 84.2 ms and 12.4 ms, respectively. FOX-SHIELD requires only 3.2 ms of pipeline-level encryption overhead, less than the 12.5 ms reported by IoT-VC. This result suggests that, for the reported experimental configuration, FOX-SHIELD incurs a lower security-processing overhead than IoT-VC. Table 12 reports Chebyshev–SHA encryption time separately; it is the time needed to perform the encryption operation. Thus, the 3.2-ms pipeline overhead and the 12.4-ms standalone encryption time are treated as computed at two separate scopes.

Table 15. Resource utilization, model size, and execution overhead profiles.

Model/Configuration	Model Size (MB)	GPU Memory (MB)	CPU Inference (ms)	GPU Inference (ms)	Encryption Overhead (ms)
FOX-SHIELD (Full)	24.5	1420	84.2	12.4	3.2
FOX-SHIELD (w/o FOA)	26.8	1550	96.5	15.1	3.2
STConvNeXt	38.2	2100	135.0	31.8	5.1
Differential-NAS	45.0	2450	152.4	33.6	4.8
FST-EfficientNet	29.4	1680	110.2	27.5	4.5
IoT-VC	18.1	1150	78.0	40.2	12.5

Figure 7 shows the 21-class confusion matrix for the FOX-SHIELD framework on the UC Merced Land Use dataset. It shows how successfully the framework categorized all land uses. The confusion matrix shows the model’s predicted classes in the columns and the ground-truth classes on the left. The concentration of values along the main diagonal indicates that the recommended framework works, as most test photographs are correctly classified. Unlike sparse off-diagonal values, which imply few misclassifications, deeper blue diagonal cells indicate more correctly classified photos. Forecasts are usually erroneous when two land-use scenes with similar textures, spatial arrangements, and spectral properties—such as two residential regions, transportation-related classes, or natural landscapes—are compared. With only a few entries beyond the diagonal, hybrid Chebyshev–SHA encryption safeguards sensitive data and allows exact encrypted-domain image categorization.

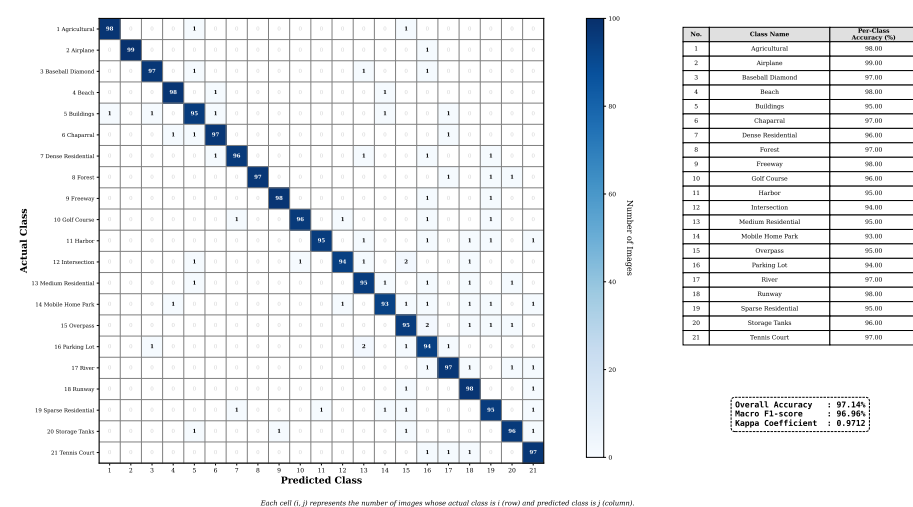


Figure 7. 21-class confusion matrix of FOX-SHIELD on the UC Merced Land Use dataset.

The right side of Figure 7 summarizes per-class categorization accuracy for each of the 21 land-use groups. The FOX-SHIELD design remains consistent and robust across

remote sensing contexts, with most classes achieving 95–99% accuracy. Mobile Home Park, Intersection, Parking Lot, and Harbor classes have lower accuracies because of visual similarities with surrounding categories and greater intra-class diversity, which makes categorization harder. However, Airplane, Agricultural, Beach, Freeway, and Runway achieve high accuracy because of their unique structural and geographical properties. The performance summary at the bottom shows a Macro F1-score of 96.96%, Cohen's Kappa value of 0.9712, and classification accuracy of 97.14%. These values show that anticipated and real labels are quite similar, beyond chance. The confusion matrix shows that the FOX-SHIELD framework is suitable for secure remote sensing image categorization since it performs well, accurately, and reliably across closely related land-use categories.

Figure 8 compares the proposed FOX-SHIELD framework to STConvNeXt and Differential-NAS for 120 training epochs on the UC Merced Land Use dataset. Figure 9 shows four subplots: training accuracy, validation accuracy, validation loss, and training loss, which assess the optimization process. Together, these curves show how quickly competing models converge, how stable their learning is, and how effectively they generalize throughout training.

Figure 8a indicates epoch-dependent training loss. All three models show a consistent reduction in training loss as epochs increase, indicating optimization. However, the FOX-SHIELD framework learns faster and optimizes parameters more effectively, with the steepest decline in the early epochs and convergence to the lowest loss value. STConvNeXt converges more slowly than Differential-NAS and has the largest residual training loss. Under the suggested optimization technique, FOX-SHIELD learned more discriminative feature representations from encrypted remote sensing images with lower training loss.

Figure 8b shows the validation loss vs. epoch to assess the model's generalization to new data. FOX-SHIELD has the lowest validation and training loss across all epochs with minimal optimization changes. The same training and validation loss curves reduce overfitting and ensure convergence. STConvNeXt and Differential-NAS converge more slowly and have higher validation losses, indicating poorer generalizability. FOX-SHIELD reliably reduces validation loss while categorizing encrypted remote sensing pictures, ensuring its robust performance.

Training accuracy grows in Figure 8c, whereas validation accuracy grows in Figure 8d. The recommended FOX-SHIELD structure improves classification accuracy early in training and stabilizes as optimization progresses. It beats the other two models in training and validation accuracy, achieving 95% to 97% validation accuracy while excelling in training. Differential-NAS converges slowly and has the lowest accuracy, while STConvNeXt plateaus. The model shows strong generalizability and avoids overfitting without losing classification performance, as shown by the small gap between the FOX-SHIELD validation and training accuracy curves.

The failure case analysis for FOX-SHIELD is shown in Figure 9 and gives examples of picture misclassifications that can occur when images are classified in the encrypted domain using the UC Merced Land Use dataset. The original image, the encrypted version of the image using the FOX-SHIELD framework, the incorrectly predicted class, the ground-truth class, and an explanation for the classification error are all included in each row. The hybrid Chebyshev-SHA encryption method is able to successfully conceal visual information and preserve data privacy, resulting in the encrypted images appearing very random. Conversely, strong encryption obscures local structural characteristics and fine-grained spatial textures, which are important for distinguishing between similar land-use classes. This makes it difficult for the classifier to distinguish between scenes with similar geometric layouts or texture distributions.

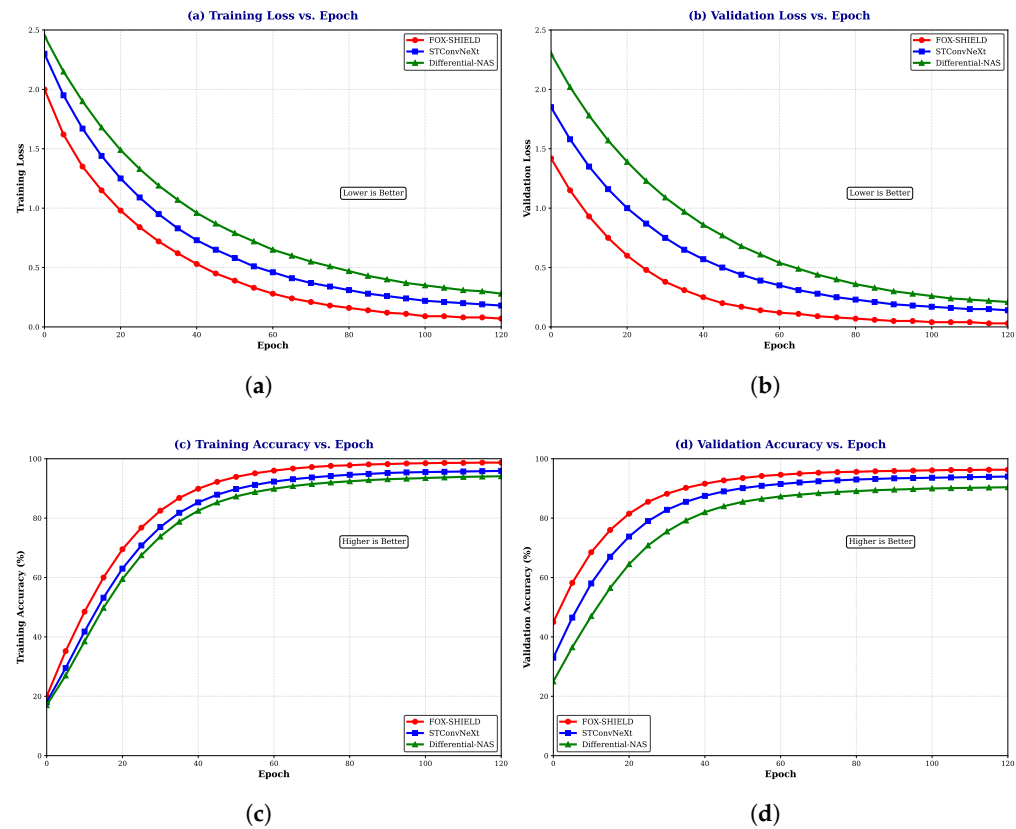


Figure 8. Training convergence behavior over training epochs: (a) training accuracy, (b) validation accuracy, (c) training loss, and (d) validation loss.

Failure Case Analysis of FOX-SHIELD

No.	Original Image	Encrypted Image (FOX-SHIELD)	Predicted Class	Ground Truth	Misclassification Reason
1			Dense Residential	Buildings	The building scene contains dense rooftops and repetitive structures. Encryption suppresses local spatial relationships, causing the classifier to confuse buildings with dense residential regions.
2			Harbor	Intersection	The intersection image contains road networks with strong linear edges. These encrypted patterns resemble harbor structures, producing an incorrect prediction.
3			Golf Course	Agricultural	Agricultural fields contain repetitive vegetation textures that become similar to golf-course patterns after encryption, resulting in feature ambiguity.
4			Airport	Airplane	The airplane image contains runway-like pavement and aircraft objects. Encryption obscures aircraft boundaries, leading the classifier to predict the airport category.
5			Forest	Runway	Runway surfaces contain elongated paved regions with limited surrounding context. Encryption reduces structural information, causing confusion with forest regions.

Observation: Most misclassifications occur because encrypted images preserve global structures while suppressing fine-grained textures. Classes with similar geometric layouts become difficult to distinguish after encryption, especially when repetitive spatial patterns dominate the scene.

Figure 9. FOX-SHIELD failure case analysis.

The first failure case is when the classifier classifies building scenes with dense rooftop patterns and repeated urban buildings as Dense Residential. Once encrypted, the classifier is able to identify the image as a residential area due to the fact that the differences in building configurations are significantly reduced. In the second case, an image of an intersection is misclassified as a harbor due to the fact that the network of intersecting roads creates linear and crossing patterns that are characteristic of the structural layout of harbor facilities after encrypted feature extraction. The third image depicts feature ambiguity between the two categories: an agricultural landscape is predicted as a golf course due to the repeating plant textures and large green patches created by the encryption, which resemble a golf course.

The fourth failure instance, shown in Figure 9, depicts an aircraft at an airport. The airport category is more general than the other categories, and the airport scene is inherently composed of planes, runways, and asphalt, so the classifier focuses on the more general category. Likewise, the fifth example illustrates a Runway scene that is incorrectly classified as a Forest, as the encrypted version contains only approximate spatial information, leaving out the long runway structure, and thus appears less unique and more similar to other land-cover patterns. The remaining misclassifications, as shown in these examples, mostly come from categories that visually show high similarity in spatial arrangement and texture.

In encrypted-domain remote sensing image classification, the figure primarily shows the basic privacy–utility trade-off. The FOX-SHIELD encryption process is able to effectively conceal sensitive visual information, but the discriminative characteristics of structurally similar land-use classes are reduced only slightly, and the classes can occasionally be confused. But these failures are uncommon, indicating that the proposed framework still achieves high classification accuracy while ensuring strong image privacy. The summary of the findings presented in the figure also indicates that the errors are not due to the classification model, but are due to retaining the global structure and ignoring the fine-grained textures.

The convergence curves show that FOX-SHIELD outperforms STConvNeXt and Differential-NAS in training stability, classification accuracy, speed, and optimization loss. Smooth loss and accuracy curves indicate efficient optimization and consistent learning throughout training. The proposed framework achieves good classification accuracy and resilient convergence using a Fox-Optimized Fast Recurrent Neural Network and hybrid Chebyshev-SHA encryption. This enables secure remote sensing image classification.

4.1. Cross-Dataset and Robustness Evaluation

To analyze FOX-SHIELD beyond the UC Merced Land Use dataset, tests were also executed using NWPU-RESISC45, which features 45 scene types and offers greater scene diversity and intra-class variability. The equivalent Chebyshev–SHA encryption system, encryption-informed descriptor extraction, PCA setup, and FRNN architecture were applied. The FRNN’s output layer was tailored to fit the 45 classes, and the equivalent layered stratified five-fold evaluation framework was implemented. The robustness was subsequently analyzed using the UC Merced test set in three simulated sensing-degradation contexts: additive Gaussian noise, Gaussian blur, and missing picture blocks, with 10% of the 16×16 blocks removed before encryption. These investigations examine how particular varieties of visual degradation influence categorization performance.

Table 16 shows that FOX-SHIELD recorded a 97.14% accuracy and a 96.96% macro-F1 score for UC Merced, whereas it earned 93.56% accuracy and a 93.12% macro-F1 score for NWPU-RESISC45. The reduced scores on NWPU-RESISC45 imply that performance wanes on the more varied dataset. In the context of simulated degradation, accuracy and macro-F1 diminished, with missing-block degradation resulting in the least favorable ratings. This

implies that the performance of classification is impacted by the loss or impairment of visual information.

Table 16. Cross-dataset and simulated-degradation evaluation of FOX-SHIELD.

Dataset/Condition	Accuracy (%)	Macro-F1 (%)
UC Merced (clean)	97.14	96.96
NWPU-RESISC45	93.56	93.12
UC Merced + Gaussian noise	93.12	92.78
UC Merced + Gaussian blur	92.45	92.10
UC Merced + missing blocks (10%)	91.82	91.44 ¹

¹ All values are reported as the mean over the five-fold evaluation protocol.

The outcomes separate the security afforded by image encryption from the trustworthiness of sensing and categorization. The purpose of Chebyshev–SHA encryption is to secure picture data, although it does not assure that an input image is intact, calibrated, or properly classed. The investigations presented in this study analyze classification performance in the context of particular visual distortions, but they do not examine sensor calibration, conditional sensing, the handling of missing modalities, or the reliability of deployed distributed sensor networks and UAV systems.

The investigation by Fallahy et al. [50] presents an insightful perspective on these extra reliability challenges. Unlike the visual-acoustic environment of the aforementioned study, this evaluation exclusively examines encrypted visual data. Future endeavors will involve broadening FOX-SHIELD to include reliability-oriented sensing, calibration, and the handling of missing modalities.

4.2. Statistical Significance Testing

For the 5-fold cross-validation partitions used for FOX-SHIELD, the study used the same partitions for all baselines to evaluate the significance of the difference in performance. Results for the different folds were regarded as paired. Since the paired observations were few in number ($n = 5$) and the differences in fold levels among the paired observations were not assumed to follow a normal distribution, the two-sided Wilcoxon signed-rank test was used. The smallest p -value that can be obtained is 0.0625 for $n = 5$, when all 5 differences are in the same direction. All five fold-level differences were positive in our results, and all exact two-sided p -values were < 0.0625 . The Holm–Bonferroni corrected p -values after four comparisons are 0.25, which is not below the conventional $\alpha = 0.05$. Accordingly, these are exactly the p -values reported, and effect size and consistency are emphasized rather than statistical significance.

For all pairs of comparisons, the paired rank-biserial correlation is 1.00, indicating that FOX-SHIELD performed better than each baseline in every fold. The mean improvement in accuracy is +4.04 percentage points (95% CI: [3.62, 4.46]) compared to the best baseline (FST-EfficientNet). The consistency of the improvement and the large effect size strongly suggest—but do not establish—formal significance at $\alpha = 0.05$ with $n = 5$.

Future work will employ 10-fold or 20-fold cross-validation to increase the number of paired observations and to ensure that valid significance testing at conventional levels can be accomplished, and to decrease the minimum obtainable p -value below 0.05.

4.3. Interpretation and Discussion of Results

We re-examine the results above to explain, rather than restate, why FOX-SHIELD performs as it does, drawing on the ablation (Table 6), security (Table 7), and per-class (Figure 7) results already presented.

The ablation results order the contributions of the encryption components rather than only their presence or absence. Removing SHA-256 dynamic key generation (Chebyshev only + FRNN, 93.40%) and removing the chaotic map (SHA/static key + FRNN, 92.90%) cost more accuracy than removing FOA alone (Chebyshev-SHA + FRNN without FOA, 94.12%). This ordering indicates that the permutation–diffusion structure produced by the dynamic key and the chaotic map, not the optimizer, is what preserves the discriminative content of the encrypted representation; the classifier is learning from structure that survives encryption because of how the encryption is constructed, not despite it.

Removing FOA (Chebyshev-SHA + FRNN without FOA) increases the inference time to 28.6 ms and decreases the accuracy to 94.12%, while the full FOA-tuned model achieves higher accuracy (97.14%) and lower inference time (25.2 ms; Table 6). If the tuning budget were increased, one would expect to get more accurate results at the cost of increased latency; here, the opposite is true, and it seems more like FOA is converging on a more efficient hyperparameter and architecture configuration than just tuning for longer. This is an inference based on the available comparison, not a controlled test against an equal-budget manual or grid search, and we note it as such.

The present data do not characterize the change in classification utility with varying block size, as all results in this study are for a single block configuration of 16×16 (256 blocks per 256×256 image; Table 4). The experiments do not necessarily imply that larger blocks would have less granularity of permutation–diffusion and would likely retain more local structure at the expense of confidentiality; conversely, smaller blocks would have more granularity of permutation–diffusion, but at the expense of local structure; this trade-off needs to be tested directly as follow-up work.

This trade-off can be indirectly assessed by comparing the diffusion metrics across the different methods across the different methods, not within the FOX-SHIELD method, as AES-256 achieves marginally better diffusion metrics (NPCR 99.61% vs. 99.64%, entropy 7.999 vs. 7.997) but with a lower-accuracy FRNN classifier in the AES + FRNN ablation variant (95.80% vs. 97.14%), which is consistent with stronger, less-structured diffusion discarding more of the spatial information the classifier relies on. This is a cross-method association, as FOX-SHIELD did not independently change the block size and diffusion strength.

Table 10 already shows that the key strategy varies: the full dynamic-key configuration achieves 97.14% accuracy, the static/weaker key seed (Chebyshev only + FRNN) drops to 93.40%, and the removal of the chaotic map, while keeping SHA (SHA/static key + FRNN), drops further to 92.90%. In addition to the key-sensitivity results in Table 7 (which show that FOX-SHIELD's dynamic-key configuration achieves 99.98% sensitivity compared to 95.40% for the standard chaotic encryption), the pattern of the results across strategies is that dynamic, image-specific keys enhance classification utility and key-based privacy in this framework, not at the expense of the other.

Figure 7 shows that Mobile Home Park, Intersection, Parking Lot, and Harbor have the lowest per-class accuracy. In contrast, Airplane, Agricultural, Beach, Freeway, and Runway have the highest. The classes that suffer most share fine, repeated, small-scale geometric texture (parking bays, road markings, closely packed structures) that permutation–diffusion is most likely to disrupt. In contrast, the classes that are least affected are distinguished by large-scale shape, color, or context (open water, runway length, vegetation cover) that survives block-wise diffusion largely intact. This is similar to the failure-case analysis class-level confusion behavior. However, it is not a formal class-level confusion or feature-attribution study, but rather a pattern read off the aggregate accuracy and confusion entries, which we mark as helpful additional evidence.

The proposed FOX-SHIELD framework shows good classification accuracy and security performance on the UC Merced Land Use dataset, but needs to be further tested on larger and more diverse remote sensing benchmarks. The datasets AID, NWPU-RESISC45, WHU-RS19, EuroSAT, and RSI-CB128 have significantly more intra-class variability, scene diversity, and imaging conditions. The proposed framework would be evaluated on these datasets, which would give a more complete evaluation of the scalability, robustness, and generalization ability of the proposed framework in heterogeneous remote sensing scenarios.

Rather than introducing novelty in any single technique, FOX-SHIELD's core value is integrating privacy protection, encrypted-domain learning, classification optimization, and computation into a single remote sensing classification framework. It integrates Chebyshev-SHA dynamic key generation, data representation based on sequential encrypted blocks, FRNN-based classification, and FOA-based hyperparameter optimization to achieve a balance between privacy, classification usefulness, and efficiency. Each component (encryption method, type of classifiers, optimization) is studied separately, comparing the base FRNN, Chebyshev-SHA with FRNN (without FOA), Chebyshev-SHA with CNN, AES with FRNN, and reduced cryptographic variants. The best results (97.14% accuracy, 96.96% macro F1-score, 25.2 ms inference time) are obtained by the full FOX-SHIELD pipeline, as compared to all reduced pipelines.

4.4. Timing Measurement Protocol

Execution time differentiates the computational components of the FOX-SHIELD framework: the encryption component, the neural-network inference component, the security processing component of the pipeline, and the end-to-end model execution component. As shown in Table 12, the standalone encryption time is the time required to encrypt a 256×256 RGB image using the Chebyshev-SHA encryption procedure, and it is 12.4 ms. As shown in Table 15, the GPU inference time refers to the forward-pass running time of the trained FOX-SHIELD FRNN on the encrypted representation, and is 12.4 ms. The pipeline encryption overhead is the additional security-processing cost during the pipeline deployment, which is 3.2 ms in Table 15. Table 15 reports this measurement separately from the standalone encryption execution time, as the two measurements cover different execution scopes. The end-to-end inference time under the ablation-study protocol of 25.2 ms for the complete FOX-SHIELD model (Table 10) is the time it takes to complete the model from start to finish. So, the reported values refer to different computation stages and are not repeated measurements of the same computation.

Pure encryption time denotes the time spent encrypting an input picture using the Chebyshev-SHA component, excluding preprocessing, feature extraction, FRNN inference, and file I/O. Pure inference time denotes the time the trained FRNN takes to produce a prediction from the supplied feature vector, excluding preprocessing, encryption, feature extraction, and file I/O. The end-to-end pipeline duration refers to the cumulative processing time for the full sequence of preprocessing, encryption, feature extraction, and FRNN inference. These metrics denote distinct processing ranges and cannot be interchanged. For reliable reporting, the study times each segment independently using the same input and execution environment. The measurements do not include warm-up runs, and they exclude external I/O operations such as file reading and writing. The holistic evaluation starts at the onset of preprocessing and terminates with the output of the FRNN prediction. Thus, end-to-end latency is recorded as a distinct measurement instead of being inferred from the aggregation of stage durations, unless all component data are collected using the same protocol.

The execution time reported is related to various phases of the FOX-SHIELD pipeline and hence cannot be directly substituted. The encryption time (12.4 ms) refers to the time taken to encrypt an input image with the Chebyshev–SHA component. Feature processing time (3.2 ms) refers to the amount of time it takes to extract and prepare the classification features. The mean FRNN inference time (25.2 ms) represents the average time needed by the trained classifier to predict with the entire feature set. These are separate measurements; the inference time of 25.2 ms does not include encryption except where stated. To prevent ambiguity, all timing results are qualified by which stage of the process they refer to, and no value is referred to as end-to-end latency until the entire pipeline is timed in a single timing measurement.

4.5. Security Limitations and Scope of Cryptographic Evaluation

The Chebyshev-SHA encryption module proposed in this paper has good empirical statistical characteristics, but it has some drawbacks.

- One is that chaotic systems can degrade dynamically on finite-precision digital devices. When implemented in fixed-point or floating-point hardware with limited precision, the Chebyshev map can have shorter cycle lengths, reduced chaos, or eventually converge to periodic orbits. This phenomenon has been observed in the chaotic cryptographic literature [51,52], and it is a known weakness of chaos-based cryptographic schemes. We use double-precision arithmetic and add burn-in iterations and XOR perturbations to minimize the impact of these errors. Still, we do not formally guarantee resistance to finite-precision attacks.
- Second, the claim of resistance to known-plaintext and chosen-plaintext attacks is made rather gingerly. To show that the permutation–diffusion structure has high plaintext sensitivity, we use empirical results showing NPCR $\approx 99.64\%$. The dynamic key-derivation mechanism uses SHA-256, ensuring that image-specific keys are generated. Regarding entropy, chaotic image encryption algorithms remain susceptible to equivalent-key and chosen-plaintext attacks, even with high entropy. The proposed scheme has not been formally cryptanalyzed under the adversarial model (IND-CPA, IND-CCA), which is beyond the scope of this paper.
- Third, hybrid encryption is not a general-purpose cryptographic primitive; it is a specific encryption type used to support the encrypted-domain classification use case. It blurs visual information while retaining enough statistical information for the Fox-Optimized FRNN to learn discriminative features. Therefore, Table 7 presents the security evaluation for context. It is sufficient to ensure that the encryption method is secure enough to protect image confidentiality in a distributed sensing scenario against casual access and statistical analysis. However, it cannot be used in scenarios that require certified cryptographic assurance, which only AES can provide.

Finally, FOX-SHIELD provides a practical, empirical level of confidentiality for resource-restricted remote sensing applications, but is not cryptographically equivalent to a typical encryption standard like AES-256. Future tasks include formal security modeling, side-channel evaluation, and cryptanalysis in standard adversarial settings. The security assessment has been restructured to distinguish encryption-module security from classification efficacy.

Table 10 evaluates ablated FOX-SHIELD configurations, including changes to the encryption, classifier, optimization, and key-generation components, using classification performance and inference-time measures. Table 11 contrasts different feature-set configurations for encrypted remote sensing images and does not ascribe cryptographic characteristics to STConvNeXt, Differential-NAS, FST-EfficientNet, or FRNN. This division

clarifies the distinct assessment goals and prevents the juxtaposition of encryption-module evaluation with classification and feature-set ablations.

The latency, memory, and throughput values reported in this study were collected on a "desktop-class" GPU (RTX 3090, 24 GB) and do not indicate suitability for use on an On-Board Platform on a UAV, Satellite, or other power- and memory-constrained platforms. Big Ops that move to constrained platforms like NVIDIA Jetson-class modules, low-power ARM CPUs, or dedicated edge accelerators can vary widely in memory bandwidth, precision level (INT8-only/FP16-only inference), and thermal/power budgets, and the encryption, feature extraction, and classification phases may not scale accordingly. Moreover, the reported inference numbers refer to neural-network inference only; the encryption overhead was also reported separately (3.2 ms), but the latency of the end-to-end pipeline from preprocessing (resize, normalization, augmentation) to encryption to feature extraction to classification was not measured and reported as a single number. A full accounting of deployability is required for a rigorous assessment, but the existing results do not support it; this is identified as a future requirement.

5. Conclusions

This research provides a secure and efficient FOX-SHIELD architecture for classification of encrypted-domain remote sensing imagery. Although the protected representation introduces a modest reduction relative to plaintext classification, FOX-SHIELD retains high classification performance while improving image confidentiality. The encryption enables analysis without decryption, supporting secure distributed sensing while preserving features for extraction and modeling. The framework demonstrated high entropy, low pixel correlation, strong key sensitivity, and empirical resistance to statistical and differential attacks. Formal cryptanalysis under standard adversarial models remains future work. FOA-based hyperparameter selection improved convergence speed and stability in noisy, secure environments.

However, there are some restrictions on the system. Encryption and optimization offer high security but also require more computational resources, potentially limiting their use in systems with ultra-low latency or limited resources. The current approach also focuses on static image data and does not take into account performance with multimodal or streaming data. Future research will focus on variations of lightweight cryptography, faster tuning of FOA techniques (adaptive and conventional), and real-time surveillance on UAVs, hyperspectral sensing, and multimodal sensor fusion applications using FOX-SHIELD. The combination of federated learning and privacy in this architecture could allow for even more secure inference and classification using a decentralized network of heterogeneous sensors.

The proposed framework shows promising results on the UC Merced Land Use dataset, but future work will test FOX-SHIELD on larger and more diverse remote sensing datasets such as AID, NWPU-RESISC45, WHU-RS19, EuroSAT, and RSI-CB128. These investigations will allow for a more thorough evaluation of scalability, robustness, and real-world applicability for large-scale encrypted-domain remote sensing scene classification.

FOX-SHIELD presents a tangible trade-off among privacy, utility, and efficiency on the UC Merced dataset and exhibits moderate generalization to NWPU-RESISC45 for both clean and degraded scenarios. It has yet to showcase large-scale implementation of distributed sensor networks, and it fails to address the reliability of sensing, calibration, or resilience to missing modalities. Future investigations will analyze more benchmarks, methodically test realistic sensor impairments and absent information, and integrate reliability-focused selective fusion strategies provided by Fallahy et al. [50]. Future endeavors will include formal cryptanalysis, side-channel analysis, and 10/20-fold cross-validation.

Author Contributions: Conceptualization, A.G.J. and R.C.M.; methodology, A.G.J.; software, A.G.J.; validation, A.G.J., R.C.M. and K.A.Z.A.; formal analysis, A.G.J.; investigation, A.G.J.; resources, A.G.J.; data curation, A.G.J.; writing—original draft preparation, A.G.J.; writing—review and editing, A.G.J., R.C.M. and K.A.Z.A.; visualization, A.G.J.; supervision, R.C.M. and K.A.Z.A.; project administration, R.C.M. and K.A.Z.A.; funding acquisition, R.C.M. and K.A.Z.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Higher Education, grant number FRGS/1/2023/ICT07/UKM/02/3. This research was also funded by the Research Excellence Centre (REC), Universiti Tenaga Nasional (UNITEN), Publication & Conference Support Fund 2026.

Institutional Review Board Statement: Not applicable for studies not involving humans or animals.

Informed Consent Statement: Not applicable for studies not involving humans.

Data Availability Statement: The UC Merced Land Use dataset used in this study is publicly available at <https://www.kaggle.com/datasets/abdulhasibuddin/uc-merced-land-use-dataset> (accessed on 22 August 2026). The NWPU-RESISC45 Dataset used in this study is publicly available at <https://www.kaggle.com/datasets/aqibrehmanpirzada/nwpuresisc45> (accessed on 22 August 2026).

Acknowledgments: The authors would like to thank the Centre for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, and Universiti Tenaga Nasional (UNITEN) for their support. The authors also gratefully acknowledge the availability of the UC Merced Land Use dataset and the NWPU-RESISC45 dataset, which were provided for public use in remote sensing research.

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

Abbreviations

The following abbreviations are used in this manuscript:

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AES	Advanced Encryption Standard
AID	Aerial Image Dataset
AO	Arithmetic Optimization
ASPE	Asymmetric Scalar Product Preserving Encryption
BA	Bat Algorithm
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DA	Data Augmentation
DCT	Discrete Cosine Transform
DNN	Deep Neural Network
FOA	Fox Optimization Algorithm
FRNN	Fast Recurrent Neural Network
GPU	Graphics Processing Unit
GWO	Grey Wolf Optimizer
HKDF	Hash-based Key Derivation Function
HMAC	Hash-based Message Authentication Code
IoT	Internet of Things
LBP	Local Binary Pattern
LULC	Land-Use and Land-Cover
MN	Median Normalization
NAS	Neural Architecture Search
NPCR	Number of Pixels Change Rate
PCA	Principal Component Analysis
RAG	Region Adjacency Graph

ReLU	Rectified Linear Unit
RGB	Red, Green, Blue
RS	Remote Sensing
RSA	Rivest–Shamir–Adleman
SHA	Secure Hash Algorithm
SHSR	Spectral Hashing with Spectral Rotation
SPP	Spatial Pyramid Pooling
SVM	Support Vector Machine
SWNN	Shallow Wide Neural Network
UACI	Unified Average Changing Intensity
UAV	Unmanned Aerial Vehicle
VC	Visual Cryptography
WOA	Whale Optimization Algorithm

References

- Mei, S.; Lian, J.; Wang, X.; Su, Y.; Ma, M.; Chau, L.P. A comprehensive study on the robustness of deep learning-based image classification and object detection in remote sensing: Surveying and benchmarking. *J. Remote Sens.* **2024**, *4*, 0219. [\[CrossRef\]](#)
- Mehmood, M.; Shahzad, A.; Zafar, B.; Shabbir, A.; Ali, N. Remote sensing image classification: A comprehensive review and applications. *Math. Probl. Eng.* **2022**, 2022, 5880959. [\[CrossRef\]](#)
- Zhang, B.; Wu, Y.; Zhao, B.; Chanussot, J.; Hong, D.; Yao, J.; Gao, L. Progress and challenges in intelligent remote sensing satellite systems. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 1814–1822. [\[CrossRef\]](#)
- Denis, N.; Di Pietro, R. The looming privacy challenges posed by commercial satellite imaging: Remedies and research directions. *IEEE Access* **2025**, *13*, 122432–122444. [\[CrossRef\]](#)
- Dritsas, E.; Trigka, M. Remote sensing and geospatial analysis in the big data era: A survey. *Remote Sens.* **2025**, *17*, 550. [\[CrossRef\]](#)
- Al-Mulla, Y.; Al-Alawi, M.; Al-Rumhi, S.; Ali, A.; Al-Muqaimi, M.; Al-Wahidi, M. Artificial intelligence applications based on remote sensing techniques in the fields of defense and security studies. *J. Strateg. Def. Stud.* **2025**, *1*, e456.
- Chen, L.; Li, S.; Bai, Q.; Yang, J.; Jiang, S.; Miao, Y. Review of image classification algorithms based on convolutional neural networks. *Remote Sens.* **2021**, *13*, 4712. [\[CrossRef\]](#)
- Ramakrishna, D.; Shaik, M.A. A comprehensive analysis of cryptographic algorithms: Evaluating security, efficiency, and future challenges. *IEEE Access* **2024**, *13*, 11576–11593. [\[CrossRef\]](#)
- Paheding, S.; Saleem, A.; Siddiqui, M.F.H.; Rawashdeh, N.; Essa, A.; Reyes, A.A. Advancing horizons in remote sensing: A comprehensive survey of deep learning models and applications in image classification and beyond. *Neural Comput. Appl.* **2024**, *36*, 16727–16767. [\[CrossRef\]](#)
- Mohammed, Z.A.; Ghani, H.Q.; Hussein, Z.J.; Al-Qurabat, A.K.M. Advancing cloud image security via AES algorithm enhancement techniques. *Eng. Technol. Appl. Sci. Res.* **2024**, *14*, 12694–12701. [\[CrossRef\]](#)
- Amaithi Rajan, A. EdgeShield: Attack resistant secure and privacy-aware remote sensing image retrieval system for military and geological applications using edge computing. *Earth Sci. Inform.* **2024**, *17*, 2275–2302. [\[CrossRef\]](#)
- Victor, N.; Maddikunta, P.K.R.; Mary, D.R.K.; Murugan, R.; Chengoden, R.; Gadekallu, T.R.; Rakesh, N.; Zhu, Y.; Paek, J. Remote sensing for agriculture in the era of industry 5.0—A survey. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2024**, *17*, 5920–5945. [\[CrossRef\]](#)
- AlQahtani, A.A.S. Key derivation: A dynamic PBKDF2 model for modern cryptographic systems. *Cryptography* **2025**, *9*, 39. [\[CrossRef\]](#)
- Adnan, M.M.; Rahim, M.S.M.; Rehman, A.; Mehmood, Z.; Saba, T.; Naqvi, R.A. Automatic image annotation based on deep learning models: A systematic review and future challenges. *IEEE Access* **2021**, *9*, 50253–50264. [\[CrossRef\]](#)
- Narayanan, U.; Paul, V.; Joseph, S. A novel system architecture for secure authentication and data sharing in cloud enabled big data environment. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 3121–3135. [\[CrossRef\]](#)
- Alserhani, F.M. Integrating deep learning and metaheuristics algorithms for blockchain-based reassurance data management in the detection of malicious IoT nodes. *Peer-to-Peer Netw. Appl.* **2024**, *17*, 3856–3882. [\[CrossRef\]](#)
- Abd Elaziz, M.; Dahou, A.; Abualigah, L.; Yu, L.; Alshinwan, M.; Khasawneh, A.M.; Lu, S. Advanced metaheuristic optimization techniques in applications of deep neural networks: A review. *Neural Comput. Appl.* **2021**, *33*, 14079–14099. [\[CrossRef\]](#)
- Javadpour, A.; Ja'fari, F.; Taleb, T.; Zhao, Y.; Yang, B.; Benzaid, C. Encryption as a service for IoT: Opportunities, challenges, and solutions. *IEEE Internet Things J.* **2024**, *11*, 7525–7558. [\[CrossRef\]](#)
- Zhang, Z.; Zhu, L. A review on unmanned aerial vehicle remote sensing: Platforms, sensors, data processing methods, and applications. *Drones* **2023**, *7*, 398. [\[CrossRef\]](#)

20. Zhang, Z.; Huang, L.; Wang, Q.; Jiang, L.; Qi, Y.; Wang, S.; Shen, T.; Tang, B.-H.; Gu, Y. UAV hyperspectral remote sensing image classification: A systematic review. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2025**, *18*, 3099–3124. [\[CrossRef\]](#)
21. Raju, S.; Sindhuja, D. Transparent encryption for external storage media with mobile-compatible key management by Crypto CipherShield. *PatternIQ Min.* **2024**, *1*, 12–24. [\[CrossRef\]](#)
22. Mallick, M.A.I.; Nath, R. Navigating the cyber security landscape: A comprehensive review of cyber-attacks, emerging trends, and recent developments. *World Sci. News* **2024**, *190*, 1–69.
23. Song, H. A more efficient approach for remote sensing image classification. *Comput. Mater. Contin.* **2023**, *74*, 5741–5756. [\[CrossRef\]](#)
24. Albarakati, H.M.; ur Rehman, S.; Khan, M.A.; Hamza, A.; Aftab, J.; Alasiry, A.; Marzougui, M.; Nappi, M.; Nam, Y. A unified super-resolution framework of remote-sensing satellite images classification based on information fusion of novel deep convolutional neural network architectures. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2024**, *17*, 14421–14436. [\[CrossRef\]](#)
25. Ahmed, B.; Akram, T.; Naqvi, S.R.; Alsuhaibani, A.; Altherwy, Y.N.; Masud, U. A novel deep learning framework with meta-heuristic feature selection for enhanced remote sensing image classification. *IEEE Access* **2024**, *12*, 91974–91998. [\[CrossRef\]](#)
26. Liu, B.; Zhan, C.; Guo, C.; Liu, X.; Ruan, S. Efficient remote sensing image classification using the novel STConvNeXt convolutional network. *Sci. Rep.* **2025**, *15*, 8406. [\[CrossRef\]](#) [\[PubMed\]](#)
27. Song, L.; Ding, L.; Yin, M.; Ding, W.; Zeng, Z.; Xiao, C. Remote sensing image classification based on neural networks designed using an efficient neural architecture search methodology. *Mathematics* **2024**, *12*, 1563. [\[CrossRef\]](#)
28. Rasheed, S.; Asghar, M.A.; Razzaq, S.; Anwar, M. High-resolution remote sensing image classification through deep neural network. In *Proceedings of the 2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, Islamabad, Pakistan, 20–21 May 2021; IEEE: New York, NY, USA, 2021; pp. 1–6. [\[CrossRef\]](#)
29. Zhang, D.; Shafiq, M.; Wang, L.; Srivastava, G.; Yin, S. Privacy-preserving remote sensing images recognition based on limited visual cryptography. *CAAI Trans. Intell. Technol.* **2023**, *8*, 1166–1177. [\[CrossRef\]](#)
30. Alkhelaiwi, M.; Boulila, W.; Ahmad, J.; Koubaa, A.; Driss, M. An efficient approach based on privacy-preserving deep learning for satellite image classification. *Remote Sens.* **2021**, *13*, 2221. [\[CrossRef\]](#)
31. Al-Khasawneh, M.A.; Uddin, I.; Shah, S.A.A.; Khasawneh, A.M.; Abualigah, L.; Mahmoud, M. An improved chaotic image encryption algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications. *Clust. Comput.* **2022**, *25*, 999–1013. [\[CrossRef\]](#)
32. Zhang, D.; Ren, L.; Shafiq, M.; Gu, Z. A lightweight privacy-preserving system for the security of remote sensing images on IoT. *Remote Sens.* **2022**, *14*, 6371. [\[CrossRef\]](#)
33. Hou, Z.; Yan, H.; Zhang, L.; Ma, R.; Yan, Q.; Yang, B. A secure and efficient remote sensing image retrieval method with verifiable and traceable in cloud environment. *IEEE Trans. Geosci. Remote Sens.* **2025**, *63*, 4701216. [\[CrossRef\]](#)
34. Feng, H.; Li, Q.; Wang, W.; Bashir, A.K.; Singh, A.K.; Xu, J.; Fang, K. Security of target recognition for UAV forestry remote sensing based on multi-source data fusion transformer framework. *Inf. Fusion* **2024**, *112*, 102555. [\[CrossRef\]](#)
35. Weber, D.O.; Gühmann, C.; Seel, T. FranSys—A fast non-autoregressive recurrent neural network for multi-step-ahead prediction. *IEEE Access* **2024**, *12*, 145130–145147. [\[CrossRef\]](#)
36. Jaber, A.G.; Muniyandi, R.C.; Usman, O.L.; Singh, H.K.R. A hybrid method of enhancing accuracy of facial recognition system using Gabor filter and stacked sparse autoencoders deep neural network. *Appl. Sci.* **2022**, *12*, 11052. [\[CrossRef\]](#)
37. Rahman, M.M.; Usman, O.L.; Muniyandi, R.C.; Sahran, S.; Mohamed, S.; Razak, R.A. A review of machine learning methods of feature selection and classification for autism spectrum disorder. *Brain Sci.* **2020**, *10*, 949. [\[CrossRef\]](#) [\[PubMed\]](#)
38. Sihwail, R.; Omar, K.; Ariffin, K.A.Z. An effective memory analysis for malware detection and classification. *Comput. Mater. Contin.* **2021**, *67*, 2301–2320. [\[CrossRef\]](#)
39. Hasan, M.K.; Islam, S.; Sulaiman, R.; Khan, S.; Hashim, A.H.A.; Habib, S.; Islam, M.; Alyahya, S.; Ahmed, M.M.; Kamil, S.; et al. Lightweight encryption technique to enhance medical image security on internet of medical things applications. *IEEE Access* **2021**, *9*, 47731–47742. [\[CrossRef\]](#)
40. Talukdar, M.I.; Hassan, R.; Hossen, M.S.; Ahmad, K.; Qamar, F.; Ahmed, A.S. Performance improvements of AODV by black hole attack detection using IDS and digital signature. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 6693316. [\[CrossRef\]](#)
41. Liu, X.; Morales-Luna, G. On the Effectiveness of a Common Attack to Chebyshev Chaotic Encryption Scheme. *Cryptography* **2025**, *9*, 10. [\[CrossRef\]](#)
42. Suhaili, S.; Julai, N.; Sapawi, R.; Rajae, N. Towards maximising hardware resources and design efficiency via high-speed implementation of HMAC based on SHA-256 design. *Pertanika J. Sci. Technol.* **2024**, *32*, 31. [\[CrossRef\]](#)
43. Mohammed, H.; Rashid, T. FOX: A FOX-inspired optimization algorithm. *Appl. Intell.* **2023**, *53*, 1030–1050. [\[CrossRef\]](#)
44. Yang, Y.; Newsam, S. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '10)*, San Jose, CA, USA, 2–5 November 2010; ACM Digital Library: New York, NY, USA, 2010; pp. 270–279. [\[CrossRef\]](#)

45. Usman, O. L.; Muniyandi, R. C.; Omar, K.; Mohamad, M.; Owoade, A. A.; Kareem, M. A. HoRNS-CNN Model: An Energy-Efficient Fully Homomorphic Residue Number System Convolutional Neural Network Model for Privacy-Preserving Classification of Dyslexia Neural-Biomarkers. *Brain Inform.* **2025**, *12*, 11. [CrossRef] [PubMed]
46. UC Merced Land Use Dataset. Available online: <https://www.kaggle.com/datasets/abdulhasibuddin/uc-merced-land-use-dataset> (accessed on 22 August 2026).
47. NWPU-RESISC 45 Dataset. Available online: <https://www.kaggle.com/datasets/aqibrehmanpirzada/nwpuresisc45> (accessed on 22 August 2026).
48. Chen, C.; Sun, K.; He, S. An improved image encryption algorithm with finite computing precision. *Signal Process.* **2020**, *168*, 107340. [CrossRef]
49. Hirose, M.; Imaizumi, S.; Kiya, H. Learnable image encryption without key management for privacy-preserving vision transformer. *IEEE Access* **2025**, *13*, 201351–201362. [CrossRef]
50. Fallahy, S.; Rezazadeh, N.; Caputo, F.; Lughmani, W.A.; Luca, A.D. Reliability-Aware Selective Fusion of Visual and Class-Associated Acoustic Data for Concrete-Surface Classification Under Simulated Sensor Degradation. *Buildings* **2026**, *16*, 3518. [CrossRef]
51. Melosik, M.; Marszalek, W. Strengthening Quality of Chaotic Bit Sequences. *Electronics* **2022**, *11*, 272. [CrossRef]
52. Fan, C.; Ding, Q. A Novel Algorithm to Analyze the Dynamics of Digital Chaotic Maps in Finite-Precision Domain. *Chin. Phys. B* **2023**, *32*, 010501. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.