symmetry

Special Issue Reprint

Symmetry and Asymmetry in Information Security and Network Security

Edited by
Bingwen Feng

mdpi.com/journal/symmetry

MDPI

Symmetry and Asymmetry in Information Security and Network Security

Symmetry and Asymmetry in Information Security and Network Security

Guest Editor

Bingwen Feng

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editor
Bingwen Feng
College of Cyber Security
Jinan University
Guangzhou
China

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Symmetry* (ISSN 2073-8994), freely accessible at: https://www.mdpi.com/journal/symmetry/special_issues/E8107R255V.

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-8489-6 (Hbk)
ISBN 978-3-7258-8490-2 (PDF)
https://doi.org/10.3390/books978-3-7258-8490-2

Contents

About the Editor

Bingwen Feng

Bingwen Feng is an Associate Professor at the College of Cyber Security, Jinan University, Guangzhou, China. His research interests lie at the intersection of multimedia security and artificial intelligence security.

Preface

The following Reprint examines symmetric and asymmetric structures in data hiding for network security, covering steganography, watermarking, and covert channels. Its aim is to advance principled design by highlighting the trade-offs between robustness and stealth. It is motivated by the lack of unified treatment across disciplines. This Reprint is intended for researchers, engineers, and analysts. All contributions are peer-reviewed. We thank the authors and reviewers for their valuable effort and contributions.

Bingwen Feng
Guest Editor

symmetry

MDPI

Article

MVCA: Multi-View Cross-Attention Framework for Robust Shilling Attack Detection

Zhengli Zhai *, Cheng Xu *, Yang Li and Shunqi Su

School of Information and Control Engineering, Qingdao University of Technology, Qingdao 266520, China
* Correspondence: zzl@qut.edu.cn (Z.Z.); 202322050865@stu.qut.edu.cn (C.X.)

Abstract

Recommender systems are now integral to many online platforms, including e-commerce, social media, and content streaming services. However, their widespread use also exposes them to significant security threats. One of the most critical is the shilling attack, where fake user profiles are injected to manipulate recommendation results. Such attacks undermine system fairness and erode user trust. Traditional detection methods mostly rely on a single perspective, such as a fake profile, temporal behavior or a graph structure, and they have difficulty dealing with complex and changeable attack strategies. Therefore, we propose a multi-view cross-attention (MVCA) attack detection framework. This system integrates three complementary features: the user–item interaction graph structure, the temporal behavior sequence, and the local scoring mode. We propose a bidirectional cross-attention mechanism to achieve deep information interaction, dynamically mine the potential correlations between different views, solve the collaborative optimization of each module, and improve the accuracy of identifying fake users. Extensive experiments conducted on the MovieLens and Netflix datasets have shown that MVCA generally outperforms several established baseline methods. Its strong performance in handling different types and scales of attacks demonstrates the method's adaptability and robustness for detecting shilling attacks.

Keywords: recommendation systems; shilling attack; fake profile; cross-attention mechanism

1. Introduction

Recommender systems have become an indispensable component of modern online platforms, facilitating personalized content delivery and enhancing user engagement [1]. By analyzing historical interactions and user preferences, these systems effectively alleviate information overload and improve user satisfaction [2]. However, their widespread adoption has also made them a prime target for malicious activities, particularly shilling attacks, wherein attackers inject biased profiles into the system to manipulate recommendation outcomes [3,4]. Such attacks not only degrade the quality of recommendations but also undermine user trust and platform credibility [5].

The core of recommender systems lies in their ability to model user preferences and item characteristics, typically through collaborative filtering, content-based methods, or hybrid approaches [6,7]. While these systems excel at predicting user interests under normal conditions, they are inherently vulnerable to manipulation due to their open nature and reliance on user-contributed data [8]. Attackers exploit this vulnerability by injecting carefully crafted fake profiles, often designed to mimic genuine user behavior, in order to

bias the system toward promoting or demoting specific items. This form of manipulation, known as a shilling attack or profile injection attack, poses a serious challenge to the integrity and reliability of recommender systems [9,10], especially in high-stakes domains such as e-commerce, social media, and content streaming platforms.

The core challenge stems from the dynamic and complementary nature of signals from different behavioral views. For example, a local rating pattern may indicate anomalies in score distribution, and a temporal sequence might reveal suspicious bursts of activity. However, strong evidence of an attack often comes from the correlation between a user's structural position in the interaction graph and their sequential behavior [11]. Shallow fusion strategies fall short of modeling these complex, cross-view dependencies. They treat each view as a separate, static information stream and lack a mechanism to allow features from one view to dynamically guide the selection and weighting of relevant cues in another [12]. This results in a representation that is less discriminative and robust against strategically crafted attacks [13].

To mitigate the threat of shilling attacks, researchers have developed a variety of detection methods [14]. Early approaches primarily relied on statistical analysis and handcrafted behavioral features, such as rating deviation or entropy, to identify anomalies [15]. More recently, graph-based methods have been adopted to capture structural anomalies in user–item interaction networks, while sequence-based models leverage temporal patterns to detect suspicious behavior [16]. Despite these advances, most existing methods are limited by their reliance on a single view of user behavior—whether structural, temporal, or local—which restricts their ability to detect sophisticated and adaptive attacks [17,18]. Moreover, even multi-view approaches often integrate heterogeneous features in a shallow manner, such as simple concatenation or linear fusion, failing to capture the deep, nonlinear interactions between different behavioral perspectives [19].

To address the issue of attack detection in recommendation systems, we propose a framework based on a multi-view cross-attention mechanism, which is used to effectively extract and fuse features from different data sources to identify potential attack behaviors. The core idea of this framework is to enhance the model's ability to recognize attack behaviors by constructing a user–item interaction graph and utilizing multiple feature extraction methods and cross-attention mechanisms. Our main contributions are summarized as follows:

- We propose a multi-view feature fusion method. Unlike traditional methods that rely only on one data view, we systematically integrate three different feature views to comprehensively describe user behavior. By combining user–item interaction diagrams, sequence information, and high-order connectivity, rich features are extracted to enhance the expressive ability of the model.
- We design a dynamic and bidirectional cross-attention mechanism. Traditional multi-feature fusion methods often have insufficient information interaction, while this mechanism allows for in-depth interaction between two information sources, thereby unearthing the intrinsic and complex nonlinear relationships between them.

Subsequently, we conducted extensive comparative experiments on representative object detection methods and evaluated the proposed approach on two benchmark datasets to validate its effectiveness.

2. Related Work

2.1. Shilling Attack Modules

Shilling attacks, often referred to as profile injection attacks, involve the intentional creation of counterfeit user profiles that closely simulate authentic user behavior with the aim of manipulating recommendation algorithms. The primary objective of these

https://doi.org/10.3390/sym18030497

attacks is to either enhance push attacks or diminish nuke attacks on the visibility of specific target items by introducing biased ratings [20,21]. Attackers typically employ a systematic approach in constructing these profiles, which can be categorized into four principal components:

Target Item Set (I_T): This set comprises the specific items that the attacker intends to either promote or demote. In the context of a push attack, the target items are assigned the maximum possible ratings to enhance their visibility and appeal. Conversely, in a nuke attack, these items receive the minimum ratings to diminish their perceived value and popularity.

Selected Item Set (I_S): This category includes items that are strategically chosen to augment the efficacy of the attack. For instance, in a bandwagon attack scenario, the selected items are typically those with the highest popularity, which are then rated favorably. This tactic is employed to bolster the credibility of the fake profile by aligning it with mainstream preferences.

Filler Item Set (I_F): The filler items are randomly selected and rated to simulate the behavior of a legitimate user, thereby enhancing the authenticity of the fake profile. The rating strategy for these items is contingent upon the specific attack methodology. For example, in a random attack, filler items may be assigned arbitrary ratings, whereas in an average attack, the ratings are calibrated to approximate the mean rating of the item, thereby maintaining a semblance of normal user behavior.

Unrated Item Set (I_N): This set consists of items that the attacker deliberately chooses not to rate. The rationale behind this decision is to avoid raising suspicions that could arise from an excessive number of ratings, which might otherwise signal the presence of a fabricated profile. By selectively omitting ratings, the attacker aims to maintain a low profile and reduce the likelihood of detection.

Several shilling attack models have been identified in the literature, each with its own strategy for selecting and rating items:

Random Attack: In this model, the filler items are randomly selected and assigned random ratings. The target items are given the highest or lowest ratings depending on whether the attack is a push or nuke attack.

Average Attack: Here, the filler items are randomly selected but are assigned ratings close to the average rating of the item. This makes the fake profile appear more realistic, as the ratings are consistent with the general user behavior.

Bandwagon Attack: This attack leverages the popularity of certain items. The selected items are the most popular ones, which are given the highest ratings. The filler items are randomly selected and assigned random ratings. This strategy is effective because popular items are more likely to be rated by genuine users, making the fake profile less suspicious.

The division and construction strategies of the different attack models are shown in detail in Table 1.

2.2. Detection of Shilling Attacks in Recommendation Systems

The existing research on shilling attack detection has evolved through several paradigms, which can be broadly categorized into methods based on statistical analysis, graph-based modeling, sequential pattern mining, and more recently, multi-view integration. Each category captures a distinct facet of user behavior yet frequently proves insufficient for delivering comprehensive security protection.

https://doi.org/10.3390/sym18030497

Table 1. Division and construction tactics of attack models. $r_{\max}$ and $r_{\min}$ represent the highest and lowest rating values in the Collaborative Recommender System. μ and μ_i represent the mean values of all items and Item i, respectively. σ and σ_i represent the standard deviation of all items and Item i, respectively. $N(\mu, \sigma)$ represents obtaining a random number that conforms to a normal distribution. $r_i = r_{max}$ for push attacks and $r_i = r_{\min}$ for nuke attacks.

Attack Models	**Division and Construction Tactics**			
	I_S	I_F	I_N	I_t
Random	$\varnothing$	Items are randomly selected from $I - I_t$. For $i \in I_F, r_i \sim N(\mu, \sigma)$	Do not rate items in I_N	For $i \in I_t$, $r_i = r_{\max}$ or $r_{\min}$
Average	$\varnothing$	Items are randomly selected from $I - I_t$. For $i \in I_F, r_i \sim N(\mu_i, \sigma_i)$		
Bandwagon	K most popular items. For $i \in I_s$, $r_i = r_{max}$	Items are randomly selected from $I - I_S - I_t$. For $i \in I_F, r_i \sim N(\mu, \sigma)$		

Statistical and Behavior-Based Methods: Early detection approaches primarily relied on identifying statistical anomalies in user rating patterns. For instance, Mehta et al. proposed PCA-Based Detector [22], which utilizes dimensionality reduction techniques such as principal component analysis, combined with prior knowledge and observed scoring data, to calculate the abnormal probability of users. Similarly, Yang et al. proposed BayesDetector [23] based on the Bayesian inference idea, which identifies abnormal users by fusing prior distributions with observed data. While these methods are interpretable and computationally efficient for simple attack models, they often struggle to capture the complex, nonlinear patterns characteristic of sophisticated attacks. Their performance is highly dependent on hand-crafted features and strong assumptions about the distribution of genuine and attack profiles, which limits their adaptability and generalization to evolving attack strategies.

Graph-Based Methods: With the success of graph neural networks, numerous studies have modeled user–item interactions as bipartite or similarity graphs to capture high-order structural dependencies for attack detection. For example, Zhang et al. proposed GraphRFI [24], which combines GCNs with random forests, achieving both robust recommendation and fraud detection. Simultaneously, Wu et al. proposed USG-SAD [25], which utilizes the comprehensive score correlation and bias to construct a user similarity map, and it then uses a GCN to learn user embeddings for classification. These methods excel at modeling the global structural relationships and connectivity patterns within the system, which are crucial for identifying users who form abnormal clusters or have suspicious link structures. However, a significant limitation is that they often treat the user–item graph as static, overlooking the rich temporal dynamics and evolution inherent in user interaction sequences. This temporal information can be a critical signal for identifying malicious behavior, such as burst-rating activities.

Sequence-Based Methods: To capture the temporal evolution of user behavior, sequence-based models have been introduced, such as DL-DRA [26] proposed by Zhou et al., which employs deep neural networks with dynamic linear dimensionality reduction to learn from rating sequences efficiently. This recurrent neural network and its variants can be used to simulate user interaction sequences, preserving the historical background to detect anomalies that change over time. While these approaches are effective at learning sequential patterns and identifying unusual time intervals or rating sequences, they typically operate in isolation. They fail to leverage the complementary, global struc-

https://doi.org/10.3390/sym18030497

tural information encapsulated in the user–item graph, which can provide crucial context about a user's role in the entire network.

Methods Focusing on Local Patterns: Convolutional Neural Networks (CNNs) have been adapted to analyze local rating patterns or the aggregation of user and item features. For example, Zhou et al. proposed robust recommendation-oriented malicious attack detection [27], which integrates multiple CNN models through bagging and synthesizing judgments from different perspectives through a majority voting mechanism, through which more stable and comprehensive detection results can be obtained. This method treats user and item feature vectors as one-dimensional signals and uses convolutional layers to extract salient, localized features that might be indicative of anomalous rating behavior. However, these CNN-based detectors often focus exclusively on this local view, failing to integrate the broader temporal and structural context of user behavior. This narrow focus limits their holistic understanding and makes them vulnerable to attacks designed to appear normal in local feature windows. "While these works establish CNN's effectiveness for local pattern analysis, they typically employ CNN as the sole feature extractor. In contrast, our framework positions CNN as one of three complementary views, specifically responsible for capturing local rating anomalies, while delegating temporal dynamics to GRU and global structure to GCN. This division of labor is motivated by the heterogeneous nature of shilling attack signatures—no single architectural inductive bias suffices for comprehensive detection."

Multi-View and Hybrid Methods: Recognizing the limitations of single-view models, some recent works have attempted to integrate multiple perspectives, such as CoDetector [28] proposed by Dou et al., which integrates matrix factorization and word embedding techniques to uncover latent features from both user–user and user–item symbiosis matrices. Another method is DGA-MFCA, proposed by Xu et al. [29], which detects attackers within groups by grouping based on user characteristics and using Gaussian-RBF analysis. However, their integration strategies have inherent architectural limitations. CoDetector and DGA-MFCA can be described as multi-stage pipelines rather than deeply integrated multi-view learning systems. They handle different feature spaces separately and combine them at the decision level or through shallow statistical measures, which prevents the models from capturing complex, nonlinear interactions, such as the complex and nonlinear relationship between user structural roles and their temporal behaviors. The ITRN attempts to combine temporal and structural cues, but its fusion mechanism performs the shallow concatenation of the independently learned representations before the final classifier. This static aggregation lacks the ability to dynamically adjust the importance of each view based on the context of other views.

Although these methods have achieved relatively good results, they often rely on a single perspective of user behavior, such as rating patterns, temporal dynamics or graph structures, which limits their ability to detect complex and evolving attack strategies. Moreover, many multi-perspective detection frameworks integrate heterogeneous features in a shallow or static manner and are unable to capture the deep interrelationships between different behavioral perspectives. This gap prompts us to propose MVCA, which aims to systematically and deeply integrate the local, sequential and structural perspectives of user behavior through a dynamic cross-attention mechanism.

The MVCA we propose is designed as an end-to-end multi-perspective collaborative learning framework in its architecture. The core difference lies in the introduction of a bidirectional cross-attention mechanism, which systematically and deeply integrates the local, sequential and structural perspectives of user behavior through the dynamic cross-attention mechanism. Unlike shallow fusion, this mechanism allows for deep and dynamic information interaction between different perspectives. Specifically, features from the con-

https://doi.org/10.3390/sym18030497

tinuous perspective (GRU) can guide the aggregation of the structural perspective (GCN), and vice versa, enabling the model to discover potential correlations that are not easily detectable when individual perspectives are processed alone. This architecture enables MVCA to more effectively handle complex and constantly evolving attack strategies.

3. The Proposed Method

In this section, we present our MVCA framework, whose overall architecture is shown in Figure 1. We design an attack detection framework consisting of several attack feature extraction components. It is composed of an interactive attention mechanism network based on RNNs and GCNs and a feature extraction network based on CNNs, which respectively model the information of different scales of user–item interactions, learn different patterns and rules of user interaction with the project, and comprehensively capture potential attack behavior patterns. Finally, the features of users and items obtained from multiple sources are embedded in multiple views and input into the multi-layer perceptron to obtain the final user classification result. More detailed information will be explained in the following sections.

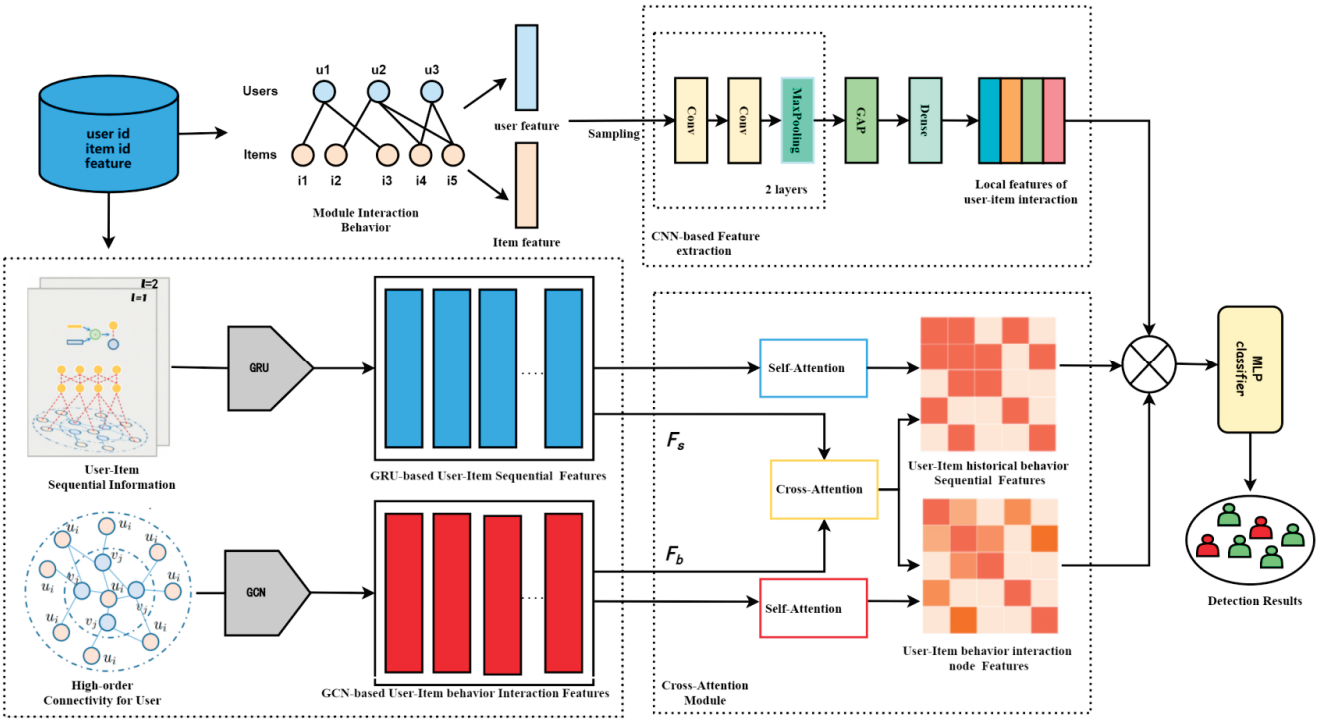


Figure 1. The detection framework of MVCA.

3.1. CNN-Based Feature Extraction

To extract local, representative features from the original user and item data, we employ a CNN as the local feature extraction module. CNNs are chosen for their inherent advantages in capturing local correlations and providing translation invariance. This enables them to identify local abnormal patterns in user–item interactions, such as atypical rating combinations or densely clustered anomalous regions. Secondly, compared to fully connected networks, CNNs myopically reduce model complexity through parameter sharing and local connections, enhancing the generalization ability. This module first concatenates user vectors ($\boldsymbol{u}$) and item vectors ($\boldsymbol{i}$) to obtain the initial input feature matrix ($\boldsymbol{X}$) and

processes them as follows. Firstly, the input features are passed through two convolutional layers to detect local correlations and anomalous patterns. This is formulated as:

$$C^{(l)} = \text{ReLU}(\text{Conv1D}(C^{(l-1)})) \tag{1}$$

where $C^{(0)} = X_i$, and Conv1D denotes a one-dimensional convolution operation.

Subsequently, a max-pooling layer is applied for down-sampling, reducing feature dimensionality while retaining the most salient information:

$$P = \text{MaxPool}(C^{(L_c)}) \tag{2}$$

This is followed by a global average pooling layer to aggregate the features into a compact representation. Finally, a fully connected layer transforms the pooled features into the final local feature representations:

$$F_{uc}, F_{ui} = \text{FC}(\text{GlobalAvgPool}(P)) \tag{3}$$

Here, F_{uc} and F_{ui} denote the extracted local features for the user and item, respectively, which are utilized in the subsequent fusion and classification stages. Its architecture is shown in Figure 2.

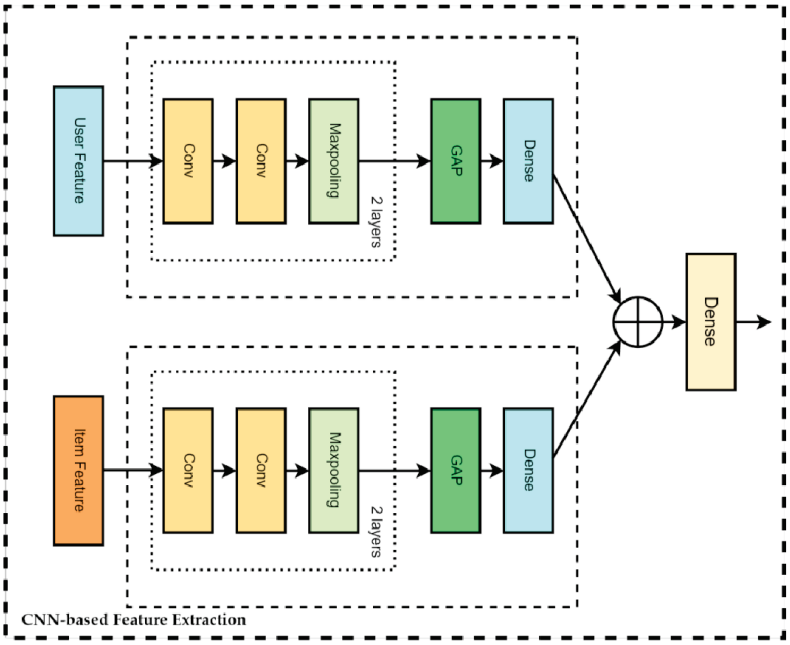


Figure 2. The architecture of the CNN-based feature module.

3.2. Sequential Feature Extraction

To capture the temporal dynamics and sequential dependencies inherent in user–item interactions, we employ a Gated Recurrent Unit (GRU) network to model the historical behavior sequences of users. The GRU network is chosen for its ability to effectively capture long-range dependencies while mitigating the vanishing gradient problem commonly encountered in traditional RNNs. We construct the interaction sequence for each user by sorting their historical ratings in chronological order. Each interaction at time step t is represented as a feature vector (x_t), which is formed by concatenating the user embedding, item embedding, and corresponding rating value:

$$x_t = \text{Concat}(\text{u}_t, i_t, r_t) \tag{4}$$

where u_t and i_t denote the user and item embeddings at time t, and r_t represents the rating value. For users with fewer than T interactions, we apply zero-padding to maintain a fixed sequence length. The GRU processes the input sequence $\boldsymbol{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ through iterative updates of its hidden state. The update mechanism at each time step is represented as:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{5}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{6}$$

$$\widetilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{7}$$

$$h_T = (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h}_t \tag{8}$$

where z_t and r_t represent the update and reset gates, respectively, σ denotes the sigmoid activation function, and $\odot$ indicates element-wise multiplication. The matrices W_*, U_* and bias vectors (b_*) are learnable parameters.

After processing the entire sequence, we extract the final hidden state as the sequential feature representation:

$$F_s = h_T \tag{9}$$

where $F_s \in \mathbb{R}^{d_h}$ encapsulates the temporal behavior patterns of the user and serves as input to the subsequent modules for attack detection. Its architecture is shown in Figure 3.

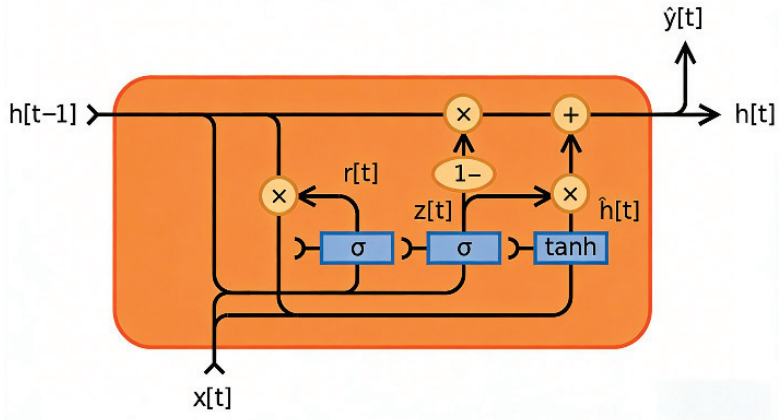


Figure 3. The GRU structure for information preservation.

3.3. GCN-Based Feature Extraction

To capture the similarities among users, the correlations among items, and the complex high-level interaction patterns between users and items, we propose an interactive feature learner for the GCN based on the correlation between users and items. It can comprehensively detect anomalous behavior within complex interaction networks, and the model integrates both localized and global relational patterns. Its architecture is shown in Figure 4 and consists of three main components: the embedding layer provides the initial embedding, the embedding propagation layer simulates the high-order connectivity within the bipartite graph through a stack of enhanced graph convolutional network layers, and the aggregation layer connects the embeddings of different layers to acquire a fine-grained representation of interaction features.

https://doi.org/10.3390/sym18030497

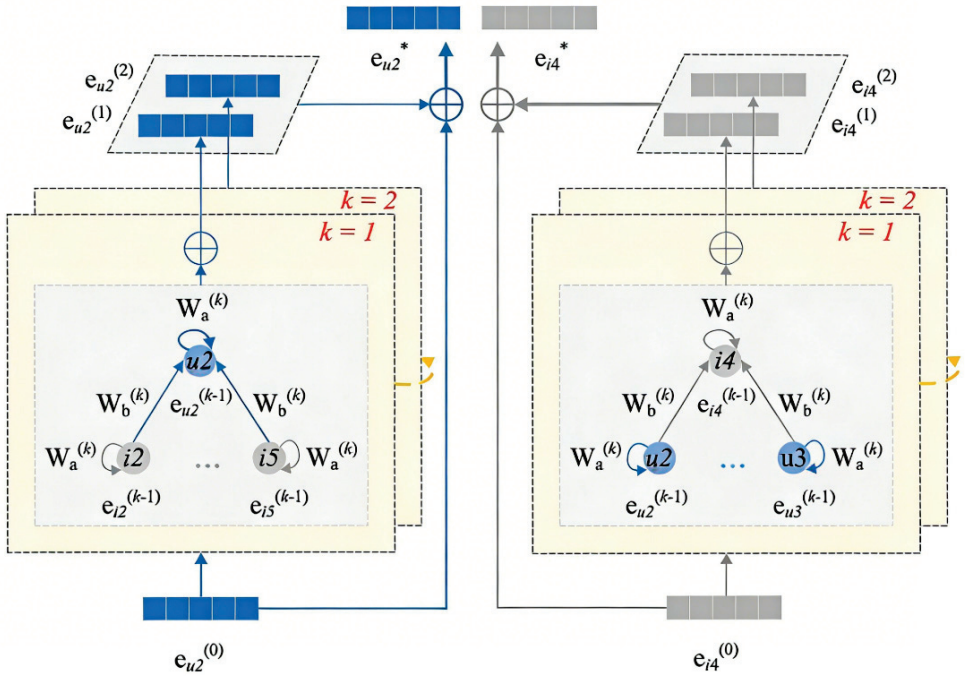


Figure 4. The architectural design of the GCN-based interaction feature extractor.

The GCN feature extractor we proposed is divided into an embedding layer and an embedding propagation layer. In the embedding layer, the initial embedding vector for users and items is generated. The embedding vector $e_u^{(0)} \in \mathbb{R}^d (e_i^{(0)} \in \mathbb{R}^d)$ represents a user (u) and an item (i), and d represents the embedding size. Then, Equation (10) shows the embedding matrix (E), where N and M indicate the total counts of users and items, respectively. This embedding matrix can map each user and item to an initialized embedding representation:

$$E = \left[e_{u1}^{(0)}, \ldots, e_{uN}^{(0)}, e_{i1}^{(0)}, \ldots, e_{iM}^{(0)} \right] \tag{10}$$

Within the embedding propagation framework, multiple enhanced GCN modules are sequentially stacked atop the interaction graph topology to facilitate the integration of high-order collaborative signals, thereby iteratively refining latent representations for both users and items. The GCN operation within this propagation schema comprises two fundamental phases: message generation and neighborhood aggregation. Specifically, by cascading k such propagation layers, each node becomes receptive to information emanating from its k-hop neighborhood, enabling the capture of multi-hop relational patterns. At the k-th order of the GCN, the message $m_{u \leftarrow i}^{(k)}$ passed from node i to node u is defined by Equation (11). Considering the u node's characteristics, we also add a self-connection to the node, and the self-connection message $m_{u \leftarrow u}^{(k)}$ of u is shown in Equation (12):

$$m_{u \leftarrow i}^{(k)} = pui\left(W_a^{(k)} e_i^{(k-1)} + W_b^{(k)} \left(e_i^{(k-1)} \odot e_u^{(k-1)} \right) \right) \tag{11}$$

$$m_{u \leftarrow u}^{(k)} = W_a^{(k)} e_u^{(k-1)} \tag{12}$$

where $W_a^{(k)} \in \mathbb{R}^{d' \times d}, W_b^{(k)} \in \mathbb{R}^{d' \times d}$ represents the learnable transformation parameters employed at the k-th propagation stage for distilling salient features during information diffusion, where d signifies the dimensionality of the latent representations, and d' denotes the transformation size. $e_i^{(k-1)}$, $e_u^{(k-1)}$ are the item and user embedding representations generated from the previous message propagation stages, and they store information about

their ($k-1$)-order neighbors. Our approach extends beyond conventional GCNs by explicitly modeling the direct interaction between the user (u) and item (i), rather than relying solely on neighborhood information aggregation. The element product $\odot$ incorporates the relationship between users and items into the information generation process. *pui* is set to Laplace's parametrization, as shown in Equation (13), where N_u and N_i denote the k-order neighbors of the user (u) and item (i), *respectively*. A normalization term (*pui*) is introduced to the adjacency matrix. This term adaptively modulates the weighting contributions from neighboring nodes, thereby mitigating potential bias caused by the dominance of certain information flows during multi-layer convolutional aggregation:

$$pui = \frac{1}{\sqrt{|N_u||N_i|}} \tag{13}$$

During the message aggregation stage, the representation of node u is refined by incorporating messages propagated from its k-th-order neighbors. The specific function used for this aggregation is provided in Equation (14):

$$e_u^{(k)} = \text{LeakyReLU}\left(m_{u \leftarrow u}^{(k)} + \sum_{i \in \text{N}_u} m_{u \leftarrow i}^{(k)} \right) \tag{14}$$

Motivated by the inherent sparsity of user–item interactions in recommendation systems, we incorporate the LeakyReLU activation function to introduce necessary nonlinearity into the model. To capture higher-order dependencies, we utilize a k-layer GCN architecture. This design propagates and aggregates information across k hops, ultimately yielding the k-th-order representations for both user ($e_u^{(k)}$) and item ($e_i^{(k)}$).

$$e_i^{(k)} = \text{LeakyReLU}\left(m_{i \leftarrow i}^{(k)} + \sum_{u \in \text{N}_i} m_{i \leftarrow u}^{(k)} \right) \tag{15}$$

In the aggregation layers, in contrast to standard GCNs that draw upon representations from only a single layer (typically either early or final), our approach integrates features across all layers to capture information at heterogeneous granularities. From the perspective of system architecture, the lower layer of the model is responsible for encoding the real-time interaction signals of users at the micro-level, while the upper layer is used to extract trends and patterns at the macro-level, including dynamically evolving user preferences and unconventional network structure features. By directly concatenating the vectors output by each layer, the system achieves the effective fusion of information of different granularities, and we obtain user embeddings (e_u^*) and item embeddings (e_i^*) that encode both local and global interaction information, as shown in Equations (16) and (17):

$$e_u^* = e_u^{(0)} \| \cdots \| e_u^{(k)} \tag{16}$$

$$e_i^* = e_i^{(0)} \| \cdots \| e_i^{(k)} \tag{17}$$

Algorithm 1 outlines the implementation of the interactive feature learner based on the GCN.

Then, the user embedding (e_u^*) and the item embedding (e_i^*) are input into the CNN to obtain the user–item behavior interaction feature (F_b).

https://doi.org/10.3390/sym18030497

Algorithm 1. GCN-based interaction feature learner

Input: User-item interaction graph G, user set U, item set I, the embedding propagation depth K, GCN network parameters $\left\{W_a^{(1)}, W_a^{(2)}, \cdots W_a^{(K)}, W_b^{(1)}, W_b^{(2)}, \cdots W_b^{(K)}\right\}$, user initialized embedding $e_u^{(0)}$, item initialized embedding $e_i^{(0)}$

Output: Final embedding of user and item representations e_u^* and e_i^*

$e_u^* = e_u^{(0)}$

$e_i^* = e_i^{(0)}$

for k in range(1:K) **do**

  for $u \in U$ **do**

  $m_{u \leftarrow u}^{(k)} = W_a^{(k)} e_u^{(k-1)}$

    for $i \in N_u$ **do**

      $m_{u \leftarrow i}^{(k)} = pui\left(W_a^{(k)} e_i^{(k-1)} + W_b^{(k)}\left(e_i^{(k-1)} \odot e_u^{(k-1)}\right)\right)$

    end for

    $e_u^{(k)} = \text{LeakyReLU}\left(m_{u \leftarrow u}^{(k)} + \sum_{i \in \mathrm{N}_u} m_{u \leftarrow i}^{(k)}\right)$

    end for

    $e_u^* = e_u^* \left\| e_u^{(k)} \right.$

end for

for k in range(1:K) **do**

  for $i \in I$ do

    $m_{i \leftarrow i}^{(k)} = W_a^{(k)} e_i^{(k-1)}$

    for $i \in N_u$ **do**

      $m_{i \leftarrow u}^{(k)} = pui\left(W_a^{(k)} e_u^{(k-1)} + W_b^{(k)}\left(e_u^{(k-1)} \odot e_i^{(k-1)}\right)\right)$

        end for

          $e_i^{(k)} = \text{LeakyReLU}\left(m_{i \leftarrow i}^{(k)} + \sum_{u \in \mathrm{N}_i} m_{i \leftarrow u}^{(k)}\right)$

      end for

      $e_i^* = e_i^* \left\| e_i^{(k)} \right.$

end for

return e_u^*, e_i^*

3.4. Interactive Cross-Attention Module

To enable the deep and dynamic fusion of these views, we design a bidirectional cross-attention module that facilitates interactive information exchange across feature representations. This module allows the model to adaptively highlight task-relevant cues from each view in relation to the others, thereby generating discriminative representations tailored for attack detection. The entire framework is trained end to end under a unified cross-entropy loss, ensuring that all feature extractors are co-optimized toward the ultimate goal of accurately classifying malicious users. On this basis, the interactive attention mechanism can assign different weights to the features extracted by the GCN and GRU according to different task requirements. Through the interactive attention mechanism, the model can selectively obtain key information from the features extracted by the GCN and GRU, effectively integrating structural information and sequence information. This enables the model to comprehensively utilize these two types of complementary information, thereby gaining a more comprehensive understanding of user behavior and item features.

Firstly, the user–item behavior interaction feature (F_b) extracted by the GCN and the user–item sequence feature (F_s) extracted by the GRU capture the dependency re-

https://doi.org/10.3390/sym18030497

lationships among the elements in the sequence through the self-attention mechanism. $F_s = [f_{s1}, f_{s2}, \ldots, f_{sn}] \in \mathbb{R}^{n \times d_s}$, where n denotes the sequence length and d_s denotes the feature dimension. Similarly, the user–item sequence features $F_b = [f_{b1}, f_{b2}, \ldots, f_{bm}] \in \mathbb{R}^{m \times d_b}$, where m represents the number of nodes related in the graph, and d_b is the feature dimension.

Firstly, self-attention processing is carried out on the two features to enhance their internal representational capabilities.

For the sequence feature (F_s):

$$Q_s = F_s W_q^s,\ K_s = F_s W_k^s,\ V_s = F_s W_v^s \tag{18}$$

$$F_s^{\text{self}} = \text{softmax}\left(\frac{Q_s K_s^T}{\sqrt{d_k}}\right) V_s \tag{19}$$

For the behavioral interaction feature (F_b):

$$Q_b = F_b W_q^b,\ K_b = F_b W_k^b,\ V_b = F_b W_v^b \tag{20}$$

$$F_b^{\text{self}} = \text{softmax}\left(\frac{Q_b K_b^T}{\sqrt{d_k}}\right) V_b \tag{21}$$

where W_q^s, W_k^s, W_v^s, W_q^b, W_k^b and W_v^b denote the learnable parameters that map the original feature matrix to query, key, and value vectors, respectively. d_k is the dimension of K. F_b^{self} and F_s^{self} respectively represent the features of the user–item behavior interaction feature (F_b) extracted by the GCN and user–item sequence feature (F_s) extracted by the GRU after self-attention processing. Then, cross-attention is used to capture the interaction relationship between F_s and F_b, which can be defined as:

$$Q_{sb} = F_s^{\text{self}} W_q^{sb},\ K_{sb} = F_b^{\text{self}} W_k^{sb},\ V_{sb} = F_b^{\text{self}} W_v^{sb} \tag{22}$$

$$F_{sb} = \text{softmax}\left(\frac{Q_{sb} K_{sb}^T}{\sqrt{d_{sb}}}\right) V_{sb} \tag{23}$$

where F_s^{self} denotes the query, and F_b^{self}, as the key and value, generates the Q_{sb} query through the weight matrix $W_q^{sb} \in \mathbb{R}^{d_s \times d_{sb}}$ for F_s^{self}, and generates the key (K_{sb}) and value (V_{sb}) for F_b^{self} through $W_k^{sb} \in \mathbb{R}^{d_b \times d_{sb}}$.

In the same way, the cross-attention features of the item center (F_{bs}) are obtained:

$$Q_{bs} = F_b^{\text{self}} W_q^{bs},\ K_{bs} = F_s^{\text{self}} W_k^{bs},\ V_{bs} = F_s^{\text{self}} W_v^{bs} \tag{24}$$

$$F_{bs} = \text{softmax}\left(\frac{Q_{bs} K_{bs}^T}{\sqrt{d_{bs}}}\right) V_{bs} \tag{25}$$

where d_{sb} denotes the interaction feature dimension, and $W_v^{sb} \in \mathbb{R}^{d_b \times d_{sb}}$ calculates the attention score matrix. Finally, the cross-attention feature centered on the user (F_{sb}) is obtained.

In this way, two cross-attention outputs are ultimately obtained, which are represented as feature F_{sb}, centered on the sequence perspective and integrating structural information, and feature F_{bs}, centered on the structural perspective and integrating sequence information. These two features are used as the input for the multi-view fusion for the final attack detection classification.

https://doi.org/10.3390/sym18030497

3.5. Feature Fusion for Attack Detection

After obtaining features from various sources, the local features of the CNN and the associated features of the interactive attention are concatenated by dimensions to form a multi-source feature joint matrix, which can be defined as:

$$F_{\text{fusion}} = F_{uc} \oplus F_{ui} \oplus F_{sb} \oplus F_{bs} \tag{26}$$

where $\oplus$ denotes the concatenation of the features; F_{ui} and F_{uc} denote the local features of the user and item processed by the CNN. F_{sb} and F_{bs} respectively denote the user–item historical sequence features and high-order behavior node features processed by the self-attention mechanism.

To further enhance the representation ability and reduce feature redundancy, we adopt a linear projection layer to map the fused features into a shared latent space, and we then input it into the multi-layer perceptron to predict the attack classification, and the loss function adopts cross-entropy loss, which can be defined as:

$$y = \text{softmax}(W_o \cdot F_{\text{fusion}} + b_o) \tag{27}$$

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{28}$$

where W_o and b_o denote the learnable parameters used to output a probability distribution of behavior for each user. $y_i = 1$ if the user (u) is a genuine user, and $y_i = 0$ otherwise, and N denotes the number of training users.

3.6. Complexity Analysis

To comprehensively evaluate the computational efficiency of the MVCA framework, this section analyzes the complexity and time consumption of the model algorithm. The main parameters are defined as follows: the total number of users (U), the total number of items (I), the number of interaction edges between users and items (E), the embedding dimension (d), the number of GCN propagation layers (K), the average length (L) of the user's historical behavior sequence, and the number of heads in the multi-head cross-attention (H). The complexity analysis of each module is as follows:

The local feature extraction module performs one-dimensional convolution on the embeddings of users and items. Let the size of the convolution kernel be k and the number of filters be F. Then, the complexity of processing all users and items is approximately $O((U + I)\cdot k\cdot F\cdot d)$.

The sequence feature extraction module uses the GRU to process each user's sequence with a length of L. The behavioral sequence of L is modeled. The complexity of the GRU is $O(L\cdot d^2)$, and the total complexity of all users is $O(U\cdot L\cdot d^2)$.

The structure feature extraction module performs message passing and aggregation on the user–item bipartite graph through K-layer graph convolution. Each layer needs to traverse all interaction edges for neighbor aggregation, with a complexity of $O(E\cdot d)$; and it also performs linear transformation, with a complexity of $O((U + I)\cdot d^2)$. Therefore, the total complexity of the GCN module is $O(K\cdot E\cdot d + K\cdot(U + I)\cdot d^2)$.

The cross-attention module conducts bidirectional interaction on the features extracted by the GCN and GRU. Firstly, the sequence features (with dimension L $\times$ d) and the graph node features ($|\mathcal{N}| \times$ d, where $|\mathcal{N}|$ is the average number of nodes related to the user, approximately E/U) are subjected to self-attention calculations, with complexities of $O\left(L^2\cdot d\right)$ and $O((E/U)2\cdot d)$, respectively. Then, mutual attention calculation is carried out, with a complexity of $O(L\cdot(E/U)\cdot d)$. After considering the multi-head mechanism, the complexity needs

https://doi.org/10.3390/sym18030497

to be multiplied by the number of heads (H), where H is a constant. Therefore, the total complexity of the cross-attention module can be expressed as $O(H \cdot (L^2 + (E/U)^2 + L \cdot (E/U)) \cdot d)$.

Therefore, the overall complexity of the proposed approach is $O(K \cdot E \cdot d + K \cdot (U + I) \cdot d^2 + U \cdot L \cdot d^2 + (L^2 + ((E/U)^{2)} + L \cdot E/U) \cdot d)$. In the actual recommendation system scenario, the number of interaction edges (E) is usually much larger than the number of users and items ($E \gg U + I$), the propagation layer number (K), sequence length (L), and embedding dimension (d) are all constants, and the data scale has a quasi-linear relationship. This indicates that MVCA has good scalability and can meet the detection requirements of large-scale recommendation systems. Theoretical analysis results are consistent with the trend of the subsequent measurement of running time, that is, the running time is roughly linearly related to the data volume. This verifies the feasibility of the algorithm in practical deployment.

4. Experiment Results and Analysis

4.1. Experimental Datasets

In our experiment, we utilized two commonly used public datasets to verify our experimental results: MovieLens 10M [30] and Netflix [31], which contain real information configuration files. Each piece of data represents the attribute information corresponding to a click behavior. The original dataset includes a user id, an item id, user features, item features, a timestamp, and the tag corresponding to each piece of data. Both of these datasets used in the attack archive were generated by the attack model. The details of each dataset are shown as follows:

MovieLens 10M: This dataset contains 1120 users, 10,657 movies and 199,040 ratings. Each user has at least 20 ratings. The rating values are between 1 and 5. The MovieLens dataset does not include any attacker profiles. For the purposes of this research, without loss of generality, fake profiles generated by attack models were injected into the dataset.

The Netflix dataset is a competition dataset, consisting of 17,770 records (each record is a quadruple of the user ID, item ID, rating and timestamp) and 480,189 users. All rating values are integer data between 1 and 5 and were collected between October 1998 and December 2005. We randomly selected 215,884 ratings from 2000 users for 4000 movies as the experimental dataset. For the purpose of this study, the data generated by the forged personal data attack model was injected into the dataset without losing generality or labels corresponding to each piece of data.

In the attack detection phase, we split the original training set into a new training set and a validation set with a ratio of 8:2. The dataset provides a test set. We processed the datasets, adjusting the attack size and fill scale to the size we needed, and generated real datasets with different attack scales and fill scales. These profiles varied in attack sizes and filler sizes. To ensure the stability and reliability of the results and to reduce the influence of random factors, in addition, we conducted random sampling (without replacement sampling) on the MovieLens 10M dataset, and we performed this operation for each set. All reported experimental data are the average of five independent runs, and the standard deviations of the key metrics are also indicated to show the stability of the results.

4.2. Evaluation Metrics

We adopt the precision, recall, and F1-measure metrics to assess the performance of USG-SAD, defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{29}$$

$$Recall = \frac{TP}{TP + FN} \tag{30}$$

https://doi.org/10.3390/sym18030497

$$F1\text{-}measure = 2 \cdot \frac{precision \cdot recall}{precison + recall} \tag{31}$$

where TP denotes the number of attack users correctly identified, FN represents the number of attack users incorrectly classified as genuine users, and FP indicates the number of genuine users mistakenly classified as attack users.

4.3. Experiment Results and Analysis

To verify the validity of our method, we used the following works as comparative methods in our experiments:

(1) CoDetector [28]: The fundamental concept of this model is to integrate matrix decomposition and word embedding techniques in order to uncover the latent features of users through both the user–user symbiosis matrix and the user–item symbiosis matrix. This approach facilitates the detection of attack behaviors within recommender systems.
(2) CNN-LSTM [32]: This model employs a hybrid CNN-LSTM deep learning model. It automatically extracts deep features from user ratings through CNNs and combines LSTM to learn sequence dependencies in order to accurately detect shilling attacks in recommendation systems.
(3) DGA-MFCA [29]: DGA-MFCA is a shilling attack detection approach. Users are first grouped based on their characteristics. Then, attackers are detected within these groups using Gaussian-RBF analysis of behavior patterns.
(4) USG-SAD [25]: A supervised learning method is proposed that constructs a user relational graph by computing user similarity through an integrated measure of rating-behavior correlation and deviation. User embeddings are subsequently generated using the Node2Vec algorithm. For attack detection, the model leverages a graph convolutional network (GCN) designed to operate on the user similarity graph, where it learns to assign importance weights to users for classification
(5) GraphRFI [23]: This model combines the GCN and NRF to realize recommendation and fraud detection in the recommendation system through graph structure modeling and random forest classification.
(6) ITRN [32]: This method divides the item scoring time series through key points, uses second-order difference to construct a cube for anomaly interval detection, and builds a bipartite graph of suspicious users–items. Combined with LightGCN to learn high-order neighbor features, it finally determines whether the user is an attacker through the linear layer and the Sigmoid function.

Figure 5 shows the comparison of the detection performances for CoDetector, CNN-LSTM, DGA-MFCA, USG-SAD, GraphRFI and the ITRN of the six baseline methods on the Netflix datasets.

As shown in Figure 5, the detection performance of MVCA exhibits a stronger performance in detecting attacking users in recommendation systems, which demonstrates the progressiveness and effectiveness of the multi-view cross-attention framework in detecting shilling attacks. The MVCA framework integrates three complementary views: the CNN local scoring model, the RNN temporal behavior sequence, and the GCN high-order graph structure. It designs a bidirectional cross-attention mechanism to achieve deep information interaction and dynamic weight distribution among views, forming goal-oriented collaborative optimization in end-to-end training, thereby comprehensively depicting the multi-dimensional characteristics of attack behaviors. In contrast, CoDetector and DGA-MFCA, which rely on matrix factorization or statistical features, lack the ability to model temporal dynamics and structural correlations. Although CNN-LSTM can extract deep features, it ignores the high-order collaborative relationships among users. Pure graph

https://doi.org/10.3390/sym18030497

methods such as USG-SAD and GraphRFI fail to utilize timing bursts and local anomaly signals; ITRNs only adopt shallow splicing and fusion and are unable to capture nonlinear dependencies across views. The limitations of this single perspective or static fusion make it difficult for baseline methods to simultaneously identify the local anomalies, temporal aggregation, and structural concealment of attacks. However, MVCA adaptively correlates the key clues of different views through the attention mechanism, significantly enhancing the discriminative and robust nature of detection, and demonstrating better generalization in complex attack scenarios.

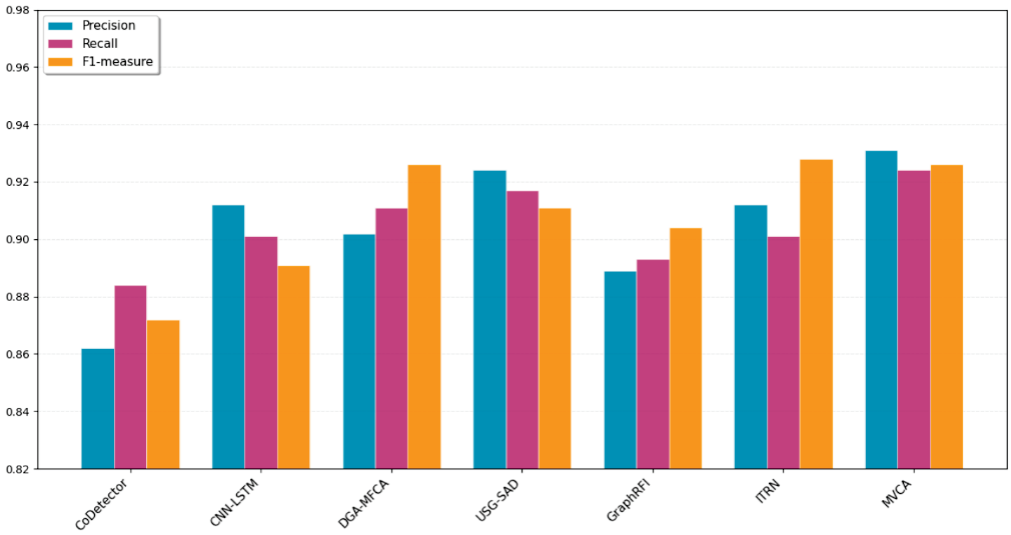


Figure 5. Comparison of detection performances for different methods on Netflix dataset.

Given that the objective of a recommendation attack is to enhance or diminish the rating of a target item and increase or decrease the number of times the target item is recommended, we used the MovieLens 10M dataset in our experiments and adopted the three commonly used recommendation attack models, namely, the random attack, the average attack, and the popularity attack, as the method of generating the attack data because these attack models need rare knowledge of the recommendation systems. We injected these attack data into the system to simulate the shilling attack process, and since newly attacked users often lack trust relationship data, we randomly selected trustees and chose the average number of trustees in the original system as the number of trustees of the attacker. In contrast to the normal shilling attack data, we needed to generate the attacker's trust relationship. As with the model for determining the size of the attacker's profile, we used the average number of users on the system's user trust list as the size of the attacker's trust list. Therefore, to simulate real users and produce better attack results, we obtained the attacker's trustees from users with high trust values.

To comprehensively evaluate the performances and impacts of the attack methods under different intensities, we set the fill sizes of the three shilling attack methods simulated in the experiment as 3%, 5%, 7%, 10% and 15%, and the fill sizes corresponded to the recommended attack sizes of 1%, 5%, 7%, 10% and 10%, respectively. The results of the experiment are shown in Tables 2–4.

https://doi.org/10.3390/sym18030497

Table 2. The results of the MVCA under average attack with different attack sizes and filler sizes.

PFillerSize \ PAttackSize		3%	5%	7%	10%	15%
1%	Precision	0.7806 ± 0.0212	0.8051 ± 0.0237	0.8449 ± 0.0162	0.8751 ± 0.0085	0.9343 ± 0.0042
	Recall	0.8207 ± 0.0194	0.8430 ± 0.0164	0.8652 ± 0.0124	0.9051 ± 0.0073	0.9252 ± 0.0053
	F1-score	0.8382 ± 0.0221	0.7841 ± 0.0129	0.8741 ± 0.0133	0.9246 ± 0.0075	0.9148 ± 0.0081
5%	Precision	0.8112 ± 0.0152	0.8662 ± 0.0095	0.8817 ± 0.0094	0.9277 ± 0.0052	0.9707 ± 0.0043
	Recall	0.8013 ± 0.0183	0.8768 ± 0.0082	0.9121 ± 0.0087	0.9182 ± 0.0077	0.9816 ± 0.0034
	F1-score	0.8190 ± 0.0142	0.8778 ± 0.0063	0.9373 ± 0.0064	0.9403 ± 0.0081	0.9647 ± 0.0043
7%	Precision	0.8456 ± 0.0872	0.8631 ± 0.0085	0.8720 ± 0.0042	0.9316 ± 0.0054	0.9649 ± 0.0024
	Recall	0.8157 ± 0.0103	0.8939 ± 0.0072	0.9023 ± 0.0086	0.9417 ± 0.0032	0.9749 ± 0.0048
	F1-score	0.8256 ± 0.0076	0.8906 ± 0.0058	0.8509 ± 0.0052	0.9213 ± 0.0041	0.9647 ± 0.0039
10%	Precision	0.8328 ± 0.0033	0.8566 ± 0.0042	0.9052 ± 0.0026	0.9410 ± 0.0062	0.9776 ± 0.0025
	Recall	0.8231 ± 0.0027	0.8772 ± 0.0035	0.9253 ± 0.0028	0.9613 ± 0.0074	0.9877 ± 0.0031
	F1-score	0.8720 ± 0.0041	0.8350 ± 0.0038	0.9392 ± 0.0031	0.9496 ± 0.0030	0.9772 ± 0.0015

Table 3. The results of the MVCA under random attack with different attack sizes and filler sizes.

PFillerSize \ PAttackSize		3%	5%	7%	10%	15%
1%	Precision	0.7742 ± 0.0223	0.7982 ± 0.0194	0.8424 ± 0.0125	0.9372 ± 0.0077	0.9542 ± 0.0032
	Recall	0.7903 ± 0.0205	0.8132 ± 0.0204	0.8631 ± 0.0096	0.9451 ± 0.0062	0.9553 ± 0.0047
	F1-score	0.7967 ± 0.0274	0.7942 ± 0.0155	0.8422 ± 0.0168	0.9346 ± 0.0058	0.9448 ± 0.0074
5%	Precision	0.8122 ± 0.0116	0.8334 ± 0.0087	0.8962 ± 0.0092	0.9582 ± 0.0052	0.9701 ± 0.0051
	Recall	0.7922 ± 0.0134	0.8432 ± 0.0072	0.9021 ± 0.0073	0.9701 ± 0.0057	0.9613 ± 0.0042
	F1-score	0.7809 ± 0.0098	0.8613 ± 0.0063	0.9073 ± 0.0042	0.9649 ± 0.0043	0.9627 ± 0.0026
7%	Precision	0.8073 ± 0.0078	0.8342 ± 0.0076	0.8891 ± 0.0043	0.9245 ± 0.0025	0.9515 ± 0.0032
	Recall	0.8057 ± 0.0087	0.8239 ± 0.0054	0.8934 ± 0.0058	0.9117 ± 0.0056	0.9448 ± 0.0056
	F1-score	0.7956 ± 0.0065	0.8406 ± 0.0049	0.8902 ± 0.0061	0.9213 ± 0.0024	0.9435 ± 0.0033
10%	Precision	0.8461 ± 0.0032	0.9042 ± 0.0054	0.9212 ± 0.0028	0.9432 ± 0.0045	0.9712 ± 0.0018
	Recall	0.8531 ± 0.0062	0.9172 ± 0.0042	0.9123 ± 0.0035	0.9321 ± 0.0031	0.9607 ± 0.0034
	F1-score	0.8720 ± 0.0048	0.9030 ± 0.0023	0.9226 ± 0.0029	0.9366 ± 0.0028	0.9783 ± 0.0022

Table 4. The results of the MVCA under bandwagon attack with different attack sizes and filler sizes.

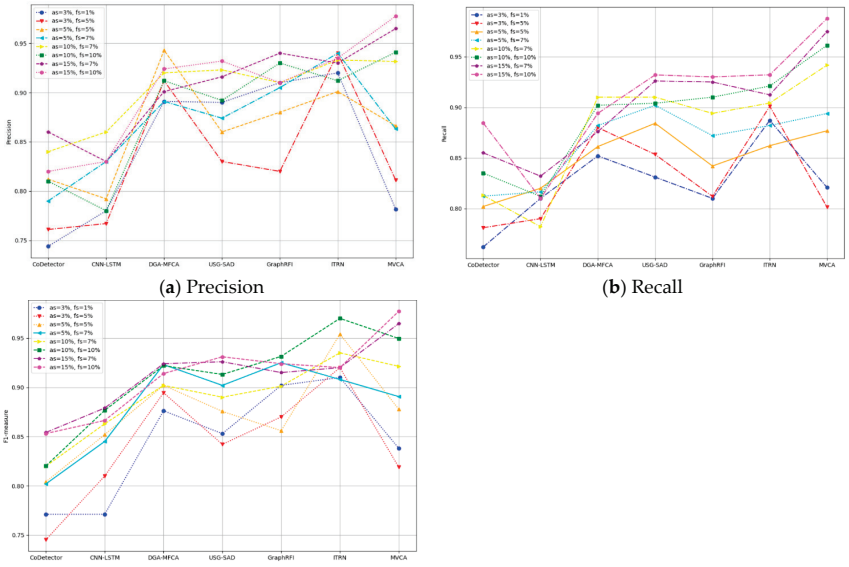
PFillerSize \ PAttackSize		3%	5%	7%	10%	15%
1%	Precision	0.8270 ± 0.0174	0.8721 ± 0.0142	0.9142 ± 0.0080	0.9423 ± 0.0056	0.9751 ± 0.0044
	Recall	0.8312 ± 0.0213	0.8804 ± 0.0165	0.9104 ± 0.0098	0.9364 ± 0.0041	0.9633 ± 0.0075
	F1-score	0.8256 ± 0.0142	0.8672 ± 0.0107	0.9230 ± 0.0072	0.9421 ± 0.0049	0.9721 ± 0.0053
5%	Precision	0.8421 ±0.0108	0.8843 ± 0.0084	0.9012 ± 0.0069	0.9477 ± 0.0054	0.9761 ± 0.0042
	Recall	0.8322 ± 0.0086	0.8921 ± 0.0063	0.9033 ± 0.0082	0.9508 ± 0.0058	0.9724 ± 0.0065
	F1-score	0.8107 ± 0.0112	0.8732 ± 0.0054	0.8938 ± 0.0064	0.9402 ± 0.0071	0.9807 ± 0.0042
7%	Precision	0.8251 ± 0.0058	0.8723 ± 0.0048	0.9212 ± 0.0043	0.9326 ± 0.0031	0.9584 ± 0.0039
	Recall	0.7923 ± 0.0061	0.8932 ± 0.0061	0.9137 ± 0.0050	0.9287 ± 0.0024	0.9532 ± 0.0053
	F1-score	0.7944 ± 0.0046	0.8955 ± 0.0052	0.9205 ± 0.0033	0.9343 ± 0.0036	0.9621 ± 0.0045
10%	Precision	0.8451 ± 0.0041	0.8963 ± 0.0041	0.9211 ± 0.0062	0.9421 ± 0.0037	0.9801 ± 0.0042
	Recall	0.8424 ± 0.0025	0.8762 ± 0.0038	0.9301 ± 0.0043	0.9361 ± 0.0051	0.9789 ± 0.0029
	F1-score	0.8122 ± 0.0039	0.8745 ± 0.0047	0.9250 ± 0.0055	0.9452 ± 0.0028	0.9773 ± 0.0038

As can be seen from Tables 2–4, MVCA maintained a stable performance at different attack sizes and filler sizes under average attack and random attack. The model demonstrated a strong performance under average attack, random attack and popularity attack. Moreover, with the increase in the attack size and fill size, the accuracy, recall rate and F1

https://doi.org/10.3390/sym18030497

score generally showed an upward trend. For instance, when the attack size was 15% and the fill size was 10%, the F1 scores of all attack types exceeded 0.94, reaching as high as 0.9783 under popular attacks. This indicates that when the scale of the attack expands, the model detection effect is better. Because a larger scale of the attack makes the attack behavior more obvious, it is easier for the multi-view feature extraction and bidirectional cross-attention mechanism of the model to dynamically fuse the information of different views, thereby capturing the attack features more comprehensively. Furthermore, the model remains robust against various attack types, verifying its ability to adapt to complex attack strategies by deeply integrating structure, timing, and local views.

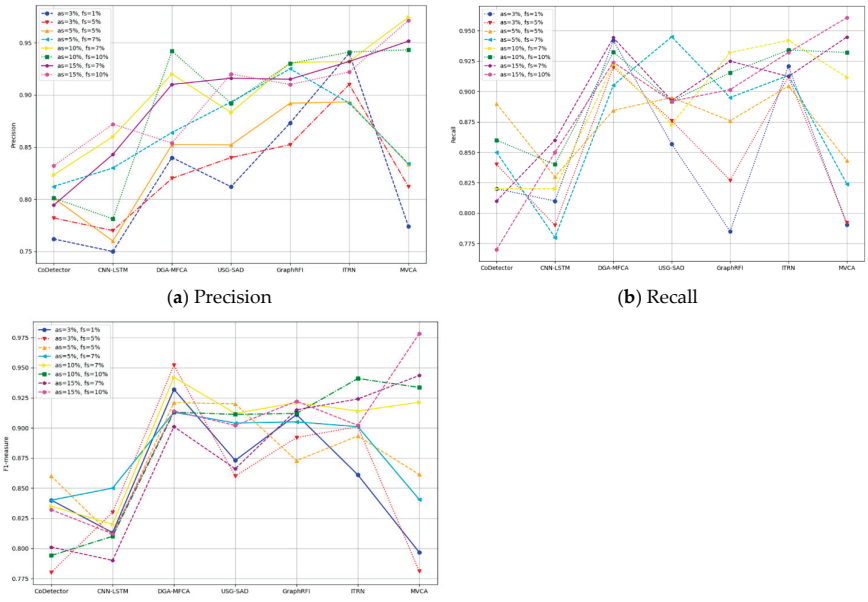
To further prove the robustness of this method for the detection of MVCA, we conducted a thorough comparison of the performances of six baseline methods in different scenarios using the MovieLens dataset. The experimental results are shown in Figures 6–8.



(**a**) Precision (**b**) Recall

(**c**) F1-measure

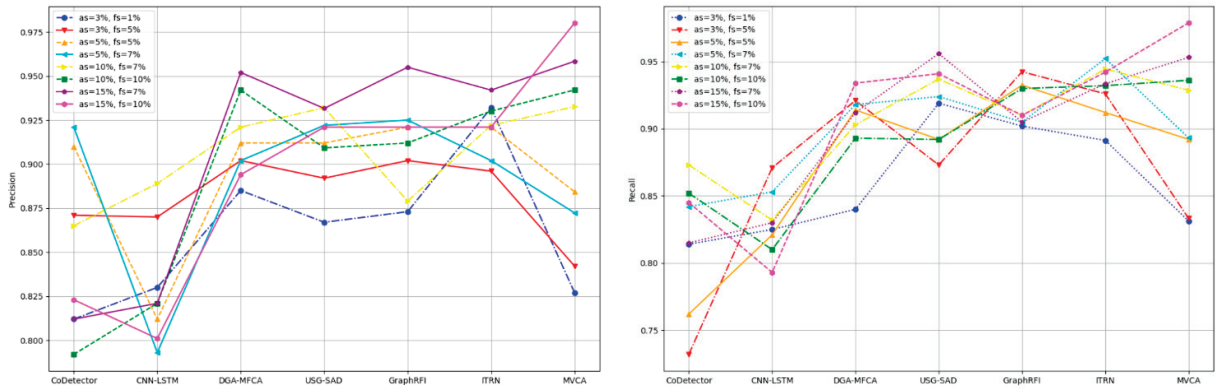
Figure 6. Five methods for detecting random attacks on MovieLens dataset.



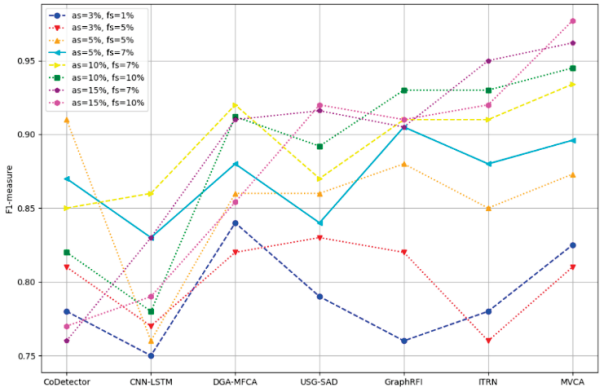
(**a**) Precision (**b**) Recall

(**c**) F1-measure

Figure 7. Five methods for detecting bandwagon attacks on MovieLens dataset.



(**a**) Precision (**b**) Recall



(**c**) F1-measure

Figure 8. Five methods for detecting average attacks on MovieLens dataset.

As shown in Figure 6, MVCA achieves a superior performance in detecting random attacks. While a baseline like CNN-LSTM can extract deep features and learn sequential dependencies, it fails to model the high-order relationships within the user–item interaction graph. Consequently, it cannot capture the collaborative patterns formed by attackers. USG-SAD and GraphRFI, which are pure graph methods, can identify structural anomalies, but they ignore the sudden pattern of ratings in the time dimension and have difficulty detecting distributed time attacks. The success of MVCA lies in the following: When random attackers randomly select to fill in items and assign random ratings, the CNN module detects the abnormality of the local rating distribution, the GRU module captures the sudden rating behavior in the time series, and the GCN module discovers the implicit connections between attackers through high-order neighbor aggregation. The cross-attention mechanism dynamically fuses these three complementary signals to form a more robust discriminative basis than single-view methods. Failure cases mainly occur at the extremely small attack scale (1%) and a low filling rate (3%). At this time, the attack signal is too weak, and it is difficult for the feature extraction of the three views to obtain sufficient discriminative information. This is similar to methods like CoDetector and DGA-MFCA based on statistical features, which face the challenge of signal-to-noise ratios under sparse attacks. The limitations of MVCA in this situation are particularly obvious: when the number of attackers is very small and the rating behavior is highly dispersed, the cross-attention mechanism lacks sufficient "evidence" to establish effective associations between views, the attention weights tend to be uniformly distributed, and the model degenerates into simple feature concatenation, resulting in a significant decline in discriminative ability.

https://doi.org/10.3390/sym18030497

According to Figure 7, in the detection of bandwagon attacks, MVCA demonstrates highest robustness, maintaining an F1-score above 0.92 across all attack scales, significantly outperforming methods such as the ITRN and CNN-LSTM. Although the ITRN constructs abnormal intervals through the use of second-order differences to build a time cube, its shallow concatenation fusion strategy fails to establish a nonlinear correlation between temporal features and graph structure features, resulting in a large number of false negatives when attackers imitate normal users to give high ratings to popular projects. CNN-LSTM, in contrast, lacks a global structural perspective and thus has difficulty identifying the group behavior patterns where attackers deliberately establish connections with popular projects to enhance credibility. The key to the success of MVCA lies in the following: in the GCN view, the popularity attackers form a clear structural aggregation; in the GRU view, they show temporal synchrony; in the CNN view, they present specific local rating patterns. The cross-attention mechanism effectively correlates these cross-view cues through bidirectional interaction; for example, when the GCN detects a user group that is closely connected in the graph structure, the attention mechanism will enhance the temporal anomaly weight of this group in the GRU view. This dynamic collaboration is something that simple feature concatenation methods (such as DGA-MFCA) cannot achieve. However, MVCA has specific limitations in such attacks: when attackers adopt the "slow popularity attack" strategy, that is, deliberately extending the attack time window, dispersing the rating moments to avoid the temporal detection of GRU, and controlling the number of connections with popular projects to reduce the structural significance in the GCN, it is difficult for the cross-attention mechanism to establish strong correlations between views, and the model may mistakenly identify the attackers as real user groups with normal preferences for popular projects. Moreover, if the system itself has a large number of real enthusiasts of popular projects, their structural aggregation and temporal active characteristics are highly similar to those of attackers, and the MVCA faces a higher risk of false positives.

According to Figure 8, in the average attack detection scenario, MVCA faces the greatest challenge, but it still outperforms all baseline methods overall. The underlying reason for this challenge lies in the "perfect camouflage" characteristic of the average attack: attackers set the filled item ratings close to the item average, making detection methods based on statistical anomalies (such as matrix decomposition in CoDetector and Bayesian inference in BayesDetector) almost ineffective; at the same time, the decentralized time strategy weakens the detection capabilities of pure sequence models (such as CNN-LSTM); and imitating the rating patterns of normal users also makes CNN methods relying solely on local features difficult to distinguish. In this scenario, the success of MVCA over other methods mainly relies on the high-order structural information of the GCN view: when the attack scale reaches more than 5%, the attacker forms a synergy on the target item to improve the effect, thereby generating detectable community structures in the user–item interaction graph. At this time, the cross-attention mechanism can automatically reduce its reliance on the CNN and GRU views and enhance the weight of the GCN view, achieving adaptive feature selection. In contrast, USG-SAD, although using a GCN, lacks deep interaction with sequence features and cannot utilize time information to assist in verifying structural anomalies; GraphRFI combines the GCN with random forests, but its static cascading design limits the information flow between views. The failure cases of MVCA are concentrated in small-scale and low-fill-rate combinations because the rating behavior of the average attacker is highly similar to that of real users, and when the number of attackers is small, it is difficult to form significant clustering in the graph structure, resulting in the simultaneous limitation of the discriminative ability of the three views. Here, the fundamental limitation of MVCA is clearly exposed:

https://doi.org/10.3390/sym18030497

When the attacker carefully designs to simultaneously evade the detection of all three views—that is, normal local rating distribution, dispersed time behavior, and sparse graph structure connection—the cross-attention mechanism falls into the predicament because no single view can provide reliable query clues to guide the feature selection of other views. In this case, even by increasing the number of attention heads or adjusting the network depth, it is difficult to break through the discriminative limit at the information theory level.

Overall, the advantage of MVCA lies in multi-view fusion. However, when a certain type of attack causes a sharp decline in the discriminative power of one or two views, the model performance will be compared. Although the cross-attention mechanism attempts to dynamically adjust the weights, the absence or weakening of the core information source inevitably leads to the model's overall sensitivity to this type of attack being lower than that of other types, thereby causing greater performance fluctuations. For instance, in terms of average attacks, attackers greatly downplay the anomalies in local rating patterns and temporal behaviors by imitating the average ratings of normal users. The model mainly relies on GCN views to discover the "graph structure evidence" of collaborative attacks. Only when the scale of the attack reaches a certain level and these attackers form a sufficiently tight and identifiable cluster in the graph structure can the GCN module provide decisive features and the model performance experience a leap. Before this, the performance improvement is relatively gentle.

Moreover, in this study, we utilized t-SNE, a powerful technique proposed by Hinton [33], to visualize high-dimensional implicit features and provide an intuitive understanding of the distribution of original user features. Figure 8 presents a 3D t-SNE visualization that effectively demonstrates the separation between genuine users and malicious attackers within our MVCA framework. The visualization employs a dual-marker scheme, with the blue spheres representing authentic users and the red tetrahedrons denoting attackers.

Notably, while attackers are distributed among legitimate users, they form distinct clusters with characteristic dispersion patterns, reflecting variations in their attack methodologies. These distinct clusters and dispersion patterns are crucial evidence of the effectiveness of our MVCA framework in identifying different types of attackers. The framework can capture the subtle differences in the high-dimensional feature space, which are not easily discernible through conventional methods.

The substantial overlap observed between certain attacker subgroups and genuine user populations suggests sophisticated behavioral emulation strategies employed by attackers. This overlap highlights cases where attackers successfully mimic normal user behavior patterns, thereby evading conventional detection mechanisms. However, despite this overlap, our MVCA framework is still able to identify these attackers by leveraging the unique dispersion patterns and subtle differences in their feature distributions. This demonstrates the robustness and effectiveness of our method in detecting malicious activities even in the presence of sophisticated attacks.

In summary, Figure 9 provides a clear and intuitive visualization of the separation between genuine users and malicious attackers, showcasing the effectiveness of our MVCA framework in identifying and distinguishing different types of attackers based on their high-dimensional implicit features.

https://doi.org/10.3390/sym18030497

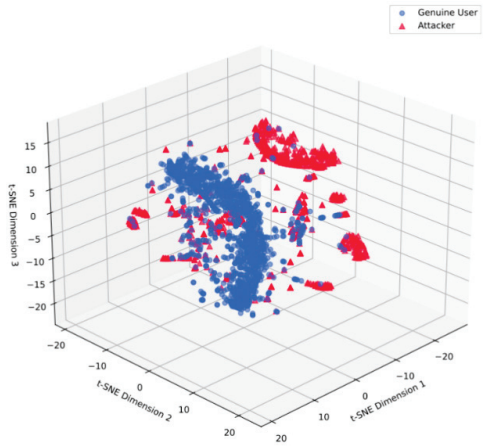


Figure 9. 3D visualization of MVCA user behavior.

4.4. Optimization of Model Hyperparameters and Setting of Key Parameters

To achieve the optimal performance of MVCA, cross-validation experiments were first conducted, and some of them were optimized. Critical parameters exist in a large number of hyperparameters. After all the hyperparameters were determined, we trained the model using a combination and evaluated it on two benchmark test sets. The parameter settings of the proposed method used in the following experiments were as shown in Table 5.

Table 5. Hyperparameters used in this study.

Hyperparameter	Setting
Optimizer	RMSProp
Learning rate	0.0002
Dropout	0.5
Epoch	500
Batch size	1024
Reg.lambda -u and -i	0.8/0.4
Gamma	1
Kernel size	5
Kernel number	64
Dimension of hidden layer in GRU	128
Length of behavioral sequence	50
Number of propagation layers (k)	3
Cross-attention dimension	128
Number of attention heads	[4, 8]

When attackers carry out co-access injection attacks, they frequently click to pop up the target items, thereby increasing the exposure of the target item. Fake users derived from the same group often launch attacks on the same target items, thereby inducing the abnormality of connectivity in the user–project interaction graph. To obtain the connected features in such interaction graphs, we achieve modeling by superimposing the embedding propagation layer of multi-layer graph convolution. Among them, the influence of the number of propagation layers (k) in the GCN is the key hyperparameter of the model. To explore the influence of this parameter on the experimental results, in this study, under the condition of fixing the other parameters, the embedding propagation depth (k) was set to 1, 2, 3, and 4 to conduct parameter optimization experiments. Figure 10 shows the influence of the number of propagation layers (k) on the classification performance. Experiments show that increasing the embedding propagation substantially enhances the model's efficacy. When $k = 3$, the overall performance of all indicators is the best, indicating

https://doi.org/10.3390/sym18030497

that the third-order embedding propagation can effectively reflect the interaction behavior information between users and items. However, compared with $k = 4$, some evaluation indicators of $k = 3$ decreased to a certain extent. The reason is that the overly deep network structure may introduce noise, thereby leading to overfitting and reduced model effect. In general, a three-layer embedded propagation structure is sufficient to effectively obtain interaction information and identify abnormal behavioral patterns.

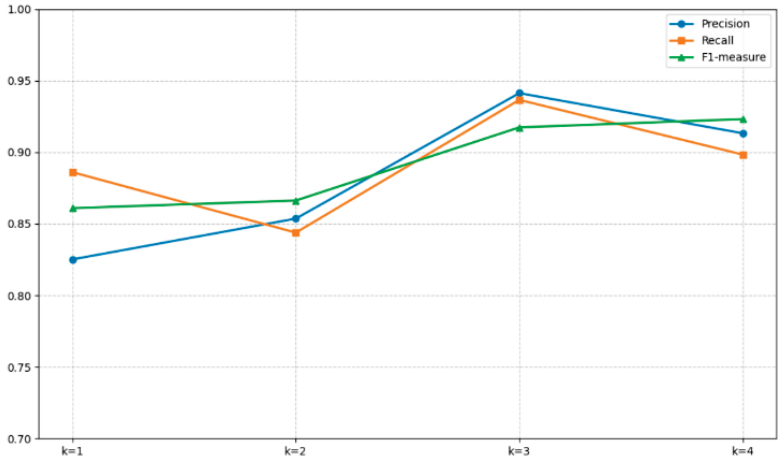


Figure 10. Analysis of influence of number of propagation layers (k) in GCN.

4.5. Ablation Study

To demonstrate the effectiveness of the proposed modules, we conducted four additional experiments to validate the efficacy of the MVCA. The experimental results on the Netflix dataset are presented in Table 6. In these experiments, we made the following modifications based on the MVCA approach: We removed the CNN module. Only the features fused by the GCN and RNN through cross-attention were used for classification. We removed the entire GCN module and its related cross-attention paths. Only the local features of the CNN and the temporal features of the RNN were used. We removed the entire RNN module and its related cross-attention paths. Only the features of the CNN and GCN were integrated. We removed the entire bidirectional cross-attention module. The features extracted by the GCN, RNN and CNN were directly concatenated and then input into the final MLP classifier.

Table 6. Comparison of experiment results between MVCA and its variants.

Method	Precision	Recall	F1-Measure
w/o Local View	0.9025 ± 0. 0148	0.8951 ± 0.0134	0.8872 ± 0.0147
w/o Sequential View	0.8380 ± 0.0152	0.8305 ± 0.0140	0.8386 ± 0.0095
w/o Structural View	0.7958 ± 0.0126	0.8106 ± 0.0134	0.8223 ± 0.0121
w/o Cross-Attention	0.8758 ± 0.0053	0.8825 ± 0.0062	0.8617 ± 0.0074
MVCA	**0.9341 ± 0.0052**	**0.9205 ± 0.0041**	**0.9322 ± 0.0038**

The best results for each metric are highlighted in bold.

As shown in Table 6, the ablation study demonstrates the key roles of each module in the MVCA framework. Based on the CNN, the local perspective can capture fine rating anomalies, such as atypical combinations of project ratings, but its limited perception range makes it vulnerable to attacks that meticulously imitate local statistical patterns. When this module is removed, the F1 score drops from 0.9322 to 0.8872. This indicates that although the local perspective provides the basic ability to detect micro-level anomalies, its independent contribution among the three single perspectives is the smallest, suggesting that

https://doi.org/10.3390/sym18030497

relying solely on local patterns is insufficient for achieving robust detection. The sequence perspective based on the GRU effectively simulates temporal dynamics and can detect sudden activities and synchronous attack windows; however, its discriminative ability is weakened by the time dispersion strategy, which dilutes the sequence cues. Removing this perspective leads to a further decrease in the F1 score to 0.8386, highlighting the importance of capturing temporal patterns for predictive results. The structural perspective based on the GCN is crucial for revealing potential collusion communities and high-order interaction patterns, but its effectiveness is weakened when attackers deliberately limit the connectivity of the graph, or when the attack group is too sparse to form detectable clusters. Its deletion resulted in the performance of the F1 evaluation metric dropping by up to 0.8223, demonstrating that it is the most critical. Finally, the bidirectional cross-attention mechanism can achieve the adaptive fusion of these perspectives by dynamically adjusting the weights of key signals. However, its performance depends on the quality of the input representations. If all perspectives can only provide weak discriminative evidence, the attention distribution will tend to be uniform, simplifying the mechanism to a shallow average value. Without this fusion mechanism, the F1 evaluation metric drops to 0.8617, indicating its key role as a coordinator. In summary, the ablation study confirms that all three modules are essential for achieving an optimal detection performance and a significant advantage.

4.6. Confusion Matrix Analysis

To conduct a more detailed evaluation of the proposed MVCA framework, we present the confusion matrix based on a representative experimental environment. Since the average fill rate of the MovieLens 10M dataset is 1.34%, we used a subset of the MovieLens dataset with a fill rate of 1.34% for the experiment. The purpose of this setting was to simulate the data distribution of the attack profile in line with the real profile to increase the difficulty of detection, as shown in Figure 11. This model achieved a large number of true positive and true negative numbers, indicating that it has a strong ability to distinguish between real users and attackers.

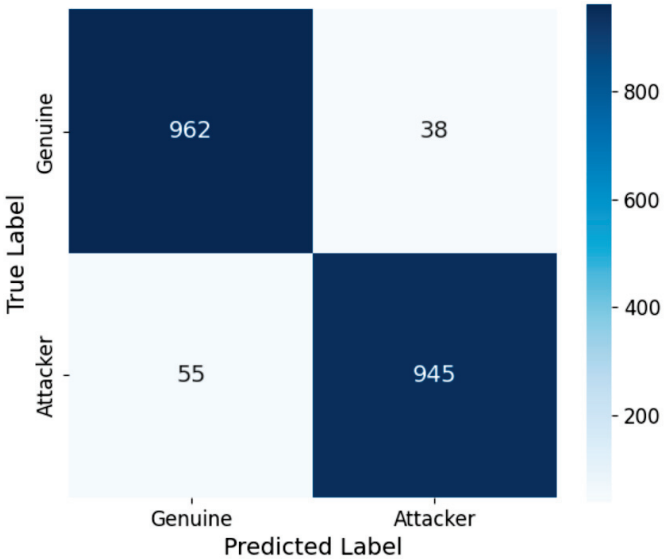


Figure 11. Confusion matrix of MVCA based on MovieLens dataset.

Based on the confusion matrix, we further compute the True Positive Rate (TPR) and False Positive Rate (FPR), two critical metrics for imbalanced detection tasks. The TPR reaches 0.945, indicating that most attack users are correctly identified, while the FPR is as low as 0.038, demonstrating the model's ability to avoid misclassifying genuine users. These results validate the robustness and reliability of MVCA in real-world deployment scenarios.

https://doi.org/10.3390/sym18030497

4.7. Time Overhead

In order to improve the overall performance of the model, a detailed analysis and representation of its runtime overhead were conducted. We define the graph structure construction and GCN feature extraction module part as Module A, the sequence modeling module and sequence feature extraction part as Module B, the CNN local feature extraction part as Module C, the interactive attention fusion module as Module D, and the detector training and prediction part as Module E. The time costs of each module under the same system conditions are shown in Table 7.

Table 7. Computational time of different modules on two datasets.

Module	Netflix	MovieLens 10M
A	3.1 min	14.1 min
B	12.3 min	51.6 min
C	2.5 min	11.5 min
D	8.7 min	45.7 min
E	1.8 min	6.9 min

The time consumption of each module is shown in Table 7. The experimental results show that the MVCA model demonstrates excellent scalability in terms of time consumption, with its computational bottlenecks mainly concentrated in the sequence modeling and cross-attention fusion modules. Although introducing higher computational costs in multi-view fusion, the resulting improvement in the detection performance is significant. In the future, the further optimization of the time efficiency through methods such as model pruning, knowledge distillation, or lightweight attention mechanisms can be considered to meet the deployment requirements of larger-scale real-time recommendation systems.

5. Conclusions

In this paper, we propose a multi-view cross-attention framework for shilling attack detection in recommender systems. The framework integrates three complementary views—user–item interaction graphs, temporal behavior sequences, and local rating patterns—and employs a bidirectional cross-attention mechanism to enable deep interaction among them. This design allows the model to capture complex and evolving attack behaviors more effectively than single-view approaches. Extensive experiments validate the superiority of MVCA towards competitive baselines. Ablation studies further validate the necessity of each module: removing any single view or the cross-attention mechanism leads to a noticeable drop in the detection performance, highlighting the importance of multi-view synergy. The experimental results show that our proposed method outperformed all other comparison methods. It demonstrated a robust performance under different attack scales and types on the MovieLens 10M dataset. Moreover, in the experiments on Netflix, the effectiveness of our method was verified.

However, this method still has some limitations. The model complexity is relatively high, and the computational overhead is large. Especially when deployed in large-scale systems, it may face efficiency challenges. Moreover, the model is highly dependent on the integrity of multi-view data. If the information of a certain view is missing or of poor quality, it may affect the overall detection performance. Future work can be carried out by optimizing the model structure and introducing lightweight design or knowledge distillation technology to enhance the efficiency and further enhance the generalization ability and practicality of the model.

https://doi.org/10.3390/sym18030497

Author Contributions: Z.Z.: funding acquisition, supervision, item administration, resources, formal analysis, writing—review and editing. C.X.: writing—original draft, writing—review and editing, methodology, software, validation. Y.L.: writing—original draft, writing—review and editing, conceptualization, visualization, methodology, formal analysis. S.S.: data curation, investigation, validation. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is partially supported by the National Nature Science Foundation of China under Grant 61906104 and Grant 61502262.

Data Availability Statement: Data is available upon request from the authors.

Acknowledgments: The authors greatly appreciate the comments from the reviewers, which helped improve the quality of this paper.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

References

1. He, X.; Liao, L.; Zhang, H. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.
2. Zhang, S.; Yao, L.; Sun, A. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.* **2019**, *52*, 1–38. [CrossRef]
3. Li, B.; Wang, Y.; Singh, A. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1893–1901. [CrossRef]
4. Masmoudi, M.; Amous, I.; Zayani, C.A.; Sèdes, F. Trust Attack Prevention Based on Spark-Blockchain in Social IoT: A Survey. *Int. J. Inf. Secur.* **2024**, *23*, 3179–3198. [CrossRef]
5. Burke, R. Multisided Fairness for Recommendation. *arXiv* **2017**, arXiv:1707.00093. [CrossRef]
6. Mehta, R.; Rana, K. A Review on Matrix Factorization Techniques in Recommender Systems. In Proceedings of the 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA), Mumbai, India, 7–8 April 2017; pp. 269–274. Available online: https://ieeexplore.ieee.org/abstract/document/8066567 (accessed on 25 January 2026).
7. Wang, X.; He, X.; Wang, M. Neural Graph Collaborative Filtering. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 165–174.
8. Tang, J.; Wang, K. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In Proceedings of the 11th ACM International Conference on Web Search and Data Mining, Marina Del Rey, CA, USA, 5–9 February 2018; pp. 565–573.
9. Gunes, I.; Kaleli, C.; Bilge, A. Shilling Attacks Against Recommender Systems: A Comprehensive Survey. *Artif. Intell. Rev.* **2014**, *42*, 767–799. [CrossRef]
10. Nawara, D.; Aly, A.; Kashef, R. Shilling Attacks and Fake Reviews Injection: Principles, Models, and Datasets. *IEEE Trans. Comput. Soc. Syst.* **2024**, *11*, 362–375. [CrossRef]
11. Wu, S.; Tang, Y.; Zhu, Y. Session-Based Recommendation with Graph Neural Networks. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 346–353. [CrossRef]
12. Sun, F.; Liu, J.; Wu, J. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 1441–1450.
13. Zayed, R.A.; Ibrahim, L.F.; Hefny, H.A. Experimental and Theoretical Study for the Popular Shilling Attacks Detection Methods in Collaborative Recommender System. *IEEE Access* **2023**, *11*, 79358–79369. [CrossRef]
14. Mehta, B.; Hofmann, T. A Survey of Attack-Resistant Collaborative Filtering Algorithms. *IEEE Data Eng. Bull.* **2008**, *31*, 14–22.
15. Rezaimehr, F.; Dadkhah, C. A Survey of Attack Detection Approaches in Collaborative Filtering Recommender Systems. *Artif. Intell. Rev.* **2021**, *54*, 2011–2066. [CrossRef]
16. Chen, R.; Hua, Q.; Chang, Y.S. A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks. *IEEE Access* **2018**, *6*, 64301–64320. [CrossRef]
17. Wang, S.; Zhang, X.; Wang, Y. Trustworthy Recommender Systems. *ACM Trans. Intell. Syst. Technol.* **2024**, *15*, 1–20. [CrossRef]
18. Lara-Gutierrez, A.; Fernandez-Gago, C.; Onieva, J.A. A Framework for Drift Detection and Adaptation in AI-Driven Anomaly and Threat Detection Systems. *Int. J. Inf. Secur.* **2025**, *24*, 199. [CrossRef]
19. Turk, A.M.; Bilge, A. Robustness Analysis of Multi-Criteria Collaborative Filtering Algorithms Against Shilling Attacks. *Expert Syst. Appl.* **2019**, *115*, 386–402. [CrossRef]
20. Huang, H.; Mu, J.; Gong, N.Z. Data Poisoning Attacks to Deep Learning Based Recommender Systems. *arXiv* **2021**, arXiv:2101.02644. [CrossRef]

21. Tian, Z.; Cui, L.; Liang, J. A Comprehensive Survey on Poisoning Attacks and Countermeasures in Machine Learning. *ACM Comput. Surv.* **2022**, *55*, 1–35. [CrossRef]
22. Mehta, B.; Nejdl, W. Unsupervised Strategies for Shilling Detection and Robust Collaborative Filtering. *User Model. User-Adapt. Interact.* **2009**, *19*, 65–97. [CrossRef]
23. Yang, F.; Gao, M.; Yu, J. Detection of Shilling Attack Based on Bayesian Model and User Embedding. In Proceedings of the IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), Volos, Greece, 5–7 November 2018; pp. 639–646.
24. Zhang, S.; Yin, H.; Chen, T. GCN-Based User Representation Learning for Unifying Robust Recommendation and Fraudster Detection. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 25–30 July 2020; pp. 689–698.
25. Zhang, Y.; Hao, Q.; Zheng, W. User Similarity-Based Graph Convolutional Neural Network for Shilling Attack Detection. *Appl. Intell.* **2025**, *55*, 340. [CrossRef]
26. Zhou, Q.; Wu, J.; Duan, L. Recommendation Attack Detection Based on Deep Learning. *J. Inf. Secur. Appl.* **2020**, *52*, 102493. [CrossRef]
27. Zhou, Q.; Huang, C. A Recommendation Attack Detection Approach Integrating CNN with Bagging. *Comput. Secur.* **2024**, *146*, 104030. [CrossRef]
28. Dou, T.; Yu, J.; Xiong, Q. Collaborative Shilling Detection Bridging Factorization and User Embedding. In Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing, Shanghai, China, 12–14 November 2017; pp. 459–469.
29. Xu, Y.; Zhang, P.; Yu, H. Detecting Group Shilling Attacks in Recommender Systems Based on User Multi-Dimensional Features and Collusive Behaviour Analysis. *Comput. J.* **2024**, *67*, 604–616. [CrossRef]
30. Harper, F.M.; Konstan, J.A. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 1–19. [CrossRef]
31. Narayanan, A.; Shmatikov, V. How to Break Anonymity of the Netflix Prize Dataset. *arXiv* **2006**, arXiv:cs/0610105.
32. Liu, H.; Ji, K.; Ma, K. Robust Recommendation-Oriented Malicious Attack Detection Method. *Inf. Sci.* **2025**, *715*, 122213. [CrossRef]
33. Maaten, L.V.D.; Hinton, G. Visualizing Data Using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.

symmetry

MDPI

Article

Compulsory Black-Box Traceable CP-ABE with Outsourcing of Computation

Ying Hu [1,*], Huidong Qiao [1], Jiangchun Ren [2], Zhiying Wang [2], Junxian Li [1] and Peng Han [1]

[1] College of Information Science and Engineering, Hunan Institute of Engineering, Xiangtan 411100, China
[2] College of Computer, National University of Defense Technology, Changsha 410073, China; jcren@nudt.edu.cn (J.R.); zywang@nudt.edu.cn (Z.W.)
* Correspondence: huying1983@hnie.edu.cn

Abstract

As an asymmetric encryption method capable of performing one-to-many encryption, the ciphertext-policy attribute-based encryption (CP-ABE) is widely recognized as an ideal cryptographic tool for cloud-based applications. It can empower data owners to independently and flexibly define and enforce access control policies for cloud-stored data. However, the practical implementation of CP-ABE-based cryptographic access control remains hindered by critical challenges. Firstly, malicious users may engage in key abuse by delegating attribute keys to unauthorized parties or exploiting their keys to construct decryption black-boxes for providing illegal decryption services. Consequently, a secure CP-ABE scheme must incorporate the capability to trace such malicious users who misuse their privileges. Secondly, for resource-constrained IoT devices, the substantial computational overhead of CP-ABE becomes prohibitive, making its deployment in scenarios like IoT-cloud services particularly challenging. In this paper, we propose a new CP-ABE scheme with black-box traceability and computational outsourcing capabilities. Our scheme can improve the tracing efficiency from $O(N^3)$ or $O(rlogN)$ (as seen in traditional schemes) to $O(1)$, where N is the number of system users. Furthermore, the proposed scheme features compulsory traceability and maintains outstanding performance in the aspects of encryption, decryption, and tracing operations.

Keywords: cryptographic access control; CP-ABE; black-box traceability

1. Introduction

In recent years, with the continuous integrated development of the Internet of Things (IoT) and cloud computing, research on related security issues has also been constantly updated and iterated. Among these studies, there is emerging research on encryption and decryption algorithms that utilize complex dynamic mechanisms to achieve efficient and secure protection of data in IoT environments—such as the three-dimensional memristive hyperchaotic mapping applicable to image encryption [1,2] and the parallel color image encryption algorithm based on neural mapping [3]. There are also ABE (Attribute-Based Encryption) algorithms used to control data access permissions within systems; these algorithms can play a role in various aspects and solve various security issues emerging in new application scenarios. Due to conflicts of interest or other reasons, a Cloud Service Provider (CSP) sometimes cannot be fully trusted by its customers. Kamara et al. [4] proposed applying cryptographic access control in cloud data management to deal with the problem. The main idea lies in encrypting sensitive data before uploading it to the

cloud. By controlling the distribution of keys, the access control policies defined by the data owner can be implemented, so that, under the condition that the CSP is semi-trusted, user-autonomous access control can be realized. A semi-trusted CSP refers to a CSP that will honestly execute all operations but may snoop on users' data. Kamara proposed leveraging one-to-many encryption mechanisms like Attribute-Based Encryption (ABE) to achieve fine-grained access control, thereby reducing key distribution costs to a manageable level. Among such mechanisms, ABE—with CP-ABE (Ciphertext-Policy Attribute-Based Encryption) in particular—is regarded as an ideal cryptographic tool for data sharing. In a CP-ABE system, each user is associated with a set of attributes. During encryption, an access control policy—defined based on these attributes—is embedded within the ciphertext. Only users whose attributes meet this access control policy can successfully decrypt the ciphertext to retrieve the plaintext. As a result, data owners need not concern themselves with users' specific identities; instead, they only need to specify the type of individuals authorized to access the data. Thanks to these advantages, since Bethencourt introduced the first CP-ABE scheme [5], the development of CP-ABE schemes has remained a central focus in research on cryptographic access control.

White-box traceability and black-box traceability. However, there are still some obstacles in the practical application of CP-ABE. One of the major issues is privilege abuse, that is, malicious users may abuse their keys and access privileges, leak them to unauthorized users in the form of new keys, or construct a Decryption Black-box (DB) to provide illegal decryption service for profits while hiding their private keys. In traditional CP-ABE schemes, it is almost impossible to track or obtain evidence for these violations. To tackle the issue of privilege misuse, Li et al. [6] pioneered an accountable CP-ABE scheme. In this approach, user-specific information is embedded into the private key during the key generation phase, enabling the identification of malicious users who engage in unauthorized key sharing through analysis of the key itself. However, the challenge of tracing the builder of a database (DB) persisted until Liu et al. [7] introduced the first CP-ABE scheme with black-box traceability. In their work [7], Liu and colleagues defined the concept of traceability explicitly and categorized it into two types: white-box traceability and black-box traceability. Notably, the scheme presented in [6] falls under the category of white-box traceability. However, it is generally believed that only black-box traceable schemes can achieve complete traceability because black-box traceability implies white-box traceability. Although the scheme in [7] achieves black-box traceability, it can only track the key-like black-box and is unable to track the *policy-specific* black-box. Subsequently, Liu put forward a scheme [8] that can track *policy-specific* black-box. Nevertheless, both of these schemes exhibit inefficiencies in encryption and decryption. Moreover, the length of the ciphertext is correlated with the number of users. Therefore, Ning et al. [9] proposed a scheme with a fixed ciphertext length to prevent the ciphertext length from expanding with the number of users. Liu proposed a scheme based on prime-order bilinear groups, which greatly reduces the computational cost and makes the application of the black-box tracing algorithm practical.

These schemes, together with subsequent black-box traceable schemes, employ the same tracing algorithm framework. Within this framework, encryption operations must be executed $O(N^3)$ times during a single tracing process, where N denotes the number of users in the system. Encryption itself carries relatively high computational costs, as it requires frequent bilinear pairing operations and exponentiation operations. To address this, Qiao et al. [10] proposed *marker tracing* to develop an efficient and lightweight tracing algorithm. In their work [10], the number of encryption operations is reduced to $O(N)$ or even $O(1)$, and this approach has also been adopted in subsequent studies such as [11–14] for tracing purposes.

Computation outsourcing capability. Another obstacle faced by CP-ABE in practical applications is the computational overhead of conventional encryption and decryption. This issue is particularly prominent in emerging scenarios such as IoT, smart healthcare, and edge computing, and has also become a core motivation for driving relevant research. From the perspective of these application scenarios, a large number of terminal devices (e.g., sensors and smart wearable devices) are limited by hardware costs and power consumption design, and they generally feature weak computing capabilities and limited storage resources. For instance, the processor frequency of a temperature monitoring sensor is usually lower than 1 GHz, and its memory is only a few MB. However, as a relatively complex asymmetric cryptographic algorithm, CP-ABE involves multiple rounds of bilinear pairing operations and attribute set matching in its encryption and decryption processes. A single device may take hundreds of milliseconds or even seconds to complete the decryption of a single ciphertext, which makes it difficult to meet the requirements of real-time data collection (e.g., industrial equipment status monitoring) and low-latency control (e.g., smart home command response). Green et al. [15] put forward a new approach for secure decryption outsourcing. By adding a blinding factor to generate the transform key for the proxy, the proxy server can perform the majority of decryption computation; meanwhile, it cannot pry into the content of the ciphertext. The user only needs to perform a simple ElGamal-type decryption at last to obtain the plaintext. Most of the later research [16–19] also adopted similar methods in [15] to transform the major workloads to the proxy server.

To achieve computational outsourcing and black-box traceability simultaneously, Yu et al. [20] proposed an accountable CP-ABE scheme with the ability of computational outsourcing, it is white-box traceable as well as the scheme in [21,22]. As far as we know, only a very few CP-ABE schemes [14,23] can achieve both black-box traceability and computational outsourcing at the same time. Zhang et al. [23] first put forward the scheme that meets the two requirements. They add operations of a subset cover algorithm [24] into encryption and decryption to meet the requirements of revocation and traceability. However, in a subset cover algorithm, encryption operations must be performed for each user subset in every data encryption. In the worst case, the number of user subsets can reach $N/2$, so that the corresponding ciphertext also contains $N/2$ ciphertext blocks, where N is the number of users. Therefore, it incurs additional costly overhead for the system. Moreover, similarly to the scheme of [7], their tracing algorithm also needs to issue a series of inquiries to gradually locate a certain subset, and then divide this subset to find the traitor within it. The tracing process is rather inefficient. Yin et al. [14] adopt the method of [10] to trace the black-box efficiently. However, in their system, the tracing algorithm can achieve successful tracking only if it obtains both the final decryption result output by the black-box and an intermediate result, which is the partially decrypted ciphertext, output by the Cloud Auxiliary Server. Thus, if the black-box completes all the decryption on its own and only outputs the final decryption result, the system will be unable to track the black-box.

Our contribution. To apply the CP-ABE system securely in scenarios with limited computation resources, such as IoT cloud, it is necessary to realize a practical black-box traceable CP-ABE scheme with outsourcing of computation (*OC-BTCP-ABE*). So far, in the few existing OC-BTCP-ABE schemes, issues such as low encryption and decryption computational efficiency, poor scalability, or limitations in the tracing algorithm can be observed. This makes it difficult to implement these schemes in reality, especially for those IoT systems with a large number of users. To obtain a secure and efficient scheme, we have proposed a new OC-BTCP-ABE scheme with the following characteristics.

- **High practicality**: Our scheme is constructed based on prime-order groups rather than composite prime-order groups, and a concise architecture similar to that of [11] is

adopted, which makes the scheme highly efficient in computational operations such as encryption and decryption. Moreover, the scheme overcomes the problem that the tracing efficiency drops rapidly with the increase in the number of system users. In our proposed scheme, the computational load of the tracing algorithm is $O(N)$, and its complexity can be further optimized to $O(1)$ in efficient tracing mode. Compared with the existing OC-BTCP-ABE schemes, our solution has significant advantages in encryption and tracing efficiency.

- **Compulsory traceability**: Our scheme possesses the compulsory traceability proposed in [10], which makes it impossible for the adversary to identify the tracing ciphertext, thus unable to counter the tracing. This ensures that the effectiveness of the tracing results cannot be compromised.

2. Background

In this section, we begin by providing background details on Linear Secret Sharing Schemes (LSSSs). Next, we outline the security definitions specific to OC-BTCP-ABE. Lastly, we formulate the formal descriptions of the decisional q-parallel Bilinear Diffie-Hellman Exponent (BDHE) assumption and bilinear groups.

2.1. Access Structure

An access structure $\mathbb{A}$ refers to a collection composed of sets of attributes. And $\mathbb{A}$ is a monotone collection, that is, $\forall B, C : if B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. The sets in $\mathbb{A}$ are authorized sets, and the sets not belonging to $\mathbb{A}$ are unauthorized sets. The formal definition of access structure is given in [25]. It should be noted that this monotone access structure can also implement a general access structure. In this case, it becomes necessary to create a "not" counterpart for each attribute, which in turn doubles the total number of attributes within the system.

2.2. Linear Secret Sharing Scheme (LSSS)

A LSSS, which is proposed in [25], is applied to implement the access structure of ciphertexts in our scheme.

Definition 1. *(Linear Secret-Sharing Scheme (LSSS)) A secret-sharing scheme* Π *over a set of parties* $\mathcal{P}$ *is called linear (over* $\mathbb{Z}_p$*) if the following hold:*

1. *The shares for each party form a vector over* $\mathbb{Z}_p$*.*
2. *There exists a matrix an* M *with* ℓ *rows and* n *columns called the share-generating matrix for* Π*. For all* $i = 1, ..., \ell$*, the function* ρ *maps the each row* M_i *of* M *to an associated attribute* $\rho(i)$*. For a vector* $v = (s, r_2, ..., r_n)$*, where* $s \in \mathbb{Z}_p$ *is the secret to be shared, and* $r_2, ..., r_n \in \mathbb{Z}_p$ *are randomly chosen,* Mv *is the vector of* ℓ *shares of the secret* s *according to* Π*. The share* $\lambda_i = (Mv)_i$ *belongs to attribute* $\rho(i)$*.*

It is shown in [25] that a share-generating matrix M can enjoy the linear reconstruction property. Suppose that there is a corresponding share-generating matrix M for an access structure $\mathbb{A}$. For an attribute set S, let $I_S = \{i | \rho(i) \in S\}$. For every set $S \in \mathbb{A}$, the "target" vector $(1, 0, 0, \ldots, 0)$ must lie within the span of the rows indexed by I_S. Specifically, for any $S \in \mathbb{A}$, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I_S}$, that can be computed in polynomial time, such that for any set of shares $\{\lambda_i\}_{i \in I_S}$ corresponding to a secret s, the equation $\sum_{i \in I_S} \omega_i \lambda_i = s$ holds. Conversely, for any unauthorized set $S' \notin \mathbb{A}$, the target vector $(1, 0, 0, \ldots, 0)$ does not lie within the span of the rows indexed by $I_{S'}$.

2.3. Bilinear Map

Assuming $\mathcal{G}$ denotes a bilinear group generator and taking a security parameter λ, $\mathcal{G}$ produces $(p, g, \mathbb{G}, \mathbb{G}_T, e)$ as the output, $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups of prime order p, where p is a big prime, and g is a generator of $\mathbb{G}$. e denotes a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and it has the following properties:

Let $\mathcal{G}$ represent a bilinear group generator. Given a security parameter λ, $\mathcal{G}$ outputs $(p, \mathbb{G}, g, \mathbb{G}_T, e)$. Here, $\mathbb{G}$ and $\mathbb{G}_T$ are both multiplicative cyclic groups of prime order p, and g serves as a generator of $\mathcal{G}$. The symbol e denotes a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$, which possesses the following properties:

1. Computability: For any elements $u, v \in \mathbb{G}$, $e(u, v)$ is efficiently computable.
2. Non-degeneracy: $e(g, g) \neq 1$.
3. Bilinearity: For any elements $u, v \in \mathbb{G}$ and exponents $a, b \in \mathbb{Z}_p$, the property $e(u^a, v^b) = e(u, v)^{ab}$ holds.

The security of our proposed scheme relies on the decisional q-parallel BDHE assumption [26]. Due to space constraints, the formal definition of this assumption is not included herein.

2.4. OC-BTCP-ABE

An OC-BTCP-ABE construction functionality includes five algorithms: Setup, Encrypt, KeyGen, Transform and Decrypt.

`Setup`$(U, \lambda) \to (PK, MK)$. This algorithm accepts security parameter λ and the attribute universe U as inputs, and outputs the public parameter PK and the master secret key MK.

`KeyGen`$(MK, S) \to (TK, SK)$. The `KeyGen` algorithm takes the master key MK as input and generates a transformation key TK and a private key SK according to the user's attribute set S.

`Encrypt`$(PK, \mathbb{A}, M) \to (CT)$. This algorithm encrypts a message M using PK under a specified access structure $\mathbb{A}$. It generates a ciphertext CT that is decryptable only by users whose attribute sets satisfy the policy of $\mathbb{A}$.

`Transform`$(CT, PK, TK) \to (CT')$. The `Transform` algorithm takes as input the ciphertext CT that was encrypted under $\mathbb{A}$, the public key PK and the transformation key TK according to the attribute set S. It outputs the partially decrypted ciphertext CT' if S satisfies $\mathbb{A}$, or it outputs $\perp$.

`Decrypt`$(CT', SK) \to (M)$. The `Decrypt` algorithm decrypts the ciphertext CT' under the user's secret key SK and outputs the message M.

2.5. Security Model for OC-BTCP-ABE

Due to differences in structure from ordinary CP-ABE schemes, with the addition of public transformation keys and the requirement of going through two steps (transformation and decryption) during the decryption process, the security model for the OC-BTCP-ABE scheme also differs from that for ordinary CP-ABE schemes. This is typically characterized by a security game played between a challenger and the adversary $\mathcal{A}$.

Setup. The challenger performs `Setup` algorithm and sends the public parameter PK to $\mathcal{A}$.

Phase 1. The challenger creates an empty table T, an empty set KS and an index $i = 0$. The adversary can adaptively make any of the following queries:

- *Create*(S): The challenger sets $i := i + 1$, and performs `KeyGen` algorithms on the attribute set S. It outputs the pair (SK, TK) and stores (i, S, TK, SK) in table T. Then, it returns TK to $\mathcal{A}$.

- $Corrupt(j)$: If the j-th entry exists in table T, the challenger gets the entry (j, S, TK, SK), updates KS as $KS := KS \cup \{S\}$, and then returns the private key SK to $\mathcal{A}$. In the absence of such an entry, it returns $\perp$ instead.

Challenge. $\mathcal{A}$ chooses two messages M_0, M_1 of equal length and gives them to the challenger. In addition, $\mathcal{A}$ gives an access structure $\mathbb{A}^*$ such that none of the attribute sets S of KS satisfy $\mathcal{A}$. The challenger flips a random coin to select $b \in \{0,1\}$. Subsequently, the challenger encrypts the message M_b under the target access structure $\mathbb{A}^*$ and provides the resulting ciphertext to the adversary $\mathcal{A}$.

Phase 2. Phase 1 is repeated with the restriction that $\mathcal{A}$ cannot issue a `Corrupt` query that would result in an attribute sets S which satisfies $\mathbb{A}^*$.

Guess. Ultimately, the adversary $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. The adversary $\mathcal{A}$ is deemed to win the game if $b' = b$. Within this game, the advantage of $\mathcal{A}$ is defined as $Pr[b' = b] - 1/2$.

Definition 2. *An OC-BTCP-ABE scheme is CPA secure (secure against chosen-plaintext attacks), if for any polynomial time adversary $\mathcal{A}$, the advantage is always negligible in the above game.*

Note that a scheme is *selectively* secure if an **Init** stage is added before **Setup** where $\mathcal{A}$ submits structure $\mathbb{A}^*$ in advance. We will prove our construction secure under the selective security model.

2.6. Traceability for CP-ABE

The black-box traceability of CP-ABE refers to the ability to uniquely identify the user who created the black-box by analyzing the black-box's output results, without being able to view the internal key of the black-box, and it should not incorrectly output a user who is not the creator. In contrast to the decryption black-boxes characterized as probabilistic devices in [7], we adopt a streamlined concept of the decryption black-box, as first proposed in [10]: A decryption black-box $\mathcal{D}$ linked to an attribute set $S_{\mathcal{D}}$ is capable of decrypting a ciphertext CT and outputting the correct plaintext M if $S_{\mathcal{D}}$ satisfies the access structure of CT; otherwise, it returns $\perp$. This formulation also simplifies the handling of collusion issues. Notably, when two malicious users collude to build a joint decryption black-box, ref. [10] shows that the decryption capability of this collusive device is no stronger than the sum of the individual decryption capabilities of the two separate black-boxes each user would construct independently. Therefore, we mainly consider the black-box independently constructed by a single malicious user.

When tracing, the tracing algorithm always needs to interact with the black-box. Similarly to all existing BT-CP-ABE schemes, in our work, the tracing algorithm functions by sending specific ciphertexts to the black-box and endeavoring to identify the black-box's creator using the information it returns. To facilitate this, we propose an additional encryption algorithm designed to generate such special ciphertexts, whose decryption results can be leveraged to pinpoint the black-box's creator. This algorithm is defined as follows:

$\texttt{Encrypt}_{Trace}(PK, \mathbb{A}, M) \rightarrow (TCT, trap)$. This algorithm takes as input the message M, the access structure $\mathbb{A}$ and the public key PK. It outputs the tracing ciphertext TCT that can be decrypted under the keys whose corresponding attributes satisfy $\mathbb{A}$. When the decryption result is returned, the $trap$ will be used to search for the private key used in decryption.

2.7. Security Model of Compulsory Traceability

To trace a black-box, it is necessary to analyze the correct decryption results of tracing ciphertexts. However, if the black-box is capable of distinguishing between normal ciphertexts and tracing ciphertexts, it might intentionally output incorrect decryption results for the latter in an attempt to evade tracing while maintaining the ability to correctly decrypt normal ciphertexts. Therefore, it must be ensured that no adversary can distinguish between tracing ciphertexts and normal ciphertexts. When this condition is met, we term it as *Compulsory Traceability*. The original definition is presented in [10] using a security game between a challenger and an adversary. The main design intuition of this game is that a user, even if his attribute sets satisfy the access policy of the ciphertext, still cannot distinguish whether the ciphertext is a tracing ciphertext or a normal one. Due to the differences from the ordinary CP-ABE structure, we present the security model description of Compulsory Traceability for the OC-BTCP-ABE scheme as follows:

Setup. The challenger executes the `Setup` algorithm and transmits the public parameter PK to the adversary $\mathcal{A}$.

Phase 1. The challenger initializes an empty set KS, an empty table T, and sets an index $i = 0$. The adversary $\mathcal{A}$ is then allowed to adaptively issue any of the following types of queries:

- $Create(S)$: The challenger sets $i := i + 1$, and performs `KeyGen` algorithms on the attribute set S. It outputs the pair (SK, TK) and stores (i, S, TK, SK) in table T. Then, it returns TK to $\mathcal{A}$.
- $Corrupt(j)$: If the j-th entry exists in table T, the challenger gets the entry (j, S, TK, SK), updates KS as $KS := KS \cup \{S\}$, and then returns the private key SK to $\mathcal{A}$. In the absence of such an entry, it returns $\perp$ instead.

Challenge. The adversary $\mathcal{A}$ submits a target access structure $\mathbb{A}^*$ to the challenger. The challenger first selects a message $M \in \mathbb{G}_T$ uniformly at random and flips a random coin to determine $b \in \{0, 1\}$. For $b = 0$, the challenger executes the standard encryption algorithm: $\texttt{Encrypt}(PK, \mathbb{A}^*, M) \rightarrow CT_0$, and outputs CT_0. For $b = 1$, it runs the tracing encryption algorithm instead: $\texttt{Encrypt}_{\text{Trace}}(PK, \mathbb{A}^*, M) \rightarrow (CT_1, \text{trap})$, and outputs CT_1. Finally, the challenger sends CT_b to $\mathcal{A}$.

Phase 2. Phase 1 is repeated.

Guess. $\mathcal{A}$ outputs a guess b' of b.

Note that $\mathbb{A}^*$ can be satisfied by at least one of the attribute sets in KS.

If $b' = b$, the adversary $\mathcal{A}$ wins the game. The advantage of $\mathcal{A}$ is defined as $Pr[b' = b] - 1/2$.

Definition 3. *An OC-BTCP-ABE scheme is compulsorily traceable, if for any polynomial time adversary $\mathcal{A}$, the advantage is always negligible in the above game.*

3. Our Construction

Our scheme is constructed by modifying the scheme of [11]. Note that U denotes the attribute universe of the system. To prevent collusion among users, we adopt the similar method of [26], assigning each user's key a random exponent r, which is embedded in the main component D of the key and in the components corresponding to each attribute. This binds together all components of a user's key, making them incompatible with the key components of other users. During decryption, each share must be computed using the same consistent exponent r to recover the shared secret and thus the plaintext, meaning components from different keys cannot be combined or used together.

3.1. Concrete Construction

`Setup`$(U, \lambda) \rightarrow (PK, MK)$. The algorithm takes the security parameter λ as input and outputs $(p, g, \mathbb{G}, \mathbb{G}_T, e)$. It then randomly selects $\alpha, \beta, t \in \mathbb{Z}_p$. Furthermore, for each attribute $x \in U$, it randomly chooses $\{h_x, f_x \in \mathbb{G}\}$. The public key is published as follows:

$$PK = (g, \mathbb{G}, \mathbb{G}_T, g^t, e(g,g)^\alpha, h = g^\beta, \{h_x, f_x\}_{x \in U})$$

The authority retains the master key $MK = (\beta, g^\alpha)$in secrecy and initializes an empty table LID.

`KeyGen`$(MK, S) \rightarrow (SK, TK, ID_{SK})$. Using the master key MK, the algorithm generates a private key SK corresponding to the attribute set S. It randomly selects exponents $k, r, z, \{r_j\}_{j \in S} \in \mathbb{Z}_p$ and obtains the transformation key $TK =$

$$(D = g^{\frac{(\alpha + rt)/\beta}{k}}, \forall j \in S : D_j = g^{r/z} \cdot f_j^{r_j/z}, D_j^{'} = g^{r_j/z}, D_j^{''} = h_j^{r/z}, D_j^{'''} = h_j^{r_j/z})$$

It then sets the private key $SK = \{k, z, TK\}$ and the key index $ID_{SK} = e(g^t, g^r)$. The authority records ID_{SK} in table LID.

`Encrypt`$(PK, \mathcal{M}, (M, \rho)) \rightarrow (CT)$. The algorithm takes as input the public key PK and encrypts the message $\mathcal{M}$ with an access control structure (M, ρ). Here, M is an $l \times n$ LSSS matrix corresponding to the size of the ciphertext access policy, and each row of the matrix can be mapped to an attribute $\rho(i)$. The algorithm selects a random vector $v = (s, v_2, ..., v_n) \in \mathbb{Z}_p^n$. Then, it randomly selects values $a_i, b_i \in \mathbb{Z}_p$ for each row M_i of M and performs calculations $\mu_i = M_i \cdot v$. It computes and outputs the ciphertext as follows:

$$CT = (C = \mathcal{M} e(g,g)^{\alpha s}, \widetilde{C} = h^s,$$

$$\forall i \in [l] : C_i = (g^t)^{u_i} h_{\rho(i)}^{a_i}, C_i^{'} = (f_{\rho(i)}^t)^{u_i} h_{\rho(i)}^{b_i}, C_i^{''} = g^{a_i}, C_i^{'''} = f_{\rho(i)}^{a_i}, C_i^{''''} = g^{b_i})$$

CT is published along with (M, ρ). The notation $[l]$ herein represents the set $\{1, 2, 3, ...l\}$.

`Transform`$(PK, TK, CT) \rightarrow (CT')$. Using a transformation key TK and the public key PK, the algorithm partially decrypts the ciphertext CT that contains the access structure (M, ρ). If S, where S is the attribute set associated with TK, does not satisfy the access structure, it outputs $\perp$. Suppose a certain subset $I \subseteq S$ satisfiesthe access policy of (M, ρ), the algorithm can obtain the set of constants $w_i \in \mathbb{Z}_p$ for which $\sum_{\rho(i) \in I} w_i M_i = \{1, 0, ..., 0\}$. It then proceeds to compute the following:

$$T_0 = e(D, \widetilde{C}) = e(g,g)^{(\alpha + rt)s/k},$$

$$T_1 = \prod_{\rho(i) \in I} \left(\frac{e(D_{\rho(i)}, C_i) e(D_{\rho(i)}^{'''}, C_i^{''''})}{e(D_{\rho(i)}^{'}, C_i^{'}) e(D_{\rho(i)}^{''}, C_i^{''}) e(D_{\rho(i)}^{'''}, C_i^{'''})}\right)^{w_i} = e(g,g)^{rts/z}$$

It then outputs the partially decrypted ciphertext $CT' = (C, T_0, T_1)$.

`Decrypt`$(SK, CT') \rightarrow (M)$. The algorithm takes as input the private key SK. It computes $M = C \frac{T_1^z}{T_0^k}$ to finish the decryption. Noted that CT' must be the intermediate result generated by decrypting with TK contained in SK; otherwise, the decryption result will be incorrect.

3.2. Tracing the Black-Box

In the process of black-box tracing, one must first send a tracing ciphertext to the black-box and then analyze the output of it to pinpoint the key employed in the decryption

process, thereby exposing the identity of the black-box's creator. In our scheme, the tracing algorithm is described as follows:

$\texttt{Encrypt}_{Trace}(\mathcal{M}, (M, \rho), PK) \rightarrow (trap, TCT)$. This algorithm encrypts the message $\mathcal{M}$ with the public key PK and an access control structure (M, ρ). The steps performed here are basically the same as those of the `Encrypt` algorithm. The main difference is that it selects two random numbers s, s' and uses s' to construct the vector $v = (s', v_2, ..., v_n) \in \mathbb{Z}_p^n$. Thus, s' instead of s is distributed in the constants u_i for $i = 1, 2, 3, ..., l$. The algorithm outputs the tracing ciphertext as follows:

$$TCT = (C = \mathcal{M}e(g,g)^{\alpha s}, \widetilde{C} = h^s,$$

$$\forall i \in [l] : C_i = (g^t)^{u_i} h_{\rho(i)}^{a_i}, C_i^{'} = (f_{\rho(i)}^t)^{u_i} h_{\rho(i)}^{b_i}, C_i^{''} = g^{a_i}, C_i^{'''} = f_{\rho(i)}^{a_i}, C_i^{''''} = g^{b_i})$$

It outputs TCT and $trap = s - s'$.

To trace a black-box $\mathcal{D}$, the authority selects an access control structure (M, ρ) that is satisfiable by the attributes set $S_{\mathcal{D}}$ of $\mathcal{D}$, as well as a message $\mathcal{M}$ randomly selected from the message space $\mathbb{G}_T$. It then runs the algorithm $\texttt{Encrypt}_{Trace}$, sends the TCT to $\mathcal{D}$, and saves the $trap$ at the same time. If $\mathcal{D}$ performs the decryption correctly, it firstly runs `Transform` and obtains the following:

$$T_0 = e(D, \widetilde{C}) = e(g,g)^{(\alpha + rt)s/k},$$

$$T_1 = \prod_{\rho(i) \in I} \left(\frac{e(D_{\rho(i)}, C_i) e(D_{\rho(i)}^{'''}, C_i^{''''})}{e(D_{\rho(i)}^{'}, C_i^{'}) e(D_{\rho(i)}^{''}, C_i^{''}) e(D_{\rho(i)}^{'''}, C_i^{'''})}\right)^{w_i} = e(g,g)^{rts'/z}$$

Then, $\mathcal{D}$ runs `Decrypt` as $M = C \frac{T_1{}^z}{T_0{}^k} = \mathcal{M}e(g,g)^{rt(s'-s)}$ and outputs $\mathcal{M}'$. When the authority receives the return $\mathcal{M}'$, it computes $E = M'/M = e(g,g)^{rt(s'-s)}$. For each entry in LID, it computes $(ID_{SK})^{trap}$ and compares it with E until a match is found. The (ID_{SK}) reveals which private key was used in the construction of the black-box. If the user ID corresponding to (ID_{SK}) is also recorded during the key generation process, the authority can directly use it to expose the identity of the black-box $\mathcal{D}$'s builder.

Efficient tracing. If the parameter s' is fixed to $s + 1$ instead of a randomly selected value, that is, the $trap$ is set to 1. Therefore, when searching LID, E can be directly compared with ID_{SK}. In this way, the step of performing exponentiation on ID_{SK} can be omitted, thereby achieving efficient search and tracking.

4. Performance Analysis

4.1. Theoretical Performance

In general, the overall performance of a CP-ABE scheme can be measured by the cost of encryption and decryption, the length of keys and ciphertexts, and the security model. Thus, we present a brief feature comparison with the only two OC-BTCP-ABE schemes known to us in Table 1.

In Table 1, N stands for the total number of users in the system, l refers to the number of rows in the LSSS matrix M, $|S|$ is the number of attributes owned by users, and $|I|$ denotes the number of attributes involved in decryption, m is the number of subsets applied in the Subset Cover Method that is adopted in [23]. T_P, T_E, T_H,T_S respectively represent the computational costs required for a pairing operation, an exponentiation operation, a hash operation and a symmetric encryption/decryption operation. $|\mathbb{G}|$, $|\mathbb{G}_T|$, $|\mathbb{Z}_p|$ refer to the size of elements in $\mathbb{G}$, $\mathbb{G}_T$, $\mathbb{Z}_p$ respectively.

Table 1. Theoretical performance analysis and comparison.

	Encrypt	Transform +Decrypt	Ciphertext Size	Key Size	Model	Compulsory Traceability
[23]	$T_P + (4l + 5)T_E + (l + 2)T_H + (l \times m)T_S$	$(3 + 2\|I\|)T_P + (\|I\| + 1)T_E + 2T_H + T_S$	$(2l + 3)\|\mathbb{G}\| + 2\|\mathbb{G}_T\| + (l \times m)\|\mathbb{Z}_p\|$	$(\|S\| + 3)\|\mathbb{G}\| + (\frac{1}{2}log^2N + \frac{1}{2}logN + 1)\|\mathbb{Z}_p\|$	Random Oracle	×
[14]	$T_P + (10l + 3)T_E + 2lT_H$	$(1 + 2\|I\|)T_P + (\|I\| + 1)T_E + 2T_H$	$(2l + 1)\|\mathbb{G}\| + 2\|\mathbb{G}_T\|$	$(\|S\| + 2)\|\mathbb{G}\| + 2\|\mathbb{Z}_p\|$	Random Oracle	×
This work	$T_P + (7l + 2)T_E$	$(1 + 5\|I\|)T_P + (\|I\| + 2)T_E$	$5l\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(4\|S\| + 1)\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	Standard	√

Computational outsourcing capability. Similarly to the Green scheme, our scheme primarily focuses on how to securely outsource decryption computations. In our scheme, the Transform operation— which performs a major part of the decryption— can be delegated to a proxy server; this process outputs the transformed ciphertext CT', and the user only needs to complete the final Decrypt operation on their own to recover the plaintext from CT'. In our scheme, the cost of the transform operation is $(1 + 5|I|)T_P + |I| \cdot T_E$, while the cost of the decrypting operation is $2T_E$. Tests show that the computational cost of T_E is generally about 10 times that of T_P. Therefore, in our scheme, most of the overhead in decryption computation can be outsourced. For example, if the decryption computation involves 20 attributes (i.e., $I = 20$), the outsourceable Transform operations account for approximately 94% of the total cost of decryption computation, while users only need to bear about 6% of the computational cost.

Efficiency. By comparison, it can be found that in terms of encryption performance, our scheme is superior to the two existing schemes, but its decryption performance is lower than these two schemes. This is because in the encryption phase, the scheme in [23] actually performs a re-encryption, while the scheme in [14] requires users to interact with the Cloud Encryption Server (CES) using a method similar to the blind signature to complete the relatively complex calculations. The lower decryption efficiency of our scheme is due to that in order to meet the compulsory traceability requirements, the secret parameters must be hidden more deeply. However, the decryption efficiency of this work is the same as that of [11], so its computational efficiency is still sufficient to meet the practical application. Additionally, it is worth noting that both our scheme and the scheme of [14] need to store a list LID, which is used to compare and search for keys during tracing. This list needs to store each user's ID_SK (the ID_SK mainly contains an element from group $\mathbb{G}_T$). In our tests, the ID_SK length is 48 bytes, so the storage overhead during tracing is approximately $48N$ bytes, where N is the number of users in the system.

Traceability. In our scheme, the tracing algorithm $\texttt{Encrypt}_{Trace}$ only needs to be executed once, and the computational cost is equivalent to that of an ordinary encryption algorithm `Encrypt`. Although one has to search through the LID by running one exponentiation operation for each entry in LID to find out the key of the malicious user, an exponentiation operation is much lighter than an `Encrypt` operation, and the number of such operations is at most $O(N)$. Moreover, if *efficient tracing* is adopted, the exponentiation operations in the searching of LID can also be omitted, and in this case, the computational cost is $O(1)$, that is almost equivalent to that of a single encryption operation. Compared with our scheme, the two existing schemes have the following defects, respectively.

- **Inefficiency of the scheme in [23].** The scheme of [23] is similar to the classic black-box tracing scheme, a series of tracing ciphertexts needs to be sent to gradually find out the secret key of the malicious user. It requires running $O(logN)$ rounds of tracing algorithms to identify a malicious user. In each round of the tracing algorithm, at most $2r-1$ encryptions (or an average of $1.25r$ encryptions) need to be executed. Therefore, the total number of encryptions in a tracing process is $O(rlogN)$, while the cost of each encryption is $T_P + (4l+5)T_E + (l+2)T_H + (l \times m)T_S$. Since r denotes the number of revoked users in the system, this indicates that when the number of revoked users increases, a rather high computational overhead will be incurred in the tracing process.
- **Tracing failure of the scheme in [14].** For the scheme of [14], if the black-box performs the `Transform` and `Decryption` on its own instead of relying on the Cloud Auxiliary Server (CAS) to perform the `Transform` operation, the black-box will only output the final result, and in this case, the tracing will fail.

Finally, compared with the previous two schemes, our scheme further possesses the feature of *compulsory traceability*.

Security. The security of both existing OC-BTCP-ABE schemes and our scheme is based on the decisional q-parallel BDHE assumption. However, these two schemes are constructed under the random oracle model, whereas our scheme is constructed under the standard model.

4.2. Performance Measurements

For the CP-ABE system, encryption and decryption are the two most frequently executed operations; thus, the computational performance of the system mainly depends on the efficiency of these two operations. We implemented the scheme of [14,23], and our proposed scheme for testing, compared the test results, and presented the specific details in Figure 1.

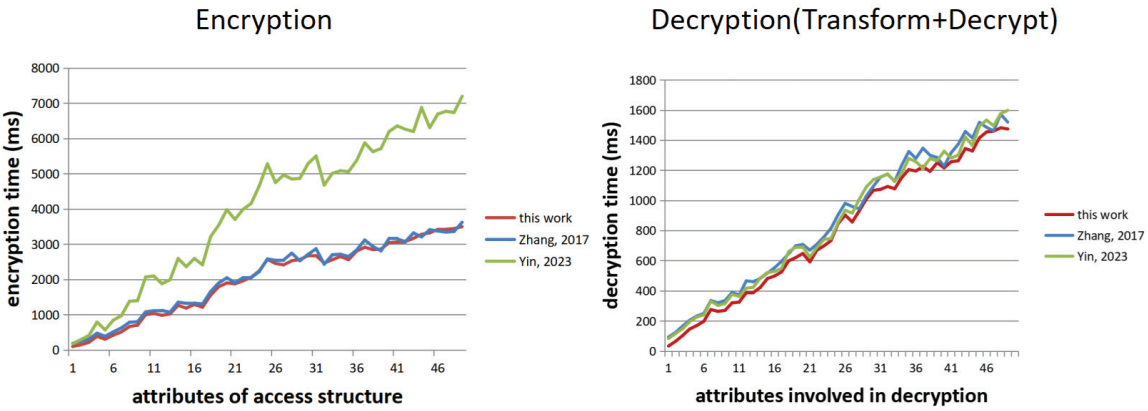


Figure 1. Performance of encryption and decryption. Zhang states that the scheme is from [23], and Yin states that the scheme is from [14].

All experiments were conducted on a PC equipped with a Core i5-12400 processor and 8 GB RAM. We utilize an elliptic curve group of Type A defined by the equation $y^2 = x^3 + x$, where the element representations are 512 bits in length and the group has an order of 160 bits. The scheme is implemented based on the Java Pairing Based Cryptography (JPBC version 2.0.0) library [27].

In the experiment, we set the access control structure in the form $(A_1$ and A_2 and ... and $A_s)$, where A_i is an attribute, in order to maintain the consistency of the decryption process and prevent potential differences that may arise when decrypting with different decryption keys.

We tested all cases where the number of attributes s ranged from 1 to 50. For each test point, we experimented 100 times and then took the average value.

The experimental results are consistent with the performance expectations from theoretical analysis, and the computational costs of encryption and decryption are proportional to the number of attributes involved in the operations. It should be noted that although more pairing operations are performed in the decryption process of our scheme, the efficiency of decryption operations in our scheme does not show a significant decrease compared with existing schemes. It is because the efficiency of pairing operations is far higher than that of exponentiation operations. Our experimental tests show that the average time for one exponentiation operation in $\mathbb{G}_T$ is approximately 12 ms, while the time for one pairing operation is about 1 ms.

As can be seen from Figure 2, the time required for tracing in our scheme is basically independent of the scale of system users. This is because only one tracing ciphertext needs to be generated for each tracing process in our scheme. The scheme of [14] is similar to our scheme. However, the number of tracing ciphertexts generated by the scheme of [23] is $O(rlogN)$, where N represents the number of users in the system and r represents the number of revoked users in the system (r was set to 50 in the test). Therefore, the tracing cost of [23] is much higher than that of our scheme, and it will increase rapidly as the number of users and revoked users increases.

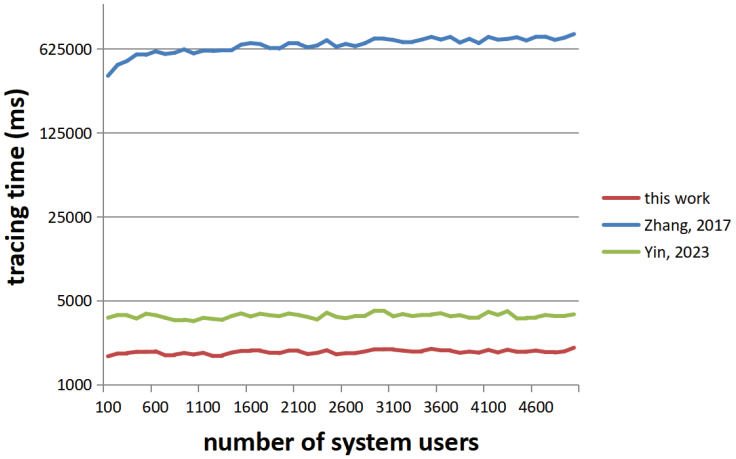


Figure 2. Performance of tracing. Zhang states that the scheme is from [23], and Yin states that the scheme is from [14].

5. Security Proof

Theorem 1. *Suppose the construction of [11] is a selectively CPA secure scheme. Then our scheme presented above is selectively CPA secure.*

Proof of Theorem 1. Suppose there is a polynomial-time adversary $\mathcal{A}$ that can win with non-negligible advantage $Adv_{\mathcal{A}}$ in the selective security game against our scheme. Then we show how to build a simulator $\mathcal{B}$ that can win in the selective security game against the scheme of [11] with the advantage $Adv_{\mathcal{A}}$. Since the scheme of [11] is selectively CPA-secure under the decisional q-parallel BDHE assumption, thus, a contradiction arises, and therefore the theorem is proven. The simulator operates as follows.

Init. The simulator $\mathcal{B}$ invokes the adversary $\mathcal{A}$. $\mathcal{A}$ gives the challenge access structure (M^*, ρ^*), where M^* is an $l \times n$ LSSS matrix. $\mathcal{B}$ then sends (M^*, ρ^*) to the challenger of [11] as the access control structure it intends to challenge.

Setup. The simulator $\mathcal{B}$ queries the challenger of [11] and obtains the public key $PK = (g, \mathbb{G}, \mathbb{G}_T, g^t, e(g,g)^\alpha, h = g^\beta, \{h_x, f_x\}_{x \in U})$ It then sends PK to $\mathcal{A}$.

Phase 1. The simulator $\mathcal{B}$ creates an empty table T, an empty set KS and an index $i = 0$. It responds to the adversary's request as follows:

- *Create*(S): $\mathcal{B}$ sets $i := i + 1$, and it continues to operate in one of the following two ways according to the situation.
  - If S satisfies the access policy contained in (M^*, ρ^*), the simulator $\mathcal{B}$ forms the transformation key TK as follows. $\mathcal{B}$ randomly chooses $z', k', r', \{r'_j\}_{j\in S} \in \mathbb{Z}_p$ and implicitly sets $k = \frac{\alpha}{\beta k'}, r = \frac{r'\alpha}{k'}, z = \frac{r'\alpha}{k'z'}, r_j = \frac{r'_j r'\alpha}{k'z'}$ by generating

$$TK = (D = g^{k'}(g^t)^{r'}, \forall j \in S : D_j = g^{z'} \cdot f_j^{r'_j}, D'_j = g^{r'_j}, D''_j = h_j^{z'}, D'''_j = h_j^{r'_j}).$$

    It then sets $SK = (k', z', TK)$. Note that SK is not a well-formed private key, but in the view of adversary $\mathcal{A}$, TK is properly distributed at random.
  - Otherwise, $\mathcal{B}$ calls the key generation oracle of [11] to get

$$SK' = (D = g^{\frac{\alpha+rt}{\beta}}, \forall j \in S : D_j = g^r \cdot f_j^{r_j}, D'_j = g^{r_j}, D''_j = h_j^r, D'''_j = h_j^{r_j}).$$

    It then randomly chooses $z, k, \in \mathbb{Z}_p$ and computes

$$TK = (D = g^{\frac{(\alpha+rt)/\beta}{k}}, \forall j \in S : D_j = g^{r/z} \cdot f_j^{r_j/z}, D'_j = g^{r_j/z}, D''_j = h_j^{r/z}, D'''_j = h_j^{r_j/z}).$$

    $\mathcal{B}$ then sets $SK = (k, z, TK)$.

  The simulator $\mathcal{B}$ stores (i, S, SK, TK) in table T and returns TK to the adversary $\mathcal{A}$.
- *Corrupt*(j): If there exists an jth entry in table T, the simulator $\mathcal{B}$ obtains the entry (j, S, SK, TK) and sets $KS := KS \cup \{S\}$. It then returns the private key SK to $\mathcal{A}$. If no such entry exists, then it returns $\perp$. Note that in the selective security game $\mathcal{A}$ cannot query for any key that satisfies the challenge structure (M^*, ρ^*).

Challenge. The adversary $\mathcal{A}$ chooses two messages M_0, M_1 of equal length and gives them to $\mathcal{B}$. $\mathcal{B}$ passes them to the challenger of [11] and obtains the returned ciphertext $CT = (C, \widetilde{C}, \forall i \in [l] : C_i, C'_i, C''_i, C'''_i, C''''_i)$. It then gives CT to $\mathcal{A}$.

Phase 2. Phase 1 is repeated.

Guess. $\mathcal{A}$ finally outputs a guess $b' \in \{0, 1\}$ of b. Then, the simulator $\mathcal{B}$ also outputs b' as its guess. The simulator $\mathcal{B}$ provides a perfect simulation. If $\mathcal{A}$ guesses correctly and wins the game, $\mathcal{B}$ will also win in the security game challenging scheme of [11]. Therefore, $\mathcal{B}$ wins the selective security game against the scheme of [11] with the non-negligible advantage $Adv_{\mathcal{A}}$. So we complete the proof. □

Theorem 2. *Suppose the scheme of [11] is compulsorily traceable. Then our scheme presented above is compulsorily traceable.*

Proof of Theorem 2. Suppose there exists a polynomial-time adversary $\mathcal{A}$ that can play the compulsory traceability security game against our scheme with a non-negligible advantage $TRAdv_{\mathcal{A}}$. Then we show how to build a simulator $\mathcal{B}$ that can win in the compulsory traceability security game against the scheme proposed in [11] with the same advantage. In [11], the advantage of any adversary is proved to be $O(n^2/p)$, where n denotes the total number of group elements that the adversary can obtain from its queries, and p is the group order. This indicates that if the group order p is sufficiently large, the adversary's advantage is negligible, and in this case, our scheme is compulsorily traceable. The simulator operates as follows.

Setup. The simulator $\mathcal{B}$ queries the challenger of [11] and obtains the public key $PK = (g, \mathbb{G}, \mathbb{G}_T, g^t, e(g,g)^\alpha, h = g^\beta, \{h_x, f_x\}_{x \in U})$ It then sends PK to $\mathcal{A}$.

Phase 1. The simulator $\mathcal{B}$ creates an empty table T, an empty set KS and an index $i = 0$. It responds to the adversary's request as follows:

- $Create(S)$: $\mathcal{B}$ sets $i := i + 1$, and it continues to operate in one of the following two ways according to the situation. $\mathcal{B}$ calls the key generation oracle of [11] to get

$$SK' = (D = g^{\frac{\alpha + rt}{\beta}}, \forall j \in S : D_j = g^r \cdot f_j^{r_j}, D_j' = g^{r_j}, D_j'' = h_j^r, D_j''' = h_j^{r_j}).$$

  It then randomly chooses $z, k, \in \mathbb{Z}_p$ and computes

$$TK = (D = g^{\frac{(\alpha + rt)/\beta}{k}}, \forall j \in S : D_j = g^{r/z} \cdot f_j^{r_j/z}, D_j' = g^{r_j/z}, D_j'' = h_j^{r/z}, D_j''' = h_j^{r_j/z}).$$

  $\mathcal{B}$ then sets $SK = (k, z, TK)$. The simulator $\mathcal{B}$ stores (i, S, SK, TK) in table T and returns TK to the adversary $\mathcal{A}$.
- $Corrupt(j)$: If there exists an jth entry in table T, the simulator $\mathcal{B}$ obtains the entry (j, S, SK, TK) and sets $KS := KS \cup \{S\}$. It then returns the private key SK to $\mathcal{A}$. If no such entry exists, then it returns $\perp$.

Challenge. $\mathcal{A}$ submits an access control structure (M^*, ρ^*) to $\mathcal{B}$. $\mathcal{B}$ passes (M^*, ρ^*) to the challenger of [11] and obtains the ciphertext CT_b output by this challenger. $\mathcal{B}$ then sends CT_b to $\mathcal{A}$.

Phase 2. Phase 1 is repeated.

Guess. $\mathcal{A}$ outputs a guess b' of b. $\mathcal{B}$ also outputs b' as its guess. The simulator $\mathcal{B}$ provides a perfect simulation. If $\mathcal{A}$ guesses correctly and wins the game, $\mathcal{B}$ will also win. Thus, the advantage of the simulator in its security game against the scheme in [11] is equal to that of the adversary $\mathcal{A}$ in the game against our scheme. So we complete the proof.

Note that in the same manner, we can prove that when s' is set to $s + 1$, the adversary is also unable to distinguish whether the ciphertext is a tracing ciphertext or an ordinary ciphertext. Therefore, in the case of efficient tracing, the property of compulsory traceability also holds. □

6. Conclusions

Although CP-ABE is widely recognized as an optimal access control mechanism enabling user-centric autonomy in cloud data services, its practical deployment still necessitates addressing critical challenges related to security and efficiency. From a security perspective, preventing key abuse by users—particularly tracing malicious users who exploit their keys to establish a decryption black-box for providing illegal decryption services for profit—remains a focal point of relevant research. In terms of efficiency, due to the substantial computational overhead inherent in CP-ABE schemes, developing secure computation outsourcing solutions for resource-constrained devices (e.g., IoT devices) has also garnered significant attention. To the best of our knowledge, only a handful of existing CP-ABE schemes [14,23] currently achieve the simultaneous implementation of black-box traceability and computation outsourcing capabilities. However, among these schemes, there are still issues of low computational efficiency or potential failure of tracing. Therefore, we propose a new OC-BTCP-ABE scheme. This scheme not only achieves black-box traceability and secure outsourcing of decryption computations but also features high computational efficiency, with a significant improvement, particularly in the efficiency of the tracing algorithm. Additionally, we prove that the scheme possesses the property of compulsory traceability, which ensures that the tracing algorithm can always function

effectively. Our scheme will be particularly suitable for access control applications in scenarios such as the IoT cloud and similar environments.

However, there are still aspects that need improvement for our scheme. For instance, the proposed scheme lacks support for efficient attribute revocation, and this practical barrier needs to be addressed in future research. Similarly, in terms of security, future studies should further consider and handle the following issues: how to defend against key compromise impersonation attacks when the authority issues attribute keys to users; how to realize policy-hiding; and how to assess the threats of quantum computing to the underlying primitives of the scheme.

Author Contributions: Conceptualization, Y.H. and H.Q.; methodology, Y.H.; software, H.Q.; validation, J.L. and P.H.; investigation, J.L. and P.H.; writing—original draft preparation, Y.H.; writing—review and editing, Y.H., H.Q., Z.W., and J.R.; supervision, Z.W. and J.R. All authors have read and agreed to the published version of the manuscript.

Funding: The research is funded in part by the National Natural Science Foundation of China, under grant no. 61303191. It is also supported by the Research Foundation of the Education Bureau of Hunan Province, China (grant no. 22B0735 and no. 24A0532) and the Construct Program of Applied Specialty Disciplines in Hunan Province (Hunan Engineering University), as well as the Hunan Provincial College Students' Innovation Training Program Project, China (Project Name: "Research and Implementation of an AIoT-based Intelligent Children's Desk System" and "Research on Key Technologies of Unmanned Mining Locomotives").

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

Abbreviations

The following abbreviations are used in this manuscript:

CSP	Cloud Service Provider
ABE	Attribute-Based Encryption
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
OC-BTCP-ABE	Black-Box Traceable CP-ABE with Outsourcing of Computation
DB	Decryption Black-box
IoT	Internet of Things
LSSS	Linear Secret Sharing Scheme
BDHE	Bilinear Diffie-Hellman Exponent
CES	Cloud Encryption Server

References

1. Gao, S.; Iu, H.H.; Erkan, U.; Simsek, C.; Toktas, A.; Cao, Y. A 3D Memristive Cubic Map With Dual Discrete Memristors: Design, Implementation, and Application in Image Encryption. *IEEE Trans. Circuits Syst. Video Technol.* **2025**, *35*, 7706–7718. [CrossRef]
2. Gao, S.; Iu, H.H.; Mou, J.; Erkan, U.; Liu, J.; Wu, R.; Tang, X. Temporal action segmentation for video encryption. *Chaos Solitons Fractals* **2024**, *183*, 114958. [CrossRef]
3. Gao, S.; Liu, J.; Iu, H.H.; Erkan, U.; Zhou, S.; Wu, R.; Tang, X. Development of a video encryption algorithm for critical areas using 2D extended Schaffer function map and neural networks. *Appl. Math. Model.* **2024**, *134*, 520–537. [CrossRef]
4. Kamara, S.; Lauter, K. Cryptographic cloud storage. In *International Conference on Financial Cryptography and Data Security*; Springer: Berlin/Heidelberg, Germany, 2010.
5. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 20–23 May 2007.
6. Li, J.; Ren, K.; Kim, K. A2BE: Accountable Attribute-Based Encryption for Abuse Free Access Control. *Cryptol. Eprint Arch.* **2009**, 1–16.

7. Liu, Z.; Cao, Z.; Wong, D.S. Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on eBay. In Proceedings of the ACM Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013; pp. 475–486.
8. Liu, Z.; Cao, Z.; Wong, D.S. Traceable CP-ABE: How to trace decryption devices found in the wild. *IEEE Trans. Inf. Forensics Secur.* **2014**, *10*, 55–68. [CrossRef]
9. Ning, J.; Cao, Z.; Dong, X.; Gong, J.; Chen, J. Traceable CP-ABE with short ciphertexts: How to catch people selling decryption devices on ebay efficiently. In *European Symposium on Research in Computer Security; Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2016; Volume 9879, pp. 551–569.
10. Qiao, H.; Ren, J.; Wang, Z.; Ba, H.; Zhou, H. Compulsory traceable ciphertext-policy attribute-based encryption against privilege abuse in fog computing. *Future Gener. Comput. Syst.* **2018**, *88*, 107–116. [CrossRef]
11. Qiao, H.; Ba, H.; Zhou, H.; Wang, Z.; Ren, J.; Hu, Y. Practical, provably secure, and black-box traceable CP-ABE for cryptographic cloud storage. *Symmetry* **2018**, *10*, 482. [CrossRef]
12. He, X.; Li, L.; Peng, H. An enhanced traceable CP-ABE scheme against various types of privilege leakage in cloud storage. *J. Syst. Archit.* **2023**, *136*, 102833. [CrossRef]
13. Liu, Z.; Ding, Y.; Yuan, M.; Wang, B. Black-box accountable authority cp-abe scheme for cloud-assisted e-health system. *IEEE Syst. J.* **2022**, *17*, 756–767. [CrossRef]
14. Yin, Y.; Gan, Q.; Zuo, C.; Liu, N.; Wang, C.; Jiang, Y. A Revocable Outsourced Data Accessing Control Scheme with Black-Box Traceability. In Proceedings of the 18th International Conference on Information Security Practice and Experience, Copenhagen, Denmark, 24–25 August 2023; pp. 380–398.
15. Green, M.; Hohenberger, S.; Waters, B. Outsourcing the Decryption of ABE Ciphertexts. In Proceedings of the 20th USENIX Security Symposium, San Francisco, CA, USA, 8–12 August 2011.
16. Li, J.; Huang, X.; Li, J.; Chen, X.; Xiang, Y. Securely outsourcing attribute-based encryption with checkability. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *25*, 2201–2210. [CrossRef]
17. Li, J.; Li, J.; Chen, X.; Jia, C.; Lou, W. Identity-based encryption with outsourced revocation in cloud computing. *IEEE Trans. Comput.* **2015**, *64*, 425–437. [CrossRef]
18. Parno, B.; Raykova, M.; Vaikuntanathan, V. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In Proceedings of the 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, 19–21 March 2012.
19. Guo, R.; Yang, G.; Shi, H.; Zhang, Y.; Zheng, D. O3-R-CP-ABE: An efficient and revocable attribute-based encryption scheme in the cloud-assisted IoMT system. *IEEE Internet Things J.* **2021**, *8*, 8949–8963. [CrossRef]
20. Yu, G.; Ma, X.; Cao, Z.; Zeng, G.; Han, W. Accountable CP-ABE with public verifiability: How to effectively protect the outsourced data in cloud. *Int. J. Found. Comput. Sci.* **2017**, *28*, 705–723. [CrossRef]
21. Sethi, K.; Pradhan, A.; Bera, P. Practical traceable multi-authority CP-ABE with outsourcing decryption and access policy updation. *J. Inf. Secur. Appl.* **2020**, *51*, 102435. [CrossRef]
22. Ziegler, D.; Marsalek, A.; Palfinger, G. White-box traceable attribute-based encryption with hidden policies and outsourced decryption. In Proceedings of the 20th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Shenyang, China, 20–22 October 2021.
23. Zhang, R.; Hui, L.; Yiu, S.; Yu, X.; Liu, Z.; Jiang, Z.L. A traceable outsourcing cp-abe scheme with attribute revocation. In Proceedings of the 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, 11 September 2017.
24. Naor, D.; Naor, M.; Lotspiech, J. Revocation and tracing schemes for stateless receivers. In Proceedings of the Advances in Cryptology–CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001.
25. Beimel, A. Secure Schemes for Secret Sharing and Key Distribution. Ph.D. Thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
26. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. *Lect. Notes Comput. Sci.* **2011**, *6571*, 53–70.
27. De Caro, A.; Iovino, V. JPBC: Java pairing based cryptography. In Proceedings of the 16th IEEE Symposium 595 on Computers and Communications, ISCC 2011, Kerkyra, Greece, 28 June–1 July 2011; pp. 850–855.

symmetry

MDPI

Article

Practical Verifiable Time-Lock Puzzle: Pre- and Post-Solution Verification

Zheyi Wu, Haolin Liu and Lei Wang *

School of Electronics Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; wuzheyi@sjtu.edu.cn (Z.W.); liuhaolinsjtu@sjtu.edu.cn (H.L.)
* Correspondence: wanglei_hb@sjtu.edu.cn

Abstract: A time-lock puzzle encapsulates a secret message such that the receiver needs to perform a sequential computation, which takes a specified amount of time, to recover the message. Time-lock puzzles can be used in various scenarios, such as sealed-bid auctions, fair contract signing, and so on. The time required to generate a time-lock puzzle and the time needed to solve it are asymmetric, making the verification of a time-lock puzzle crucial. Before solving the puzzle, the solver needs to verify the validity of the puzzle to avoid computing invalid time-lock puzzles. After the puzzle has been solved, it is essential for a third party to confirm the correctness of the solution. This paper proposes a framework for time-lock puzzles, providing both pre-verification and post-verification functionalities, and outlines the security requirements of this framework. Furthermore, we present a practical construction based on iterated squaring in the RSA group and analyze the security of the specific construction. Finally, we implement this construction in Python and demonstrate its efficiency in different settings when implemented in practice.

Keywords: iterated squaring; post-verification; pre-verification; time-lock puzzle

1. Introduction

The concept of time-lock puzzles was initially proposed by Rivest et al. [1] in 1996. A time-lock puzzle addresses the challenge of sending a secret message to the future, ensuring that the receiver cannot access the message until a predetermined amount of time has passed. This challenge was first identified by May [2] in the Cyberpunks community, where a solution using trusted agents as third parties to preserve secret messages was proposed. Time-lock puzzles are proposed as an alternative approach to address this problem without relying on a trusted third party. Loosely speaking, in a time-lock puzzle scheme, the sender encapsulates a secret message within a time-lock puzzle that requires the receiver to engage in a sequential computation to reveal the secret message. The evaluation time of this sequential computation is determined by a time parameter in the time-lock puzzle. Even with substantial parallel processing capabilities, the receiver cannot significantly reduce the required time.

Time-lock puzzles have a wide range of applications, including sealed-bid auctions [3–6], fair contract signing [7–9], electronic voting [10,11], zero-knowledge argument [12], and non-malleable commitment [13]. For instance, a sealed-bid auction consists of two phases. In the bidding phase, all bidders submit their encapsulated bids, which cannot be accessed by other bidders. In the opening phase, all bidders reveal their bids. However, the robustness of the protocol can be compromised if a bidder refuses or fails to reveal its bid during the opening phase. Time-lock puzzle provides a method to handle this problem [14,15]. During the bidding phase, bidders encapsulate their bids into time-lock puzzles, which guarantees the confidentiality of these bids since the puzzles cannot be solved before the specified time. In the subsequent opening phase, bidders are required to reveal their bids. If a malicious bidder refuses to do so, its bid can still be revealed by solving the corresponding time-lock puzzle.

We categorize the verification process of the time-lock puzzle scheme into two stages, depending on when it occurs: pre-verification and post-verification. Pre-verification takes place before the protocol's execution. For instance, before a message is used in the protocol execution, the message receiver can authenticate the identity of the sender using a signature scheme. On the other hand, post-verification is executed after the execution of the protocol. It involves the capacity of the verifier to verify the execution outcome. For example, the verifier confirms that the obtained output is accurately evaluated by the protocol and satisfies the claimed properties.

This paper focuses on the verifiability of time-lock puzzle schemes, which is essential to ensure that the protocol behaves as intended and maintains its expected properties, particularly in the presence of potential adversarial activities. In time-lock puzzle schemes, there is an asymmetry between the time required to generate the puzzle and the time needed to solve it. The generation of the time-lock puzzle is efficient, whereas solving it is time-consuming. Consequently, the adversary may exploit this by transmitting invalid time-lock puzzles, thereby depleting the computational resources of the receiver.

1.1. Our Contributions

The initial construction of the time-lock puzzle [1] lacks a verification algorithm, resulting in a lack of verifiability. The verifiability of a time-lock puzzle is important when applied in sealed-bid auctions. When a time-lock puzzle is used in sealed-bid auctions, it is necessary for the receiver to verify the legitimacy of a time-lock puzzle once a bidder has encapsulated its bid. This verification process is essential to detect malicious behavior that would prevent the time-lock puzzles from being opened correctly in the opening phase. On the other hand, after the time-lock puzzle is solved, the revealed bid must be publicly verifiable, ensuring that everyone can verify the correctness of this bid.

A typical method of providing pre-verification for the time-lock puzzle is to sign the puzzle. The receiver can verify the identity of the sender by checking the signature. After solving the puzzle, the receiver needs to generate a proof to convince a third party that the solution is correct. However, the validity of the puzzle itself should be verified, since a malicious puzzle generator could send invalid time-lock puzzles to the receiver. Moreover, generating a proof for post-verification brings an additional cost for the receiver.

In this paper, we propose a framework that addresses the two challenges outlined above. Since the receiver needs to check the identity of the sender, a signature scheme is required in the time-lock puzzle (implicitly). In our framework, rather than signing the puzzle itself, the sender signs the commitment of the secret value within the time-lock puzzle. This commitment can be used to provide pre-verification of the puzzle, since they are related to the same value. After solving the puzzle, a proof is necessary to help a third party to verify the correctness of the solution. In our framework, the signed commitment serves as this proof, as it is authenticated by the sender. Formally, for a secret message, m, the puzzle generator encapsulates it into a time-lock puzzle, z. It also generates a commitment value, x, for the message, m, and then signs it to produce the signature, σ. The generator sends the puzzle, z, the commitment value, x, and the signature, σ, to the receiver. If the puzzle generator is honest, then the commitment value, x, and the puzzle, z, will correspond to the same message. Consequently, the receiver can verify the validity of the puzzle before it starts solving the puzzle. Once the puzzle passes the pre-verification, the receiver solves it by spending a certain amount of sequential computation. For the secret message, m, revealed from the puzzle, a third party can verify its correctness using the commitment value, x, eliminating the need for the receiver to generate additional proof.

In simple terms, the framework should satisfy the following properties:

- *Correctness*: For all correctly generated time-lock puzzles, they can pass pre-verification and can be resolved to the corresponding secret message.
- *Sequentiality*: All receivers need to spend a certain amount of time to solve the time-lock puzzle, even with substantial parallel processing capabilities.

- *Unforgeability*: The probability that the adversary can generate a forged time-lock puzzle (or a forged solution) that can pass the pre-verification (or post-verification) is negligible.
- *Verifiability*: The validity of the time-lock puzzle and the solution should be verifiable before and after solving the puzzle, respectively.

A widely used method to construct a time-lock puzzle is based on iterated squaring. So, according to our proposed framework, we present a specific construction based on iterated squaring on an RSA group. In our framework, the signature scheme is used to sign on the commitment of the secret value within the time-lock puzzle. This signature provides the identity of the sender, ensuring that the message has not been modified during the transformation. In our framework, we do not restrict the specific signature scheme. A hash-then-sign structure signature can be used in our framework. In addition, we demonstrate that the construction satisfies the desired properties and analyze its efficiency theoretically. Furthermore, we implement the time-lock puzzle construction in Python and analyze its performance in different settings. We will demonstrate that the solving time of the puzzle increases as the time parameter increases, while the generation time of the puzzle, the verification time of the puzzle, and the verification time of the solution remain relatively unchanged. Additionally, the solving time of the puzzle is significantly larger than the other times. For example, when the security parameter is 2048 and the time parameter is 2^{21}, the receiver needs 21.7 s to solve the puzzle, while it takes 1.25 s to verify the validity of the puzzle.

1.2. Related Works

1.2.1. Timed Commitments

In a timed commitment scheme, the committer publishes a timed commitment, which can be forced open if the committer refuses to open the commitment. The receiver should be able to verify the "well-formedness" of the message before it starts a time-consuming algorithm. This problem was first considered in timed commitment [7] and following works [16,17]. In a timed commitment scheme, the committer generates a commitment string c and a time-related element, u. The commitment string, c, can be revealed by the receiver without the help of the committer. The receiver can calculate a string from a time-related element, u, and use the string to reveal the commitment. The committer generates a proof, π, to prove that the time-related element, u, is correctly generated and sends the tuple, (c, u, π), to the receiver. The receiver checks that, u, is exactly the value declared by the proof, π. However, the relationship between c and u is not verified, an attacker can change c to another value, c', while keeping the elements u, π as the same as the correct one. This faked tuple (c', u, π) can also pass the verification, but the commitment has already been changed. Hence, this verification is insufficient to convince the receiver before executing the forced open algorithm. In EuroS&P 2022, Manevich et al. [18] considered the construction of attribute verifiable timed commitment. It requires the committer to generate an interactive proof to convince the receiver that the committed value satisfies some attributes.

1.2.2. Homomorphic Time-Lock Puzzle

In ESORICS 2022, Liu et al. [19] considered the pre-verification for homomorphic time-lock puzzles. The puzzle generator provides a zero-knowledge proof to show the existence of a solution for the puzzle. It requires a trusted setup to generate the public parameters. The relationship between the time-related elements in public parameters is implicitly considered in their construction. However, this construction becomes insecure when considering the scenario in which the puzzle generator generates these public parameters. A malicious generator may generate the wrong elements to defraud the receiver. So, it is imperative to verify the correctness of the public parameters *explicitly* by the puzzle receiver. In PKC 2023, Srinivasan et al. [20] considers batchable time-lock puzzle to improve scalability when there are a large number of time-lock puzzles that need to be solved.

1.2.3. Verifiable Timed Signatures

In ACM CCS 2020, Thyagarajan et al. [21] proposed verifiable timed signature and utilized a cut-and-choose mechanism to provide verifiability. The signature is sent to the receiver using a k-out-of-n secret sharing scheme, with each share encapsulated within a time-lock puzzle. Subsequently, the receiver requires the generator to open $k-1$ randomly selected time-lock puzzles to show that these shares are correctly encapsulated. If all $k-1$ puzzles pass verification, then the receiver proceeds to solve the remaining $n-k+1$ time-lock puzzles to retrieve the signature based on k or more shares. However, this protocol suffers a security-efficiency trade-off. A potential threat arises from a malicious sender generating $k-1$ correct shares, while the remaining shares are incorrect. The protocol's vulnerability lies in the small probability that the receiver selects all $k-1$ pieces as the correctly generated ones, resulting in successful verification but an incorrect signature retrieval. Consequently, it requires a large n to improve the security bound, resulting in a long time verification. In ESORICS 2022, Thyagarajan et al. [22] proposed a verifiable timed linkable ring signature. This construction can be used to hide a linkable ring signature for a predetermined amount of time in a verifiable way. This is a special time signature and can be used to enhance the privacy in cryptocurrency.

1.2.4. Verifiable Delay Functions

In verifiable delay functions (VDFs) [23–26], the output is evaluated from the input after computation for a certain amount of time. After the evaluation, the evaluator has obtained the output and a proof. A verifier can use the proof to *efficiently* verify that the output is correctly generated. The verifiable delay functions can be used to provide post-verification for a time-lock puzzle scheme. The post-verification for a time-lock puzzle scheme means that a third party not solving the puzzle should be able to verify the correctness of the solution. A naive method of verifying the correctness is to repeat the solving process; this is an inefficient approach. So, the running time for the post-verification should be much less than the evaluation time. This property is considered in [27–29] as public verifiability. The constructions of publicly verifiable time-lock puzzles utilize the VDFs: the puzzle-solving process can be regarded as the execution of a VDF. So, aside from the solution of the puzzle, the puzzle solver can generate an additional proof to prove the correctness of the solution.

2. Preliminaries

In this section, we provide the necessary cryptographic foundations and formal definitions that utilized in our work.

2.1. Time-Lock Puzzle

A time-lock puzzle [1,20,27,29–34] is used to encapsulate a message such that nobody can obtain it before spending computation for a certain amount of time. For conceptual simplicity, we consider schemes with binary messages. Specifically, a time-lock puzzle scheme, Puz, contains the following algorithms.

- $z \leftarrow \mathsf{Encaps}(s,t)$: For a message, $s \in \{0,1\}$, and time parameter, t, the puzzle generator runs this *randomized* encapsulation algorithm to encapsulate s into a time-lock puzzle, z.
- $m \leftarrow \mathsf{Sol}(z)$: The receiver runs this *deterministic* puzzle-solving algorithm to solve the puzzle, z, and obtain the message, s.

Definition 1. *(Correctness) For all security parameter λ, all polynomials t in λ and all messages $s \in \{0,1\}$, it holds that* $\mathsf{Sol}(\mathsf{Encaps}(s,t)) = s$.

Definition 2. *(Security) A time-lock puzzle is sequential with gap $\varepsilon < 1$, if there exists a polynomial $\hat{\mathcal{T}}(\cdot)$. For all polynomial $\mathcal{T}(\cdot) \geq \hat{\mathcal{T}}(\cdot)$ and for all polynomial time adversary $\mathcal{A}$ with running times*

of less than $\mathcal{T}^{\epsilon}(\lambda)$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all security parameters,* λ*, it holds that:*

$$\Pr[b \leftarrow \mathcal{A}(z) : z \leftarrow \mathsf{Encaps}(b,t)] \leq 1/2 + \mathsf{negl}(\lambda). \quad (1)$$

Definition 3. *(Efficiency) For all messages* $s \in \{0,1\}$*, the encapsulation algorithm* $\mathsf{Encaps}(s,t)$ *can be computed in time* $\mathsf{poly}(\log t, \lambda)$*, while the puzzle-solving algorithm,* $\mathsf{Sol}(z)$*, can be computed in time* $t \cdot \mathsf{poly}(\lambda)$*.*

2.2. Signature Scheme

A signature scheme [35] consists of the following three probabilistic polynomial time (PPT) algorithms: $\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify}$:

- $(vk, sk) \leftarrow \mathsf{KGen}(1^{\lambda})$: The *randomized* key generation algorithm takes the security parameter 1^{λ} as input and outputs the verification key and signing key pairs (vk, sk). The message space is denoted as $\mathcal{M}$.
- $\sigma \leftarrow \mathsf{Sign}(sk, m)$: The signing algorithm takes the signing key, sk, and a message $m \in \mathcal{M}$ as input, and outputs the signature σ.
- $0/1 \leftarrow \mathsf{Verify}(vk, m, \sigma)$: This *deterministic* verification algorithm takes the verification key, vk, the message, $m \in \mathcal{M}$, and the signature, σ, as input. If σ is the valid signature on m, then the verification algorithm outputs 1; otherwise, it outputs 0.

Definition 4. *(Correctness) For all* 1^{λ}*, all* (vk, sk) *generated by* $\mathsf{KGen}(1^{\lambda})$ *and all message* $m \in \mathcal{M}$*, it holds that* $\mathsf{Verify}(vk, m, \mathsf{Sign}(sk, m)) = 1$*.*

Definition 5. *(Existentially unforgeable under a chosen-message attack (EUF-CMA)) All PPT adversaries,* $\mathcal{A}$*, can query messages* $\{m_1, m_2, \dots\}$*, obtaining the corresponding signatures* $\{\sigma_1, \sigma_2, \dots\}$*. Denote* $\mathcal{Q}$ *as the set of messages queried by* $\mathcal{A}$*; here, there exists a negligible function,* $\mathsf{negl}(\cdot)$*, such that:*

$$\Pr[1 \leftarrow \mathsf{Verify}(vk, m, \sigma) \wedge m \notin \mathcal{Q} : (m, \sigma) \leftarrow \mathcal{A}(\{m_i, \sigma_i\}_{i=1,2,\dots})] \leq \mathsf{negl}(1^{\lambda}). \quad (2)$$

2.3. Non-Interactive Zero-Knowledge Proofs of Discrete Logarithm Knowledge

Given four group elements $g, \hat{g}, h, \hat{h}$, a zero-knowledge proof $EQLOG(g, \hat{g}, h, \hat{h})$ is used to demonstrate the knowledge of a secret value α, such that $\hat{g} = g^{\alpha}$ and $\hat{h} = h^{\alpha}$ without revealing any secret information about α. Chaum and Pedersen presented an interactive proof construction in [36]; this can be transformed into a non-interactive version as follows:

- The prover selects a random value β, and sends $g_1 = g^{\beta}$ and $h_1 = h^{\beta}$ to the verifier.
- The prover computes $e = \mathcal{H}_e(g, \hat{g}, h, \hat{h}, g_1, h_1)$ by a hash function $\mathcal{H}_e$ and sends $z = \beta - \alpha e$ to the verifier.
- If both $g_1 = g^z \hat{g}^e$ and $h_1 = h^z \hat{h}^e$ hold, then the verifier accepts this proof.

2.4. Non-Interactive Proofs of Sequential Computation

In a sequential computation, the evaluator needs to generate a sequential proof $\Pi(x, y, t)$ to prove that the output value y is exactly evaluated based on the input x by t sequential computations. Boneh et al. [7] proposed a construction of non-interactive sequential proof $\Pi(x, y, t)$ for iterated squaring based on an RSA group $\mathbb{Z}_N^*$ when the time parameter $t = 2^{\tau}$ for a positive integer τ. That is, $N = pq$ for two distinct primes, p, q and $y = x^{2^{2^{\tau}}} \mod N$.

Denote $b_i = g^{2^{2^i}} \mod N$; the prover generates a vector $W = \langle b_0, b_1, \dots, b_{\tau} \rangle$. For each $i = 0, \dots, \tau - 1$, the prover shows that the tuple (g, b_i, b_{i+1}) satisfy the relationship (g, g^a, g^{a^2}) for some a by a NIZK proof $EQLOG(g, g^a, g^a, g^{a^2})$. By verifying that the last element in w is $b_{\tau} = y$, the verifier is assured that y is exactly $x^{2^{2^{\tau}}} \mod N$. These τ proofs can be generated by the prover in parallel, and the length of the total proof $\Pi(x, y, t)$ is $O(\log t)$ for the time parameter, t.

2.5. Sequential Squaring Assumption

The sequentiality of iterated squaring on an RSA group is based on the following assumption:

Definition 6. *(Sequential Squaring Assumption [32]) Let N be a uniform strong RSA integer, g be a generator of $\mathbb{J}_N$, and $\mathcal{T}(\cdot)$ be a polynomial. Then, there exists some $0 < \varepsilon < 1$, such that, for every polynomial-size adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ whose depth is bounded from above by $\mathcal{T}^\varepsilon(\lambda)$, there exists a negligible function* $\mathsf{negl}(\cdot)$*:*

$$\Pr\left[b \leftarrow \mathcal{A}(N, g, \mathcal{T}(\lambda), x, y) : \begin{array}{c} x \xleftarrow{\$} \mathbb{J}_N;\ b \xleftarrow{\$} \{0,1\} \\ \text{if } b = 0 \text{ then } y \xleftarrow{\$} \mathbb{J}_N \\ \text{if } b = 1 \text{ then } y = x^{2^{\mathcal{T}(\lambda)}} \end{array}\right] \leq 1/2 + \mathsf{negl}(\lambda). \tag{3}$$

Note that the restriction of x and y to $\mathbb{J}_N$ is to avoid trivial attack where the adversary computes the Jacobi symbol of the group elements.

3. The Framework

As introduced in Section 1, when a time-lock puzzle scheme is implemented in practice, it is imperative to perform verification both before and after the execution of the puzzle solution. Before solving the puzzle, the pre-verification prevents the solver from spending time-consuming operations on invalid time-lock puzzles. After solving the puzzle, the post-verification can convince a third party of the correctness of the solution.

In this section, we present an overview and the syntax of our framework, which is designed to perform verification both before and after solving a time-lock puzzle. Additionally, we discuss the adversary model and the properties that the framework should satisfy.

3.1. Overview of the Framework

The framework is given in Figure 1. A time-lock puzzle is executed between two participants: the sender $\mathcal{S}$ and the receiver $\mathcal{R}$. The sender is the generator of the puzzle, it encapsulates the secret message into the time-lock puzzle. The receiver is also the solver of the puzzle, it first checks the validity of the puzzle and solves the valid puzzle by spending computation for a certain amount of time. As shown in Figure 1, the encapsulation for the message is denoted as a time-related relationship, $F_t(\cdot)$. Here, t is the time parameter which determines that the receiver must spend *at least* time t to solve the puzzle. $F(\cdot)$ is a commitment scheme and the signing algorithm is denoted as $\mathsf{Sign}_{sk}(\cdot)$, where sk is the signing key of the sender $\mathcal{S}$.

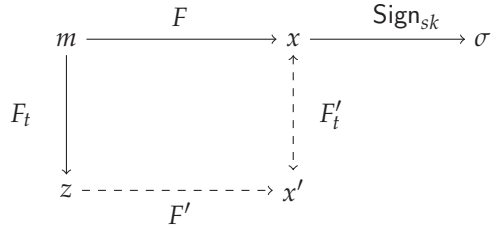


Figure 1. The framework.

For a secret message m, the time-lock puzzle is generated as $z = F_t(m)$. The sender evaluates the commitment scheme F on the message to obtain the value $x = F(m)$, and then signs on x by its signing key, sk, to produce the signature $\sigma = \mathsf{Sign}_{sk}(x)$. After receiving the tuple (z, x, σ) from the sender, the receiver verifies the validity of the tuple.

Since the sender does not sign on the puzzle, z, directly, the receiver cannot verify the validity of the puzzle by verifying the signature. This signature can ensure the validity of the commitment value, x. The dotted line in Figure 1 shows the design idea of our framework: $F'(\cdot)$ is a function related to $F(\cdot)$ that maps the time-lock puzzle, z, to an

intermediate value, x'. If the puzzle, z, is correctly generated, then the two values x and x' should satisfy a corresponding relationship, $F'_t(\cdot)$, which can be verified by the receiver. If the puzzle passes the verification, then the receiver is convinced that the time-lock puzzle, z, is correctly generated.

After a solution, m', is revealed by the receiver from the puzzle, z, a third party can verify the correctness of this solution, m', by verifying the relationship between the message, m', and the commitment value, x. The validity of x is guaranteed by the signature, σ, and a third party verifies whether x is the commitment to the message m'.

3.2. The Syntax

This section presents the syntax of the framework. The main idea of the framework is that the receiver generates an intermediate value x' based on the puzzle, z, and verifies the relationship between x and x'. For a smooth reading experience, the table of variables is presented in Table A1. Formally, the framework contains the following algorithms:

- $(pk, sk) \leftarrow \mathsf{Setup}(1^\lambda)$: The *randomized* setup algorithm takes the security parameter 1^λ as input, and outputs the public verification key, pk, and the private signing key, sk, for the sender. By convenience, the security parameter specifies the message space, $\mathcal{M}$.
- $x \leftarrow \mathsf{Commit}(m)$: The commitment algorithm takes the message $m \in \mathcal{M}$ in the message space as input and outputs a commitment value, x.
- $\sigma \leftarrow \mathsf{Sign}(sk, x)$: The signing algorithm takes the signing key of the sender, sk, and the commitment value, x, as inputs, and outputs the signature, σ.
- $z \leftarrow \mathsf{Encaps}(m, t)$: The *randomized* encapsulation algorithm is used to encapsulate the message into a time-lock puzzle. It takes the message, m, and the time parameter, t, as inputs, and outputs the time-lock puzzle, z. The running time of the encapsulation algorithm should be much less than t.
- $\pi \leftarrow \mathsf{ProofGen}(m, x, z, \sigma)$: The proof generation algorithm is used to output a non-interactive proof, π, which can help the receiver *efficiently* verify the validity of the puzzle, z. It takes the secret message, m, the commitment value, x, the puzzle, z, and the signature, σ, as inputs, and outputs a proof, π. The output π can be represented as an empty string NULL if no such proof is required.
- $0/1 \leftarrow \mathsf{PuzVerify}(pk, z, x, \sigma, \pi)$: The *deterministic* puzzle verification algorithm takes the puzzle, z, the commitment value, x, the signature, σ, and proof π as inputs. It outputs 1 if z is a valid time-lock puzzle and outputs 0 otherwise. The running time of this puzzle verification algorithm should be much less than t.
- $m' \leftarrow \mathsf{Solve}(z)$: If the puzzle verification algorithm outputs 1, then the receiver runs this puzzle-solving algorithm to reveal the message from the puzzle. The puzzle-solving algorithm takes the time-lock puzzle, z, as input and outputs a solution m'. It requires *at least* time t for the receiver to run this puzzle-solving algorithm.
- $0/1 \leftarrow \mathsf{SolVerify}(pk, m', z, x, \sigma, \pi)$: The *deterministic* solution verification takes the public verification key, pk, the solution, m', the commitment value, x', the puzzle, z, the signature, σ, and the proof, π, as inputs. It outputs 1 if the solution, m', is exactly the message encapsulated in the puzzle, z, and outputs 0 otherwise. The running time of this solution verification algorithm should be much less than t.

3.3. Property Requirements

The goal of an adversary is to deviate the protocol from its security requirements. For example, after obtaining a valid tuple of outputs, a passive adversary may want to solve the puzzle faster than the required time, and an active adversary may try to forge a puzzle. The active adversary tries to generate a forged puzzle, such that it can pass the verification of the verifier but cannot be revealed as the correct message. In this paper, we assume that all adversaries are probabilistic polynomial time, which means that the adversary cannot break the underlying cryptographic primitive.

Definition 7. *(Correctness) For all security parameters* 1^λ*, all key pairs* (pk, sk) *generated by* $\mathsf{Setup}(1^\lambda)$*, all messages,* m*, and all time parameters,* t*, if the signature is* $\sigma \leftarrow \mathsf{Sign}(sk, \mathsf{Commit}(m))$*, the puzzle is* $z \leftarrow \mathsf{Encaps}(m, t)$*, and the proof is* $\pi \leftarrow \mathsf{ProofGen}(m, \mathsf{Commit}(m), z, \sigma)$*, then it holds that* $\mathsf{PuzVerify}(z, \mathsf{Commit}(m), \sigma, \pi) = 1$*,* $m \leftarrow \mathsf{Solve}(z)$ *and* $1 \leftarrow \mathsf{SolVerify}(pk, m, z, \sigma)$*.*

Definition 8. *(Sequentiality) For all security parameters,* 1^λ*, all time parameters,* t*, all messages,* m*, and all tuples,* (z, x, σ, π)*, generated by the sender, if the puzzle verification algorithm* $\mathsf{PuzVerify}(pk, z, x, \sigma, \pi)$ *outputs* 1*, then, for* $0 < \varepsilon < 1$ *and the running time of an adversary* $\mathcal{A}$ *which is less than* $\varepsilon \cdot t$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that*

$$\Pr\left[m' = m : \begin{array}{c} z \leftarrow \mathsf{Encaps}(m, t) \\ m' \leftarrow \mathcal{A}(z, x, \sigma, \pi) \end{array}\right] \leq \mathsf{negl}(1^\lambda). \tag{4}$$

Definition 9. *(Unforgeability of puzzle) For any probabilistic polynomial time adversary* $\mathcal{A}$*, it can query the set of messages* $\{m_i\}_{i=1,2,\ldots}$ *and obtains the corresponding set of tuples* $\mathcal{T} = \{(z_i, x_i, \sigma_i, \pi_i)\}_{i=1,2,\ldots}$*. Denote the set of all required puzzles as* $\mathcal{Z}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that:*

$$\Pr\left[\begin{array}{c} 1 \leftarrow \mathsf{PuzVerify}(z', x', \sigma', \pi') \\ z' \notin \mathcal{Z} \end{array} : (z', x', \sigma', \pi') \leftarrow \mathcal{A}(\mathcal{T})\right] \leq \mathsf{negl}(1^\lambda). \tag{5}$$

In Definition 9, an active adversary wants to forge a tuple (z', x', σ', π') that can pass the puzzle verification algorithm but cannot be correctly solved by the receiver. The puzzle z' in the faked tuple should be different from the correct one, z, otherwise, the receiver can still retrieve the message encapsulated in the puzzle. So, it requires that the puzzle z' has not been queried before.

Definition 10. *(Unforgeability of solution) For all security parameters,* 1^λ*, all time parameters,* t*, and all messages,* m*, if the tuples* (z, x, σ, π) *are generated correctly, then, for all probabilistic polynomial time adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that:*

$$\Pr\left[\begin{array}{c} m' \neq m \\ 1 \leftarrow \mathsf{SolVerify}(pk, m', z, x, \sigma, \pi) \end{array} : \begin{array}{c} z \leftarrow \mathsf{Encaps}(m, t) \\ m' \leftarrow \mathcal{A}(z, x, \sigma, \pi) \end{array}\right] \leq \mathsf{negl}(1^\lambda). \tag{6}$$

Definition 11. *(Verifiability of puzzle) After receiving the tuple* (z, x, σ, π) *from the sender, the receiver should be able to verify the validity of the time-lock puzzle,* z*, by the puzzle verification algorithm,* $\mathsf{PuzVerify}(z, x, \sigma, \pi)$*. The running time of the puzzle verification algorithm should be much less than time* t*.*

Definition 12. *(Public verifiability of solution) After the receiver reveals its solution,* m'*, on the time-lock puzzle,* z*, everyone can verify the correctness of the solution by the solution verification algorithm* $\mathsf{SolVerify}(pk, m', z, \sigma)$*. The running time of the solution verification algorithm should be much less than time* t*.*

Note that the verification for the puzzle is not required to be performed publicly, since only the receiver needs to verify the validity of the puzzle before spending a large amount of sequential computation. The verifiability of the solution should be public so that everyone can verify the correctness of the result.

4. Iterated-Squaring-Based Construction

According to the framework presented in Section 3, we give a specific construction based on iterated squaring on an RSA group and discuss its properties in this section.

4.1. The Construction

The sender randomly chooses two distinct safe primes, $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes, and then it computes $N = pq$ and chooses a random generator, $g \in \mathbb{Z}_N^*$. (A zero-knowledge proof π_N can be generated to prove that N is the product of two safe primes [37]). The time-lock puzzle is constructed on $\mathbb{Z}_N^*$: for a large time parameter, t, the sender computes $h = g^{2^t} \bmod N$ *efficiently* by the trapdoor $\varphi(N)$; it first computes $e = 2^t \bmod \varphi(N)$ and computes $h = g^e \bmod N$, which is used as a mask to encapsulate the message, m. The time-lock puzzle is generated as $z = m \cdot h \bmod N$. After receiving the puzzle, z, the receiver computes h based on the element, g, and the time parameter, t. Since the receiver does not know the trapdoor, $\varphi(N)$, it must compute h by t sequential computations. The receiver can reveal the message, m, from the puzzle, z, as $m = z \cdot h^{-1} \bmod N$.

Based on the above time-lock puzzle construction, we combine it with the framework in Section 3 to verify the secret message. To facilitate the generation of the proof, we require that the time parameter, t, can be presented as $t = 2^\tau$ for a positive integer τ. For a secret message, $m \in \mathbb{Z}_N^*$, the time-lock puzzle, z, is generated as the protocol in the above section. The sender chooses a random value $a \in \mathbb{Z}_{\varphi(N)}^*$ and computes $x = m^a \bmod N$. It signs on x to obtain the signature σ. Additionally, the sender generates a sequential proof $\pi_t = \Pi(g^a, (g^a)^{2^t}, t)$ to help the receiver check the validity of the puzzle efficiently. The receiver is given the values x, a, the proof, π_t, and the signature, σ.

As a summary, the process of the sender is present in Algorithm 1. For simplicity, we denote all values except the time-lock puzzle sent by the sender as auxiliary information.

Algorithm 1: The sender's process based on iterated squaring

Setup: The RSA number, N, the group, $\mathbb{Z}_N^*$, the public verification key, pk, and the secret signing key, sk;
Input: A secret message $m \in \mathbb{Z}_N^*$;
Choose a random element $g \in \mathbb{Z}_N^*$;
Generate the puzzle as $z = m \cdot g^{2^t} \bmod N$;
// The sender can compute $g^{2^t} \bmod N$ efficiently as $g^{2^t \bmod \varphi(N)} \bmod N$
Choose a random value $a \in \mathbb{Z}_{\varphi(N)}^*$;
Compute the commitment of the message as $x = m^a \bmod N$;
Generate the sequential proof $\pi_t = \Pi(g^a, (g^a)^{2^t} \bmod N, t)$;
Sign on x: $\sigma = \mathsf{Sign}(sk, x)$;
Output: The time-lock puzzle, z, and auxiliary information g, a, x, σ, π_t.

After receiving the outputs from the sender, the receiver first checks the validity of the signature, σ, to make sure that x is correctly generated by the sender, then the receiver needs to verify the relationship between z and x. The receiver computes $x' = z^a \bmod N$ and computes $x'' = x' \cdot x^{-1} \bmod N$. Then, it verifies the time relationship between g^a and x'' by the proof π_t. If the value x'' passes the verification, then the receiver is convinced that it has obtained a value $x'' = (g^{2^t})^a \bmod N$. Since the operation to obtain the value x'' is the exponent on the puzzle, z, the receiver is convinced that the puzzle is $m \cdot g^{2^t}$, and the revealed value m is already committed by the sender.

The algorithm of the receiver is shown in Algorithm 2:

Algorithm 2: The receiver's process based on iterated squaring

Input: The time-lock puzzle, z, and auxiliary information g, a, x, σ, π_t;
`// Check the validity of the signature` σ
if Verify$(pk, x, \sigma) = 0$ **then**
  Output $\perp$
end
Compute $x' = z^a \bmod N$ and $x'' = x' \cdot x^{-1} \bmod N$;
if $x'' \neq (g^a)^{2^t} \bmod N$ **then**
  Output $\perp$
end
`// This verification can be performed by the sequential proof` π_t
Compute $h' = g^{2^t} \bmod N$;
`// The receiver needs to compute` h' `by iterated squaring`
Compute $m' = z \cdot h'^{-1} \bmod N$;
Output: A message $m' \in \mathbb{Z}_N^*$.

The security of the RSA group is based on the assumption that the adversary cannot find the factorization of the RSA number efficiently. Another group used in iterated squaring is the class group of an imaginary quadratic field [38,39]. This group has a property that there is no efficient algorithm to compute the order of such group.

An alternative method for constructing a time-lock puzzle is based on modular square roots. For a prime, p, and a quadratic residue a modulo p, a sequential time $O(\log p)$ is required to compute the square roots of a. In this method, the time of sequential evaluation is fixed for a given prime, p. In contract, with an iterated-squaring-based construction, the time can be easily adjusted by changing the number of iterations.

4.2. Security Analysis

As outlined in Section 3, the construction should satisfy the following properties.

Correctness

The correctness requires that if the outputs of the sender are correctly generated, then the time-lock puzzle must pass the verification of the receiver, and the puzzle can be revealed as the same message encapsulated by the sender.

Theorem 1. *For a secret message m, if the sender follows the protocol and sends the time-lock puzzle, z, and auxiliary information g, a, x, σ, π_t to the receiver, then the puzzle, z, can pass the puzzle verification algorithm and be revealed as the secret message, m, by the receiver.*

Proof. If the sender and the receiver both follow the protocol, the sender uses $h = g^{2^t \bmod \varphi(N)} \bmod N$ to encapsulate the secret message, h, and the receiver computes $h' = g^{2^t} \bmod N$ by iterated squaring. Since the order of $\mathbb{Z}_N^*$ is $\varphi(N)$, there is $h' = h$. Hence, the time-lock puzzle can be correctly solved by the receiver.

Additionally, if the time-lock puzzle and auxiliary information are generated correctly, it holds that $x = m^a \bmod N$, $z = m \cdot g^{2^t} \bmod N$, and $\pi_t = \Pi(g^a, (g^a)^{2^t}, t)$. The receiver computes $x'' = z^a \cdot x^{-1} \bmod N$ and can use the sequential proof $\pi_t = \Pi(g^a, x'', t)$ to verify that x'' is exactly $(g^{2^t})^a \bmod N$. So, the receiver can be convinced that the puzzle is correctly generated as $z = m \cdot g^{2^t} \bmod N$. After verifying the validity of the puzzle, z, the receiver can solve the puzzle as $m = z \cdot (g^{2^t})^{-1} \bmod N$. □

Sequentiality

The sequentiality requires all probabilistic polynomial time adversaries to spend a certain amount of time to solve the puzzle.

Theorem 2. *In the construction in Section 4, for* $0 < \varepsilon < 1$ *and any adversary* $\mathcal{A}$ *with a running time less than* $\varepsilon \cdot t$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that the probability that the receiver can solve the puzzle faster than time* $\varepsilon \cdot t$ *after receiving the puzzle,* z*, from the sender is* $\mathsf{negl}(\lambda)$*.*

Proof. The sequentiality of the time-lock puzzle is based on Definition 6. For any probabilistic polynomial time adversary $\mathcal{A}$, it does not know the factorization of N; hence, $\mathcal{A}$ cannot compute $h' = g^{2^t} \bmod N$ faster than time t.

Except for the time-lock puzzle, z, $\mathcal{A}$ is also given x, which is also related to the secret message, m. The secret message, m, can be computed from x as $m = x^{1/a} \bmod N$. Since $\mathcal{A}$ does not know the factorization of N, the probability of success for $\mathcal{A}$ is $\mathsf{negl}(\lambda)$.

Additionally, the proof π_t is related to the sequential computation and the adversary wants to compute $h' = g^{2^t} \bmod N$ faster from π_t. The specific construction of π_t is given in Section 2, the values given in π_t can be presented as $b_i = g^{2^{2^i} \cdot a} \bmod N$ for $i = 1, 2, \ldots, \tau$, where $t = 2^\tau$. The closest value to h' is $b_{\tau-1} = g^{2^{t/2} \cdot a} \bmod N$. Since $a \in \mathbb{Z}^*_{\varphi(N)}$, there is $a < 2^\lambda$. Hence, it requires at least $t/2 - \lambda$ times iterated squaring for the adversary to compute h' from $b_{\tau-1}$. In practice, t is chosen as 2^{30} and $\lambda = 2^{10}$, so $\lambda \ll t/2$.

Hence, after receiving the puzzle, z, from the sender, the receiver should spend at least time $\varepsilon \cdot t$ to solve the puzzle, where $\varepsilon < 1/2$. □

Unforgeability

The unforgeability of the puzzle requires that, for any probabilistic polynomial time adversary, the probability that it can generate a forged puzzle that can pass the puzzle verification algorithm is negligible.

Theorem 3. *In the construction in Section 4, for any probabilistic polynomial time adversary* $\mathcal{A}$*, it can query the message it chooses and obtain the corresponding outputs of the sender. There exists a negligible function,* $\mathsf{negl}(\cdot)$*, and the probability that the adversary can forge a new puzzle that can pass the verification of the receiver is* $\mathsf{negl}(\lambda)$*.*

Proof. The adversary $\mathcal{A}$ can query the set of messages $\{m_i\}_{i=1,2,\ldots}$ and obtains the corresponding set of tuples $\mathcal{T} = \{(z_i, x_i, \sigma_i, \pi_i)\}_{i=1,2,\ldots}$. Denote the set of all obtained puzzles as $\mathcal{Z}$. If $\mathcal{A}$ can forge a new puzzle $z' \notin \mathcal{Z}$ and passes the pre-verification, then it must satisfy one of the following two situations.

The first situation is that the adversary generates a tuple by choosing a new message m', and generate the puzzle z', the commitment value x' by itself. It then tries to forge the signature σ', on the commitment value x'. The unforgeability of the signature scheme ensures that the probability of $\mathcal{A}$ can forge a signature on a new commitment value x' successfully is $\mathsf{negl}(\lambda)$.

Another situation is to bypass the forgery of the signature scheme. The adversary $\mathcal{A}$ reuses the commitment value x_i of some tuple that has been queried before. In this case, if the adversary wants to generate another puzzle $z'_i \neq z_i$, then it tries to find another message $m'_i \neq m_i$ but $F(m'_i) = F(m_i)$. The construction in Section 4 sets $F(\cdot)$ as $F(x) = x^a \bmod N$ for a random $a \in \mathbb{Z}^*_{\varphi(N)}$, since a is coprime with $\varphi(N)$, the function F is a permutation on $\mathbb{Z}^*_N$. Hence, the adversary cannot find such m'_i.

In summary, the probability that a PPT adversary can forge a valid puzzle that can pass the pre-verification is negligible. □

The unforgeability of solution requires that the probability that a receiver can forge a solution that can pass the solution verification is negligible.

Theorem 4. *In the construction in Section 4, for any secret message,* m*, after receiving the corresponding valid tuple* z, x, σ, π *from the sender, there exists a negligible function,* $\mathsf{negl}(\cdot)$*, and the probability that the adversary can forge a solution,* $m' \neq m$*, that can pass the solution verification is* $\mathsf{negl}(\lambda)$*.*

Proof. After obtaining the solution, m', from the receiver, the third party first checks whether σ is the signature of the value, x, by the public key of the puzzle generator, and then verifies whether x is the commitment on m'. If the receiver can succeed forge a solution $m' \neq m$ that can pass the verification of the third party, it must satisfy one of the following two situations.

The first situation is that the receiver generates a commitment x' on the forged solution, m', and then forges the signature σ', on x'. The unforgeability of the signature scheme guarantees that the probability that this situation happens is negligible. Another situation requires the commitment x can be revealed as the forged solution, m'. The probability of this situation is negligible according to the binding property of the commitment scheme.

In summary, the probability that a receiver can forge a solution, $m' \neq m$, and pass the post-verification step, is negligible. □

4.3. Efficiency Analysis

According to Algorithm 1, in order to generate the time-lock puzzle, the sender first computes $e = 2^t \bmod \varphi(N)$ and then computes $h = g^e \bmod N$. These are two modular exponentiation operations, which require $O(\log t) + O(\lambda)$ multiplications over $\mathbb{Z}_N^*$. In the puzzle verification algorithm, the receiver needs to compute $x' = z^a \bmod N$, $x'' = x' \cdot x^{-1} \bmod N$ and verify that x'' is exactly $(g^{2^t})^a \bmod N$ by the proof π_t. The proof π_t contains $\log t$ small proofs which can be verified in parallel. The verification time complexity of each small proof is $O(\lambda)$. So the time complexity of this verification process is $O(\lambda)$. For the solution verification algorithm, a third party checks the correctness of the solution by verifying the signature and the commitment, which results in a time complexity of $O(\lambda)$. As proved in Theorem 2, the time complexity of solving the time-lock puzzle for all receiver is $O(t)$.

In practice, the time parameter t is a polynomial of the security parameter λ. This indicates that the generation and the verification of the time-lock puzzle are much faster than solving the puzzle. Hence, the pre-verification and the post-verification are both efficient.

4.4. Construction for Long Messages

In the above construction, it requires that the message, m, lies in the group $\mathbb{Z}_N^*$. In practice, N is chosen as a 2048 bits number; hence, the length of m is at most 2048 bits, which is shorter than the length of message in a real network. A long message can still be treated as a group element in the protocol, as in the case of a short message. In this method, a group with the same length as the message needs to be generated during the setup phase of the protocol. This will increase the evaluation time of each group operation, resulting in the poor efficiency of the protocol. Additionally, this method is not flexible if the length of the message changes. A new group must be generated for a longer message, which is inconvenient in practice implementation.

A common approach to handle a long message is to segment the message into short consecutive blocks: $m = m_1||m_2||\ldots||m_n$. Here, $||$ represents the concatenation of two strings, and each block m_i is κ bits. Here, κ is a parameter related to the security parameter λ. For example, $\kappa = \lambda - 1$. This ensures that each block m_i can be represented as a group element in $\mathbb{Z}_N^*$ with overwhelming probability. If the last block m_n is less than κ bits, a padding scheme can be used to fill the last block.

The element $h = g^{2^t} \bmod N$ we used to encapsulate a short message is a group element in $\mathbb{Z}_N^*$, which is at most λ bits. It can be used to encapsulate only one block each time. A naive idea is to compute different $h_i = g_i^{2^t} \bmod N$ to mask different blocks, i.e., the puzzle is generated as $z = h_1 \cdot m_1||h_2 \cdot m_2||...||h_n \cdot m_n$. In this case, the verification for each block is the same as the short message case, and the receiver needs to solve n different short time-lock puzzles to reveal the message. Another method to generate the puzzle uses only one time-delay element. The sender computes $h = g^{2^t} \bmod N$ and generates the puzzle as $z = h \cdot m_1||h^2 \cdot m_2||...||h^{2^{n-1}} \cdot m_n$. In this case, the receiver needs to solve one short time-lock puzzle to reveal the message.

5. Experiment and Analysis

This section presents the efficiency of our construction. When implemented in practice, it is necessary to require that the verification time of the receiver to check the validity of the time-lock puzzle is much less than solving the time-lock puzzle. Additionally, a third party should be able to verify the correctness of the solution efficiently. We analyze the efficiency of the protocol from both theoretical and practical aspects. Moreover, we implement the construction in Python and emulate the running time in different settings.

5.1. Experiment Setup

We implement our construction in Python and use cryptography libraries for fundamental cryptographic and arithmetic operations. Since we do not care about the specific communication method between the sender and the receiver, we implement the sender and the receiver in one laptop.

The experimental environment is presented in the following table (Table 1).

Table 1. The experiment setup.

Component	Description
CPU	Intel Core i5-10210U (1.60 GHz)
Number of threads	1
RAM	16 GB
System	Ubuntu 22.04 LTS

5.2. Comparison of Solving Time and Verification Time

In the following experiment, the security parameter is 2048, i.e., N is a 2048-bit number. First, we investigate how the running time of solving a time-lock puzzle relates to the time parameter, t, by conducting experiments with t ranging from 2^{16} to 2^{21}, while maintaining the other conditions the same. The results are depicted in Figure 2 using logarithmic coordinates, indicating linear correlations between the average running time to solve a time-lock puzzle and the time parameter, t. As a comparison, the verification time for the receiver to perform the pre-verification and a third party to perform the post-verification are given in Table 2.

To minimize single measurement errors, each of the above values is the average of ten measurements taken with the same settings. The experimental data indicate that the running time for the sender to generate the time-lock puzzle and the verification time for the receiver to check the validity of the time-lock puzzle are on the order of milliseconds and increase slowly with the time parameter. For example, when the time parameter is 2^{21}, the time it takes for the receiver to verify the validity of the puzzle is about 1/17 of the time it takes to solve the puzzle. Hence, it implicates the efficiency of our construction.

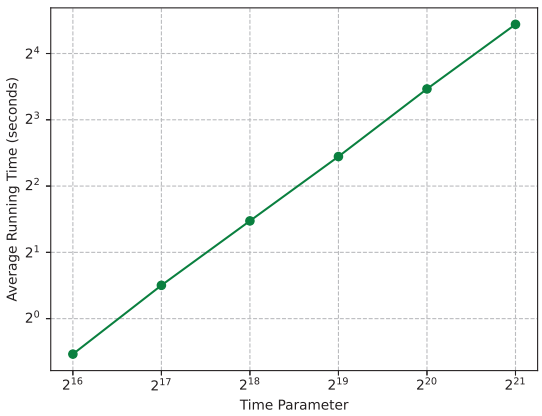


Figure 2. Average solving time under different time parameters.

Table 2. Average running time under different time parameters.

Time Para.	Puzzle Gen. (ms)	Puzzle Verif. (Sequential) (ms)	Puzzle Verif. (Parallel) (ms)	Solution Verif. (ms)
2^{16}	26.6	975.2	61.0	27.6
2^{17}	27.3	1037.9	61.0	28.3
2^{18}	27.9	1090.3	60.6	28.2
2^{19}	29.3	1141.8	60.1	27.0
2^{20}	34.8	1236.7	61.8	27.8
2^{21}	40.0	1257.9	59.9	27.8

We compare our experimental results with those in [18] in Table 3. The verification time of their construction is 38 ms and it takes 21.725 s for the receiver to solve the puzzle. A third party takes 8.673 ms to verify the solution. Their implementation also shows that the verification time is in the order of milliseconds and is much smaller than the time to solve the puzzle.

Table 3. Comparison with related work.

	Puzzle Gen. (ms)	Puzzle Verif. (ms)	Solution Verif. (ms)	Puzzle Sol. (s)
[18]	37.98	38	8.673	21.725
Our work	40	59.9	27.8	21.7

5.3. Communication Cost

In our implementation, the sender sends the time-lock puzzle, the signature, the commitment, and the proof to the receiver. When the security parameter is 2048, both the puzzle and the commitment are 256 bytes in length. The length of the signature depends on the specific signature scheme. We use ECDSA with secp256r1 as the underlying signature scheme; the length of the signature is 64 bytes. The length of the proof is related to the time parameter, when the time parameter is 2^{20}, the proof length is 15 KB.

5.4. Discussion about Solving Time

The difference in computational power between different solvers always exists when a time-lock puzzle is implemented in practice [40]. It is an inherent issue since adversaries can always gain a time advantage if they possess more sequential computational power than honest nodes. This difference in computation power brings different solving times for the same time-lock puzzle. According to the performance data given by Intel [41], the difference between updated server-grade and updated custom-grade chips is approximately 10. Hardware performance also doubles roughly every four years, as indicated by the performance growth rate given in [42].

Generating a time-lock puzzle is much more efficient than solving the puzzle, so the differences in hardware do not significantly affect puzzle generation. However, if adversaries were to use updated server-grade chips while honest nodes employed four-year-old custom-grade chips, then the adversaries would be able to solve the time-lock puzzle 20 times faster than the honest nodes. Consequently, when a time-lock puzzle protocol is used in practice, honest nodes should periodically upgrade their hardware to make sure that the disparity of computation power remains within a reasonable range. This represents a trade-off between the security and the hardware requirements of honest nodes.

6. Conclusions and Discussion

This paper considers the verifiability of a time-lock puzzle scheme, which is an important property of a time-lock puzzle scheme when implemented in practice. Before the receiver starts solving the time-lock puzzle, it should be able to verify the validity of the

puzzle to avoid invalid computation. After the puzzle is solved, a third party should verify the correctness of the solution without resolving the puzzle.

In this paper, we propose a framework of time-lock puzzles to provide both pre-verification and post-verification by combining a signature scheme. In this framework, the receiver of the time-lock puzzle can verify the validity of the puzzle before it starts solving the puzzle. After receiving the puzzle from the puzzle generator, the receiver computes an intermediate value based on the values it receives and checks the validity of the puzzle. It can be convinced that the secret value behind the time-lock puzzle has already been signed by the puzzle generator if the puzzle passes the verification. After the time-lock puzzle is solved, a third party can also verify the correctness of the solution by the signature efficiently.

Based on the framework, we present a specific construction based on iterated squaring over the RSA group and give security proofs for these constructions. We have also implemented the construction in Python. The experimental results show the efficiency of the construction. The running time for the generation of the time-lock puzzle, the verification of the validityof this approach, and the solution of the time-lock puzzle are all in the order of milliseconds, while the running time for the solver is in the order of seconds when the construction is implemented in practice. Consequently, the pre-verification and the post-verification for the time-lock puzzles are both efficient.

Our construction becomes insecure when considering quantum attack, since it is based on iterated squaring on an RSA group. There exists an efficient algorithm to find the factorization of an RSA number on quantum computers. There are four methods used to design post-quantum protocols: lattice-based, code-based, hash-based, and multivariate polynomial. In CRYPTO 2024, Agrawalr et al. [34] present a time-lock puzzle construction from lattice. Hence, it remains a future work to extend our construction with their construction or design a verifiable time-lock puzzle using the other three methods.

In this paper, we only consider the implementation when the time-lock puzzle is used on a small scale. When the time-lock puzzle is used in electronic voting protocol with a large number of voters, it requires a lot of calculations to open each encapsulated vote individually, which effects the efficiency of the protocol. There are two methods to improve the scalability of the time-lock puzzle when it is implemented. One method is using the same time-delay element to generate several time-lock puzzles. In this method, the receiver only needs to solve one time-lock puzzle to obtain the time-delay element and then use this value to solve other time-lock puzzles. Another method is using a homomorphic technique. The receiver only needs to solve one homomorphic time-lock puzzle to obtain the evaluation results of the solution to these puzzles. Moreover, it is a challenge to design scalable quantum-resistance verifiable time-lock puzzles when we both consider the scalability problem and quantum attack.

Author Contributions: Software, H.L.; Validation, Z.W.; Formal analysis, Z.W.; Data curation, H.L.; Writing—original draft, Z.W.; Writing—review and editing, L.W.; Supervision, L.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Key Research and Development (2019YFB2101600).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A. Table of Variables

Table A1. The table of variables.

Variable	Meaning
λ	security parameter
(pk, sk)	public/private key pairs
m	message to be encapsulated
x	commitment to the message
σ	signature on the commitment
t	time parameter
z	time-lock puzzle on the message
π	proof used in pre-verification
x'	intermediate value computed by the receiver
m'	solved message

References

1. Rivest, R.L.; Shamir, A.; Wagner, D.A. *Time-Lock Puzzles and Timed-Release Crypto*; Massachusetts Institute of Technology: Cambridge, MA, USA, 1996.
2. May, T.C. Timed-Release Crypto. 1993. Available online: https://cypherpunks.venona.com/date/1993/02/msg00129.html (accessed on 28 June 2023).
3. Elyakime, B.; Laffont, J.J.; Loisel, P.; Vuong, Q. First-price sealed-bid auctions with secret reservation prices. *Ann. D'economie Stat.* **1994**, *34*, 115–141. [CrossRef]
4. Athey, S.; Levin, J.; Seira, E. Comparing open and sealed bid auctions: Evidence from timber auctions. *Q. J. Econ.* **2011**, *126*, 207–257. [CrossRef]
5. Bag, S.; Hao, F.; Shahandashti, S.F.; Ray, I.G. SEAL: Sealed-bid auction without auctioneers. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 2042–2052. [CrossRef]
6. Chvojka, P.; Jager, T.; Slamanig, D.; Striecks, C. Versatile and Sustainable Timed-Release Encryption and Sequential Time-Lock Puzzles (Extended Abstract). In *Proceedings of the ESORICS 2021, Part II*; Bertino, E., Shulman, H., Waidner, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12973, pp. 64–85. [CrossRef]
7. Boneh, D.; Naor, M. Timed Commitments. In *Proceedings of the CRYPTO 2000*; Bellare, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1880, pp. 236–254. [CrossRef]
8. Wang, G. An abuse-free fair contract signing protocol based on the RSA signature. In Proceedings of the 14th International Conference on World Wide Web, Chiba, Japan, 10–14 May 2005; pp. 412–421.
9. Ben-Or, M.; Goldreich, O.; Micali, S.; Rivest, R.L. A fair protocol for signing contracts. *IEEE Trans. Inf. Theory* **1990**, *36*, 40–46. [CrossRef]
10. Kohno, T.; Stubblefield, A.; Rubin, A.D.; Wallach, D.S. Analysis of an electronic voting system. In Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 9–12 May 2004; pp. 27–40.
11. Gritzalis, D.A. *Secure Electronic Voting*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; Volume 7.
12. Dwork, C.; Naor, M. Zaps and Their Applications. In Proceedings of the 41st FOCS, Redondo Beach, CA, USA, 12–14 November 2000; IEEE Computer Society Press: Piscataway, NJ, USA, 2000; pp. 283–293. [CrossRef]
13. Lin, H.; Pass, R.; Soni, P. Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles. In Proceedings of the 58th FOCS, Berkeley, CA, USA, 15–17 October 2017; Umans, C., Ed.; IEEE Computer Society Press: Piscataway, NJ, USA, 2017; pp. 576–587. [CrossRef]
14. Tyagi, N.; Arun, A.; Freitag, C.; Wahby, R.; Bonneau, J.; Mazières, D. Riggs: Decentralized Sealed-Bid Auctions. In Proceedings of the CCS, Copenhagen, Denmark, 26–30 November 2023; ACM: New York, NY, USA, 2023; pp. 1227–1241.
15. Zhang, M.; Yang, M.; Shen, G. SSBAS-FA: A secure sealed-bid e-auction scheme with fair arbitration based on time-released blockchain. *J. Syst. Archit.* **2022**, *129*, 102619. [CrossRef]
16. Garay, J.A.; Jakobsson, M. Timed Release of Standard Digital Signatures. In *Proceedings of the FC 2002*; Blaze, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2357, pp. 168–182.
17. Garay, J.A.; Pomerance, C. Timed Fair Exchange of Standard Signatures: [Extended Abstract]. In *Proceedings of the FC 2003*; Wright, R., Ed.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2742, pp. 190–207.
18. Manevich, Y.; Akavia, A. Cross chain atomic swaps in the absence of time via attribute verifiable timed commitments. In Proceedings of the 2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P), Genoa, Italy, 6–10 June 2022; pp. 606–625.
19. Liu, Y.; Wang, Q.; Yiu, S.M. Towards Practical Homomorphic Time-Lock Puzzles: Applicability and Verifiability. In *Proceedings of the ESORICS 2022, Part I*; Atluri, V., Di Pietro, R., Jensen, C.D., Meng, W., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13554, pp. 424–443. [CrossRef]

20. Srinivasan, S.; Loss, J.; Malavolta, G.; Nayak, K.; Papamanthou, C.; Thyagarajan, S.A. Transparent batchable time-lock puzzles and applications to byzantine consensus. In *Proceedings of the IACR International Conference on Public-Key Cryptography*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 554–584.
21. Thyagarajan, S.A.K.; Bhat, A.; Malavolta, G.; Döttling, N.; Kate, A.; Schröder, D. Verifiable Timed Signatures Made Practical. In *Proceedings of the ACM CCS 2020*; Ligatti, J., Ou, X., Katz, J., Vigna, G., Eds.; ACM Press: New York, NY, USA, 2020; pp. 1733–1750. [CrossRef]
22. Thyagarajan, S.A.; Malavolta, G.; Schmid, F.; Schröder, D. Verifiable timed linkable ring signatures for scalable payments for monero. In *Proceedings of the European Symposium on Research in Computer Security*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 467–486.
23. Mahmoody, M.; Moran, T.; Vadhan, S.P. Publicly verifiable proofs of sequential work. In *Proceedings of the ITCS 2013*; Kleinberg, R.D., Ed.; ACM: New York, NY, USA, 2013; pp. 373–388. [CrossRef]
24. Boneh, D.; Bonneau, J.; Bünz, B.; Fisch, B. Verifiable Delay Functions. In *Proceedings of the CRYPTO 2018, Part I*; Shacham, H., Boldyreva, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2018; Volume 10991, pp. 757–788. [CrossRef]
25. Pietrzak, K. Simple Verifiable Delay Functions. In *Proceedings of the ITCS 2019*; Blum, A., Ed.; LIPIcs: St. Louis, MO, USA, 2019; Volume 124, pp. 60:1–60:15. [CrossRef]
26. Wesolowski, B. Efficient Verifiable Delay Functions. In *Proceedings of the EUROCRYPT 2019, Part III*; Ishai, Y., Rijmen, V., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11478, pp. 379–407. [CrossRef]
27. Freitag, C.; Komargodski, I.; Pass, R.; Sirkin, N. Non-malleable Time-Lock Puzzles and Applications. In *Proceedings of the TCC 2021, Part III*; Nissim, K., Waters, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 13044, pp. 447–479. [CrossRef]
28. Abadi, A.; Kiayias, A. Multi-instance Publicly Verifiable Time-Lock Puzzle and Its Applications. In *Proceedings of the FC 2021, Part II*; Borisov, N., Díaz, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12675, pp. 541–559. [CrossRef]
29. Katz, J.; Loss, J.; Xu, J. On the Security of Time-Lock Puzzles and Timed Commitments. In *Proceedings of the TCC 2020, Part III*; Pass, R., Pietrzak, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12552, pp. 390–413. [CrossRef]
30. Mahmoody, M.; Moran, T.; Vadhan, S.P. Time-Lock Puzzles in the Random Oracle Model. In *Proceedings of the CRYPTO 2011*; Rogaway, P., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6841, pp. 39–50. [CrossRef]
31. Bitansky, N.; Goldwasser, S.; Jain, A.; Paneth, O.; Vaikuntanathan, V.; Waters, B. Time-Lock Puzzles from Randomized Encodings. In *Proceedings of the ITCS 2016*; Sudan, M., Ed.; ACM: New York, NY, USA, 2016; pp. 345–356. [CrossRef]
32. Malavolta, G.; Thyagarajan, S.A.K. Homomorphic Time-Lock Puzzles and Applications. In *Proceedings of the CRYPTO 2019, Part I*; Boldyreva, A., Micciancio, D., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11692, pp. 620–649. [CrossRef]
33. Hiraga, D.; Hara, K.; Tezuka, M.; Yoshida, Y.; Tanaka, K. Security Definitions on Time-Lock Puzzles. In *Proceedings of the ICISC 20*; Hong, D., Ed.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12593, pp. 3–15. [CrossRef]
34. Agrawalr, S.; Malavolta, G.; Zhang, T. Time-Lock Puzzles from Lattices. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 425–456.
35. Katz, J. *Digital Signatures*; Springer: Berlin/Heidelberg, Germany, 2010; Volume 1.
36. Chaum, D.; Pedersen, T.P. Transferred Cash Grows in Size. In *Proceedings of the EUROCRYPT'92*; Rueppel, R.A., Ed.; Springer: Berlin/Heidelberg, Germany, 1993; Volume 658, pp. 390–407. [CrossRef]
37. Camenisch, J.; Michels, M. Proving in Zero-Knowledge that a Number Is the Product of Two Safe Primes. In *Proceedings of the EUROCRYPT'99*; Stern, J., Ed.; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1592, pp. 107–122. [CrossRef]
38. Buchmann, J.; Hamdy, S. A survey on IQ cryptography. In Proceedings of the Public-Key Cryptography and Computational Number Theory, Warsaw, Poland, 11–15 September 2001; pp. 1–15.
39. Hafner, J.L.; McCurley, K.S. A rigorous subexponential algorithm for computation of class groups. *J. Am. Math. Soc.* **1989**, *2*, 837–850. [CrossRef]
40. Schindler, P.; Judmayer, A.; Hittmeir, M.; Stifter, N.; Weippl, E.R. RandRunner: Distributed Randomness from Trapdoor VDFs with Strong Uniqueness. In Proceedings of the NDSS 2021, online, 21–25 February 2021; The Internet Society: Reston, VA, USA, 2021.
41. Intel. Export Compliance Metrics for Intel Microprocessors. 2022. Available online: https://www.intel.com/content/www/us/en/support/articles/000005755/processors.html (accessed on 22 November 2022).
42. Hennessy, J.L.; Patterson, D.A. *Computer Architecture: A Quantitative Approach*; Elsevier: Amsterdam, The Netherlands, 2011.

symmetry

MDPI

Article

Authentication Framework for Augmented Reality with Data-Hiding Technique

Chia-Chen Lin [1,*], Aristophane Nshimiyimana [1], Morteza SaberiKamarposhti [2] and Ersin Elbasi [3,*]

[1] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan; aristophanenshimiyimana@gmail.com
[2] Faculty of computing and informatics (FCI), Multimedia University (MMU), Cyberjaya 63100, Malaysia; msaberyk@ieee.org
[3] College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait
* Correspondence: ally.cclin@ncut.edu.tw (C.-C.L.); ersin.elbasi@aum.edu.kw (E.E.); Tel.: +886-423924505 (ext. 8730) (C.-C.L.); +965-2225-1400 (ext. 2172) (E.E.)

Abstract: Currently, most existing research on the security of augmented reality (AR) primarily focuses on user-to-user or user-to-infrastructure authentication, aiming to establish secure communication and prevent potential attacks, such as man-in-the-middle attacks, among others. AR techniques enable users to interact with virtual information and objects within the real environment. However, creating virtual objects can be time-consuming, and unfortunately, malicious individuals may unlawfully copy these objects and integrate them into their own AR scenarios. It is essential to authenticate whether AR objects and the associated content they interact with are legitimately integrated. This paper proposes a novel method for authenticating AR-delivered content using data-hiding techniques with the symmetric property. The proposed method utilizes data hiding to embed data within the content that AR objects interact with, enabling the identification of illegal and tampered AR objects. A scenario involving an AR e-book is defined, and two data-hiding methods are applied to this scenario. The experimental results demonstrate that both data-hiding methods effectively identify the tampered image page of an AR e-book under different tone versions, new trigger image(s) added, or new image(s) replacement.

Keywords: data hiding; augmented reality; authentication; data integrity

1. Introduction

Augmented reality (AR) [1] is an interactive technology that enhances the human view of the real world by overlaying what the human eye sees with computer-generated content. It combines the real world and computer-generated sensory information, including visual, auditory, haptic, and olfactory. AR is prevalent since it uses computer vision and object recognition, and incorporates AR cameras into smartphone applications. By taking the image from the camera and processing it from computer vision in real-time, the surrounding environments of the user in the real world become interactive and digitally manipulated. The computer-generated content that can be the three-dimensional (3D) components of virtual and real objects delivers contextual information, such as GPS navigation, games, and social experiences. A growing number of AR applications have been used in various industries, such as retail, healthcare, and education, since AR is an increasingly popular interactive experience of online gaming and social media.

AR serves diverse purposes, including personal creation, education, and entertainment. Programmers can craft digital 3D components within AR, while end-users can also generate content within AR interactive environments. However, the widespread applicability of AR technology and the inherently digital nature of AR creations make them susceptible to copying and dissemination. Additionally, discussions on adequate digital

content protection measures for AR works are lacking. Consequently, creators employing AR technology in their digital works face a heightened risk of copyright infringement.

Recently, the security of AR issues has attracted scholars' attention. In 2016, Gaebel et al. introduced the Looks Good To Me (LGTM) authentication protocol. LGTM utilizes the unique hardware and contextual capabilities of AR headsets to integrate natural human trust mechanisms into digital authentication, aiming for both usability and security. It achieves this by combining localization-based wireless communication with facial recognition, allowing users to authenticate each other effectively [2]. In 2020, Wazir et al. developed an innovative authentication system that uses AR to create graphical doodle passwords in a 3D space. Users can draw their passwords in 3D by interacting with their smartphone screens. Authentication is achieved by comparing the size and coordinates of the last five doodles to verify the user's identity [3]. The following year, Bhalla et al. gathered data on spatial and behavioral patterns from AR headset users who were wearing AR headset. They proposed several advanced models along with machine learning techniques, including k-Nearest Neighbors, Random Forest, Support Vector Machines, and Bag of Symbolic Fourier Approximation Symbols. These models analyze user interactions with the AR environment to create unique user signatures, with an accuracy rate of approximately 92.675% [4]. In 2022, Stephenson et al. assessed the existing authentication mechanisms for AR/VR devices by reviewing research and practical implementations. They identified key factors for effective authentication on AR and VR devices, including usability, deployability, accessibility, and security, based on user experiences [5].

While researchers have developed user authentication systems for AR interactive environments [2–5], there is a noticeable absence of literature exploring the design of content copyright protection mechanisms and security concerns for AR objects within these AR interaction environments capable of responding to user actions. In this paper, we focus on potential solutions for the copyright protection of AR content. Unlike previous research [2–5] that has concentrated on user identity verification in AR protection issues, our study stands out as the first to address AR digital content protection specifically. We hope that through this research, we can contribute to a more comprehensive understanding of AR protection issues. The main contributions and novelties of this paper are summarized as follows:

(1) We propose a data-hiding-based authentication framework for AR content, and the integrity of AR content can be verified, which has not been discussed in other state-of-the-art schemes.
(2) The proposed authentication framework works with two different data-hiding strategies, and the experimental results confirm the usability and practicality of the proposed authentication framework.
(3) The proposed authentication framework withstands the luminance attack, color saturation attack, and object replacement attack.

To offer readers a clear picture regarding the AR scope which we focus on in this paper and make this paper self-contained, Section 2 first introduces the content structure of an AR e-book, and then introduces the copyright protection techniques that can be applied to verify the content of an AR e-book. Section 3 contains a detailed exploration of the proposed embedding and detection methods. The experimental results and their analyses appear in Section 4. Finally, we conclude the paper in Section 5.

2. Related Work

2.1. Content Structure of AR e-Book

Currently, there are lots of AR applications that have been designed, such as entertainment/gaming, education/training, retail/E-commerce, healthcare, navigation and wayfinding, and so on. For different AR applications, the copyright protections of AR content could be different. In this paper, we mainly focus on AR e-books. In order to study the digital content protection of AR objects, in this paper, we define the AR interactive

environment as an e-book containing AR objects. In an AR e-book, each page can be in PDF format, image format (e.g., JPEG and PNG), or HTML format. Each page may contain titles or subtitles, as demonstrated in Figure 1. Under a subtitle, different objects such as text, image, audio, or animation will be presented.

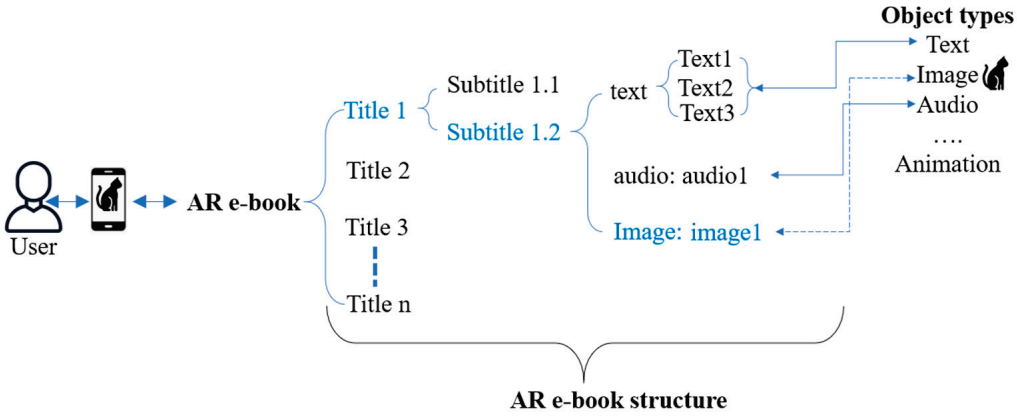


Figure 1. AR e-book structure and AR triggering example.

In an AR e-book, a specific area called a trigger image will drive the AR object. When the trigger image is scanned by a hardware device such as a mobile phone or camera, the AR app will recognize the trigger image and activate the corresponding AR object or AR content, such as 2D image, 3D image, audio, or even animation.

2.2. Copyright Protection Techniques

In general, conventional digital signature techniques [6] can employ asymmetric cryptography to verify the authenticity of digital images, and they also can provide non-repudiation. However, digital signatures require extra bandwidth and storage to attach the signature. Moreover, they cannot localize the tampered areas if the image is deemed unauthentic. Data hiding [7] provides the proper solutions to address these issues of copyright and authorship.

Data hiding, a science of concealed communication, embeds data into digital media for the purpose of secret communication, authentication, annotation, and copyright protection. Data hiding embeds the authentication data into the image, making it capable of authenticating itself without accessing the original image. The authentication data embedded and lumped in the image is sensitive to any modifications of the images and thus can localize the modifications as well as verify the content integrity of the image. Compared to digital signatures, the advantage of data hiding is obvious since the authentication data are embedded into the image rather than appended to the image. Note that hiding data in a digital image inevitably destroys the image content. However, images provide satisfactory properties for data hiding since the visual redundancy of digital images makes it possible to embed the imperceptible data in the image. The existence of embedded data can be concealed by using data-hiding algorithms. Therefore, it is intuitively clear that the resulting image (stego image) is perceptually equal to the original image.

Lin et al. [8] proposed an efficient data-hiding method for image tamper detection and recovery. They embedded the 2-bit authentication data and 6-bit recovery data into each 2×2 image block. A hierarchical detection structure was used to ensure the accuracy of tamper localization. Their method is effective in resisting the collage attack and VQ attack. In 2011, Zhang et al. [9] proposed two self-embedding data-hiding schemes by using a reference-sharing mechanism. The embedded data are a reference derived from the original principal content in different regions. After verifying the tampered blocks, both the embedded data and the original image content in the reserved area are used to recover the tampered blocks. In 2013, Chang and Tai [10] proposed a block-based data-

hiding scheme for image tamper detection and recovery. For each 2×2 image block, the data needed to be embedded is composed of the 6-bit recovery data and the 2-bit authentication data. They used the 2-LSB embedding scheme to embed data into the image. Their method can resist collage attacks, VQ attacks, and constant-average attacks [11]. The accuracy of tamper localization needs to be improved since the authentication data are only 2 bits for each block. Sarreshtedari and Akhaee [12] presented a data-hiding method for the purposes of detecting the tampered area and recovering the lost content in the tampered regions. They used Reed–Solomon codes to encode the embedded data in order to help the RS-code decoder retrieve the original source encoded image for self-recovery. The ability of self-recovery depends on the correcting ability of this error correction code.

In 2016, Qin et al. [13] proposed a self-embedding data-hiding scheme for tampering recovery. They used a reference-data interleaving mechanism and the adaptive LSB layer selection for data hiding. The reference data needed to be embedded are derived from the interleaved and scrambled MSB bits of original image pixels. They also provided detailed analyses for the theoretical values of watermarked image quality and perfect recovery probability. To improve the embedding efficiency, Lin et al. [14] proposed a hybrid image authentication scheme with the help of absolute moment block truncation coding (AMBTC). They used different embedding methods for smooth and complex blocks. The authentication data are embedded into the AMBTC compressed codes, and thus their method gives a significantly improved embedding capacity. Lin et al. [15] presented a reversible data-hiding method based on AMBTC. The authentication is reversible in the sense that the hiding distortion can be removed after image authentication. In 2018, Tai and Liao [16] proposed a self-embedding watermarking method for image authentication. The authentication data and recovery data are generated from the wavelet transform. Therefore, the tampered regions can be effectively recovered with a high image contrast.

In 2020, Yu et al. [17] proposed an adaptive bit-plane data-hiding method by using a parity check matrix based on matrix coding to enhance the hiding performance. In 2023, Nazir et al. [18] proposed a blind watermarking scheme for image authentication and protection. Data hiding on color RGB components is centered on singular value decomposition (SVD) and discrete wavelet transform (DWT). They encrypted SVD components and used a logistic map along with the hyperchaotic system to achieve confidentiality and avoid false positive problems. Li et al. [19] provided an ownership verification technique for deep neural networks (DNNs). They proposed a new framework that not only can efficiently resist linear functionality equivalence attacks but also helps the existing white-box watermarking schemes to enhance their robustness. Tang et al. [20] presented a two-stage reversible data-hiding method that embeds a robust watermark into the selected Pseudo-Zernike moments. The proposed adaptive normalization method achieves an invariance to pixel amplitude variation and gives a balance between robustness and imperceptibility. Anand and Singh [21] proposed a dual watermarking scheme for the CT scan images of COVID-19 patients. They used the Electronic Patient Record (EPR) text and medical image as multiple watermarks to ensure a high level of authentication. In order to obtain better image quality and higher hiding capacity, Chang et al. [22] proposed a data-hiding scheme based on the turtle shell. In their method, a secret digit is embedded into each cover pixel pair by looking for the turtle shell. Chen et al. [23] embedded the authentication data into AMBTC compressed codes by using the turtle shell data-hiding scheme.

For AR applications, Lee et al. [24] used the data-hiding technique to propose a tracking system based on invisible markers. The marker is embedded into the AR contents. It can be extracted from the scene image by using a general camera. Although the invisible watermarking technique is adopted, it is used to serve as the trigger image instead of copyright protection. In 2019, Li et al. [25] presented an AR-based data-hiding architecture, which combines data hiding and deep learning for secret communication. The secret

data can be transmitted in various formats with a higher embedding rate due to the property of AR. Since they used AR as a cover media to transmit the confidential message(s) via the pre-shared secret keys and reveal model, there is no copyright protection issue involved. In 2021, Bhattacharya et al. mentioned blockchain is able to prevent assets from being replicated for IoT-based smart factories because blockchain can help create and provide the identity of assets [26]. However, their research scope is defined as an IoT-based smart factory and a private blockchain platform can be set up to manage all assets within an IoT-based smart factory or multiple IoT-based smart factories belonging to the same company. Although blockchain technology has been recognized as a solution for asset identification issues within enterprises [26], how to utilize blockchain technology to address the problem of illegal misuse in AR electronic publishing (i.e., AR e-books) remains to be explored. This is because applying blockchain to AR e-publishing requires an authority to pre-assign identification numbers to each AR object within an AR e-book. Without such pre-assigned identification, the blockchain framework, which relies on object numbering for identification, will not be able to effectively recognize AR objects.

Considering the potential actions of malicious attackers, it is important to recognize that they might attempt to tamper with specific content within an AR e-book. For instance, on a page featuring two trigger images to activate two AR objects, a malicious attacker could overlay a new image to conceal one trigger image, thereby deactivating the corresponding AR object. Subsequently, they could falsely assert the ownership of the altered AR e-book. Alternatively, a malicious attacker might retain all the trigger images but alter the remaining content of an AR e-book, still asserting ownership. Considering the attack scenarios described above, individuals with malicious intent can easily plagiarize AR works with minimal effort, leading to unjust gains. To counteract such attacks, this paper proposes an AR content authentication framework along with two data-hiding strategies to verify AR content. The authentication data embedded into AR content makes it possible to verify the integrity of AR content and provide the ability to localize the tampered regions. The AR authentication can be easily performed through AR cameras.

3. The Proposed AR Content Authentication Framework

AR integrates computer vision, object recognition, and AR cameras into mobile applications. The proposed AR content authentication framework aims to verify AR content directly without requiring access to the original AR object file through AR cameras. It consists of two phases, one is the data-hiding phase depicted in Figure 2, and the other is the data extraction and tamper detection phase depicted in Figure 3. Data extraction is the inverse operation of data embedding, and tamper detection is the verification based on the extracted authentication code. The details regarding the proposed AR content authentication framework are outlined below. Based on our proposed authentication framework, two different data-hiding strategies are proposed: one is an LSB-based relative reference hiding strategy and the other is a DWT-based hiding strategy. The detailed descriptions of data embedding, and data extraction and tamper detection for LSB-based relative reference hiding strategy are introduced in Sections 3.1 and 3.2. The detailed descriptions of data embedding, and data extraction and tamper detection for DWT-based hiding strategy are introduced in Sections 3.3 and 3.4. It is noted that in the following paragraphs, the trigger image is denoted as regions of interest (ROIs), and the rest area on a page of an AR e-book is denoted as the region of non-interest (RONI) as shown in Figure 4.

Figure 4 demonstrates a page of the AR e-book for example; there are two regions of interest (ROIs), and the rest area is the region of non-interest (RONI). The ROI can be used to trigger the AR object, and the ROIs can be established in sub-sequential with corresponding 3D contents or animation files according to the script defined by the author.

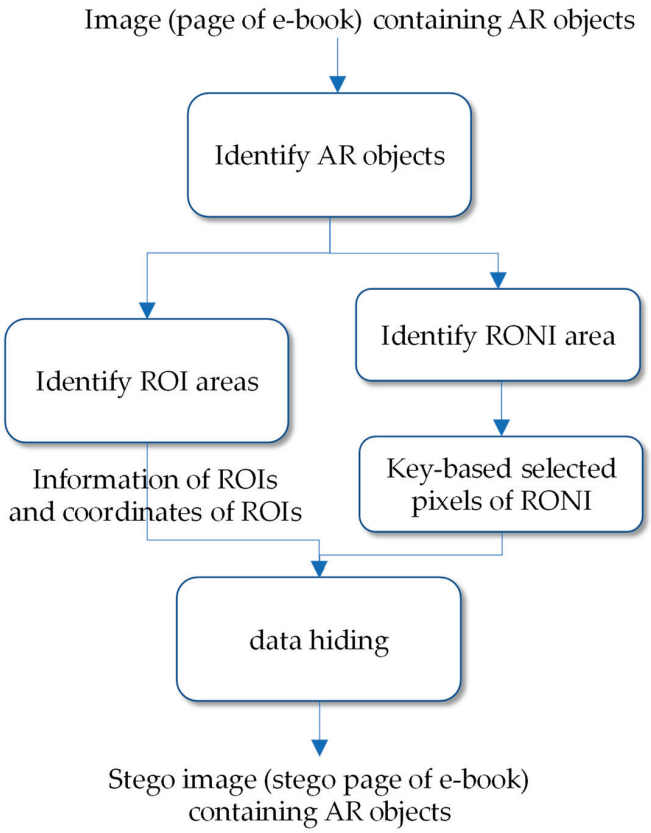


Figure 2. Flowchart of the proposed data-embedding phase for AR content.

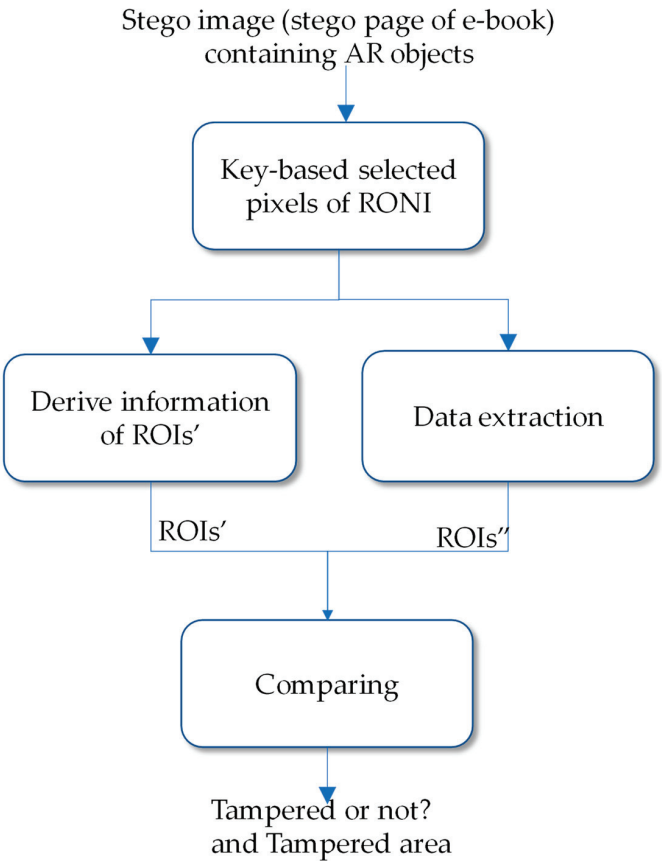


Figure 3. Flowchart of the data extraction and tamper detection phase.

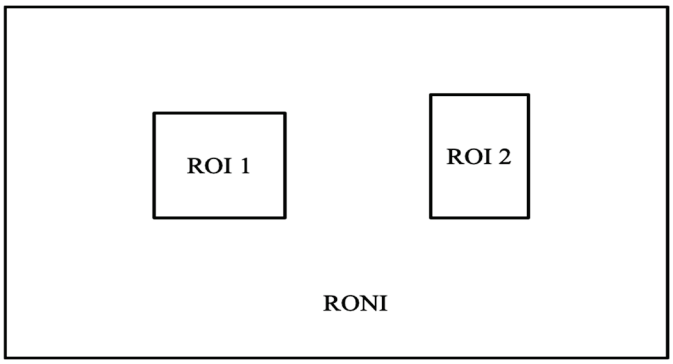


Figure 4. Example of the page of the AR e-book containing ROIs and RONI.

3.1. Data-Embedding Phase-LSB-Based Relative Reference Hiding Strategy

As mentioned earlier in Section 3, ROI represents the trigger image, and a page of the AR e-book consists of one or more ROIs, with the remaining area termed RONI. For clarity in the subsequent discussions, we will abbreviate the term "cover image" to CI when referring to the pages of the AR e-book. Similarly, we will use the abbreviation TI to denote the "trigger image," which activates the AR object and is situated within the ROI. Once the data-hiding phase is completed, the stego page of the AR e-book is obtained and it is noted as "stego image" and SI for short.

In the first data-hiding strategy, we proposed the relative reference strategy to conceal TI into the RONI of CI. The relative reference-based data-hiding algorithm is described as follows:

Input: Cover image (CI) and Trigger image (TI)
Output: Stego Image (SI)

Step 1: Identify the TI from CI and denote its location area as ROI.
Step 2: Calculate the Global Average (GA) for the Green channel and denote as GA_G of pixel values in CI according to Equation (1)

$$GA_G = \sum_{i=0,j=0}^{i=w,j=h} Gp_{ij} \Big/ w \times h \tag{1}$$

where $Gp_{i,j}$ indicates the pixels located at the Green channel in the CI. w is the width and h is the heigh of the CI.

Step 3: Generate an Authentication Bitstream (ABS) for the Green channel of TI by comparing each pixel value located at the Green channel of the OI with the corresponding GA_G according to Equation (2).

$$ABS_G = \begin{cases} 1, \; if \;\; GA_G \leq Gp'_{ij}, \\ 0, \quad otherwise \end{cases} \tag{2}$$

where Gp'_{ij} indicates the pixel value located at the Green channel of the TI. i is ranged between 1 and w', j is ranged between 1 and h', w' is the width, and h' is the height of the TI.

Step 4: Convert the coordinate of TI into binary stream and denote as coTI. Concatenate coTI and ABS_G to obtain secret stream S_G for the channel Green in CI, respectively.
Step 5: Generate a key to select pixels from the NROI of CI to conceal n copies of the size of secret stream S with LSB substitution.

Once Steps 1–5 are completed, the stego image which contains the ROI coordinate and authentication data is obtained.

Example of Data-Embedding Phase-LSB-Based Relative Reference Hiding Strategy

Here, an example is demonstrated to explain Section 2.1 for data-hiding procedures with a relative reference strategy. In this example, we assume that CI is sized 8×8 pixels as shown in Figure 5 and the pixels located in Red, Green, and Blue channels map to TI

that is sized as 2 × 2 pixels shown within the black bold border in Figure 5b are shown in Figure 6.

150	130	140	120	110	100	90	80
160	140	130	110	100	90	80	70
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	150	130	140	120
110	90	80	60	160	140	130	110
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**a**)

200	190	180	170	160	150	140	130
190	180	170	160	150	140	130	120
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	200	190	180	170
110	90	80	60	190	180	170	160
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**b**)

100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	70	80	170	160	150	140	130
110	30	40	140	150	160	170	180
100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	190	180	170	160	150	140	130
110	130	120	140	150	160	170	180

(**c**)

Figure 5. (**a**) The Red channel pixel values in CI, (**b**) the Green channel pixel values in CI, and (**c**) the Blue channel pixel values in CI.

120	110
110	100

(a)

120	110
110	100

(b)

70	80
30	40

(c)

Figure 6. (**a**) Red channel pixel values in TI, (**b**) Green channel pixel values in TI, and (**c**) Blue channel pixel values in TI.

Moreover, we assume the 2 × 2 TI shown in Figure 6b is located at the ROI framed with the dark line shown in Figure 5b. Figure 7 demonstrates three TI's locations in three channels.

150	130	140	120	110	100	90	80
160	140	130	110	100	90	80	70
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	150	130	140	120
110	90	80	60	160	140	130	110
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**a**)

200	190	180	170	160	150	140	130
190	180	170	160	150	140	130	120
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	200	190	180	170
110	90	80	60	190	180	170	160
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**b**)

100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	70	80	170	160	150	140	130
110	30	40	140	150	160	170	180
100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	190	180	170	160	150	140	130
110	130	120	140	150	160	170	180

(**c**)

Figure 7. (**a**) TI in CI Red channel, (**b**) TI in CI Green channel, and (**c**) TI in CI Blue channel.

Take CI Green channel shown in Figure 5b for example, the first three pixels in the Green channel of CI are 200, 190, 180 and 170. Finally, GA_G is obtained as 118 according to Equations (1) and (2); and ABS_G is obtained as "1000" because four Green channel pixel values in TI are 120, 110, 110, and 100 as shown in Figure 7b, respectively, following a zigzag scan; and there are one pixel (120) is larger than GA_R = 118 but there are three pixels (110, 110, 100) are less than GA_R = 118. After pending ABS_G as "1000" to the LSB of the pixels in the Green channel in CI with LSB substitution without considering the coordinate of TI, the first pixels of the stego CI in the Green channel are denoted as 201, 190, 180, and 170 as shown in Figure 8b, but the rest two channels (Red and Blue) without modification is shown in Figure 8b,c. Comparing Figures 7b and 8b, the LSB substitution can be easily followed.

150	130	140	120	110	100	90	80
160	140	130	110	100	90	80	70
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	150	130	140	120
110	90	80	60	160	140	130	110
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**a**)

201	190	180	170	160	150	140	130
190	180	170	160	150	140	130	120
140	120	110	90	80	70	60	50
130	110	100	80	70	60	50	40
120	100	90	70	200	190	180	170
110	90	80	60	190	180	170	160
100	80	70	50	140	120	110	90
90	70	60	40	130	110	100	80

(**b**)

100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	70	80	170	160	150	140	130
110	30	40	140	150	160	170	180
100	120	110	130	140	150	160	170
110	130	120	140	150	160	170	180
200	190	180	170	160	150	140	130
110	130	120	140	150	160	170	180

(**c**)

Figure 8. (**a**) Red channel pixel values in SI, (**b**) Green channel pixel values in SI, and (**c**) Blue channel pixel values in SI.

3.2. Data Extraction and Tamper Detection Phase-LSB-Based Relative Reference Hiding Strategy

The data extraction and tamper detection phase is very similar to the data-embedding phase after the user uses an AR camera or smartphone to capture the image called SI′ from the SI and then conduct the data extraction and taper detection. Once the SI′ is obtained, the user needs to use the secret key to find out the selected pixels from the Green channel in the SI′ that carry the coordinates of TI during the data-hiding phase. Once the ROI is located, the hidden ABS_G is extracted. The authentication data can be recalculated according to Equations (1) and (2) with the pixel values in TI and SI′. If the recalculated ABS_G′ = ABS_G, respectively, the SI′ is authenticated, and the AR object is retrieved and pops up in front of the user through the AR camera. Otherwise, the pixels in the ROI will be indicated as tampered.

3.3. Data-Embedding Phase-DWT-Based Hiding Strategy

Besides using relative reference hiding to generate the authentication data and embedding the coordinate of ROI and authentication code into selected pixels located in RNOI with LSB substitution, the second method consisting of data hiding, data extraction, and tamper detection based on DWT is described in Sections 3.3 and 3.4, respectively. In the DWT-based data-hiding phase, the CI first conducts Level-1 DWT transformation. The coefficients of the Green channels located at the LL of CI without covering the coefficients belonging to the ROI are denoted as GLL'_{CI}, respectively. Next, ABS_G denotes the Green channel pixel values in TI that are located at the ROI. Concatenate the coordinates of TI and ABS_G to obtain secret stream S_G for the Red coefficients in GLL'_{CI}. Finally, calculate the stego coefficients in RLL'_{CI}, GLL'_{CI}, and BLL'_{CI} according to Equation (3), Equation (4), and Equation (5), respectively.

$$Rc'_{ij} = \alpha \times S_R + (1 - \alpha) \times Rc_{ij} \tag{3}$$

$$Gc'_{ij} = \alpha \times S_G + (1 - \alpha) \times Gc_{ij} \tag{4}$$

$$Bc'_{ij} = \alpha \times S_B + (1 - \alpha) \times Bc_{ij} \tag{5}$$

where α is the weight to justify the robustness of S_R, S_G, and S_B to the coefficients in RLL'_{CI}, GLL'_{CI}, and BLL'_{CI}, respectively. α is ranged between 0 and 1. It is noted that the coefficients in GLL'_{CI} which are selected by a secret key from the right-bottom of the LL of CI without covering the coefficients belonging to ROI. However, the number of selected coefficients that are located at the right-down and left-up of the LL of CI are the same to ensure the symmetric property during data hiding. After the stego coefficient generation, the inverse DWT operation is conducted to obtain the SI. It is noted that the parameter, such as α in the DWT-based method, is ranged between 0 and 1 and it can be set as an initial value in the authentication framework; therefore, no extra data transmission is required.

3.4. Data Extraction and Tamper Detection Phase-DWT-Based Hiding Strategy

When DWT data hiding is conducted to conceal the coordinates of ROI and authentication code, only LL coefficients that are located at the right-bottom will be selected by a secret key. Because the LL coefficients mapped to ROI are not involved in the data-hiding phase, the pixel values of TI in the SI remain the same as those in the CI. Therefore, the data extraction is performed by conducting Level-1 DWT and then simply extracting the hidden data from the selected coefficients. Next, perform Equation (3), Equation (4), and Equation (5) with α pre-determined in the data-hiding phase to obtain the hidden S'_G. The extracted coordinates are used to extract the pixel values of the Green channel S_G in the ROI in the SI. The comparisons between the S'_G and S_G are performed. If $S'_G = S_G$, it means that the SI is authenticated. Otherwise, the coordinates are identified as tampered in the SI.

4. Prototype System

To prove our AR authentication framework is practical and can work well in the AR e-book scenario, the prototype system is implemented by using Python 3.9 and Unity hub with Vuforia and OpenCV. This prototype platform runs on a Mac OS-based PC equipped with a 2.9 GHz Dual-Core Intel® CoreTM i5 CPU and is enhanced by an Intel Iris GPU for accelerated performance. In addition, the AR e-book with our proposed authentication function is deployed on the Android device, which serves as the AR device, with a CPU of Octa-core 4 × 1.8 GHz Kryo 260 Gold and Adreno 509's GPU with Android version 10.

Following the flowcharts demonstrated in Figure 9, the user can use their AR reader installed in their smartphone to view the AR e-book; the smartphone's camera will capture the targeted image as shown in Figure 10; and then the authentication/verification and the tamper detection operation are conducted. It is noted that the detailed description of Step 3, as shown in Figure 9, corresponds to the data extraction and tamper detection phases discussed in Sections 3.2 and 3.3 , respectively, for various data-hiding strategies.

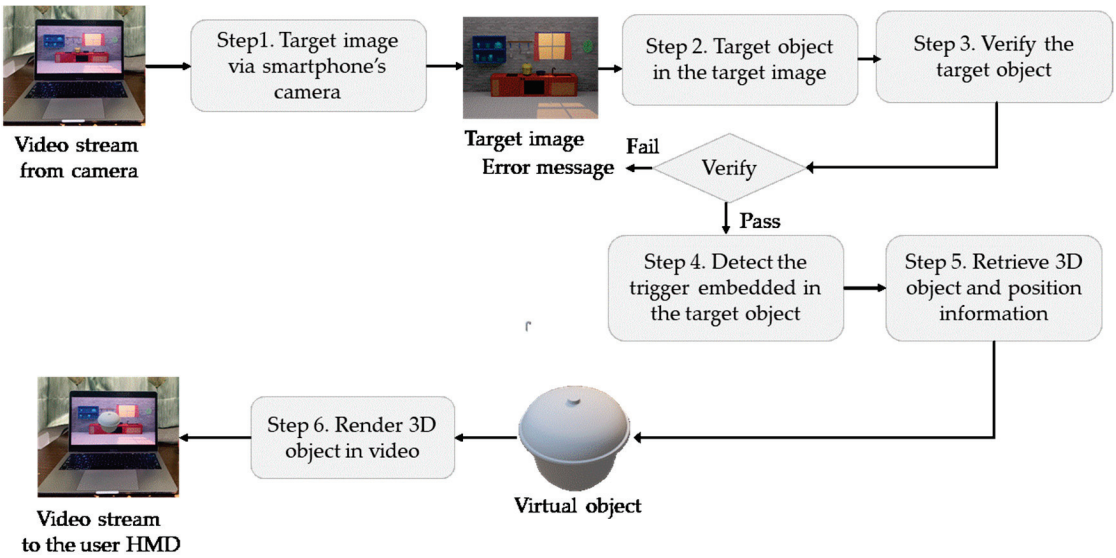


Figure 9. Authentication/verification and tamper detection model.

With our proposed hidden authentication code extracted from the RONI, if it is identical to that recalculated authentication code, a verification result such as "Verified" will appear. Meanwhile, the AR object pops up as shown in Figure 11. If the verification fails, a verification result such as "Unverified" will appear as shown in Figure 12a,b. It is noted that Figure 12a,b demonstrate two examples in which the verifications fail and the verification result is "Unverified".

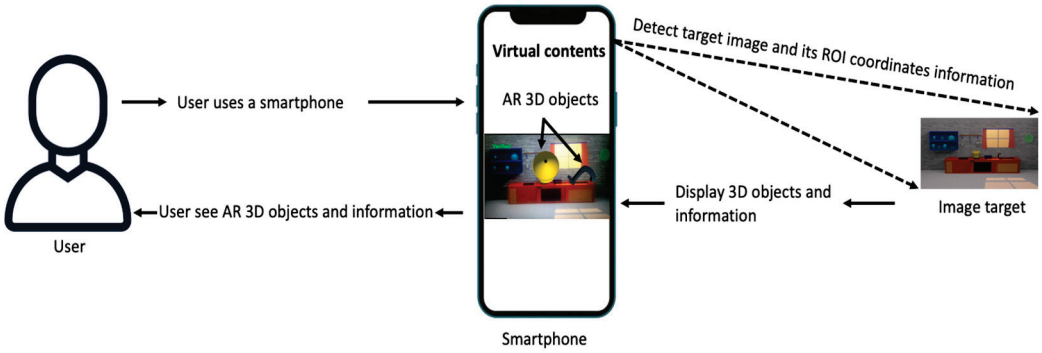


Figure 10. AR 3D object detection procedure.

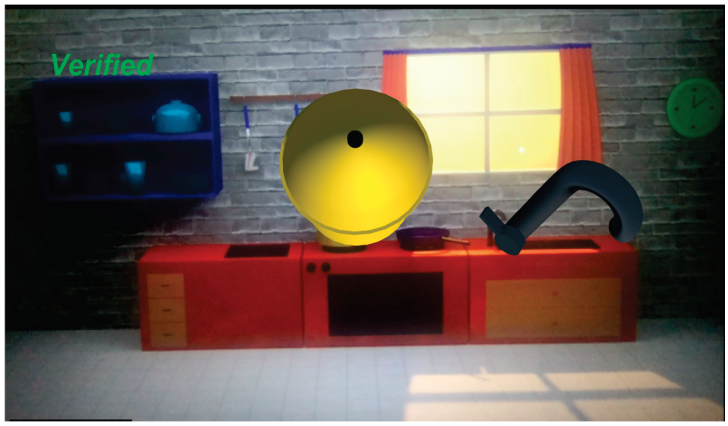
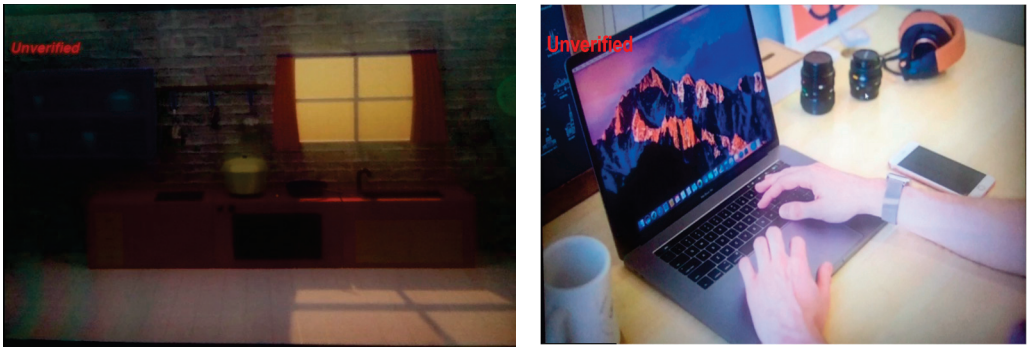


Figure 11. Verified stego image (SI).



(**a**) (**b**)

Figure 12. (**a**) Attacked stego image and the "Unverified" message appears, and (**b**) attacked stego image and the "Unverified" message appears.

5. Experimental Results

Based on our proposed AR content authentication framework, our hiding methods are designed in Section 2—one is the relative reference strategy with LSB substitution and the other is the DWT-based data-hiding strategy. To test the detection performance of the trigger image by using our AR content authentication with different data-hiding strategies, one image page of the AR e-book and the corresponding versions with different parameters are listed in Figure 13. It is noted that in Figure 13, we test five tampered images with different combinations of brightness (B) and contrast (C). With the test image page of the AR e-book shown in Figure 13a, several experiments are conducted in the following paragraphs by using the prototype system described in Section 3. In general, the visual quality of the stego image carrying the coordinates of ROI and authentication data is PSNR, which is defined in Equations (6) and (7).

$$MSE = \frac{1}{WH}\sum_{i=1}^{W}\sum_{j=1}^{H}\left(SI_{i,j} - CI_{i,j}\right)^2, \tag{6}$$

Peak signal–noise ratio (PSNR) is defined by the following (10):

$$PSNR = 10log_{10}\frac{(255)^2}{MSE}\ , \tag{7}$$

where $W \times H$-pixels indicate the size of CI and SI, respectively. To further evaluate the performance of our proposed authentication framework with two designed data-hiding strategies, the two images shown in Figure 13 carrying the trigger images such as obj image 1 and obj image 2 served as the test images.

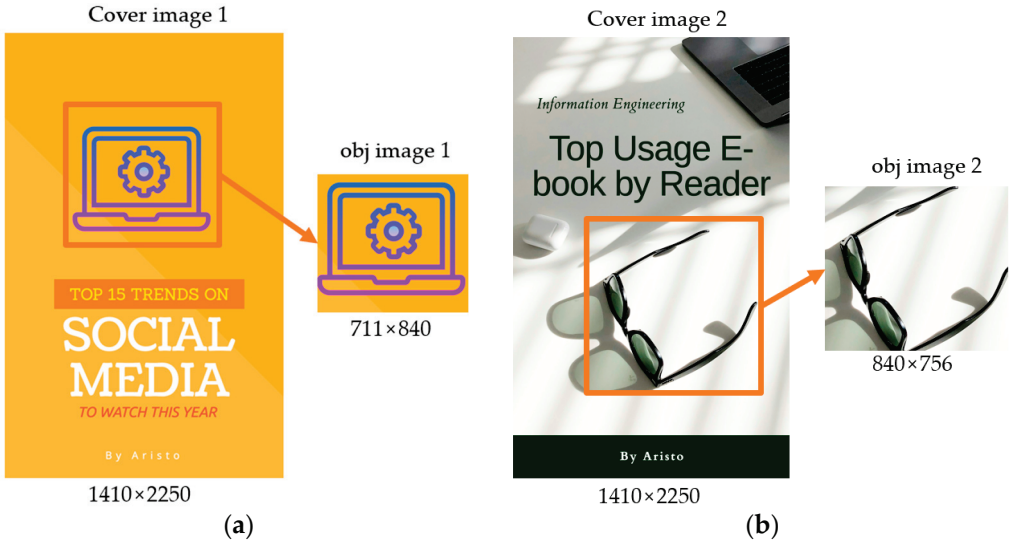


Figure 13. Two test cover images are sized as 1410 × 2250 pixels and (**a**) the trigger images (obj image1 and obj image2) are sized as 711 × 840 pixels and (**b**) the trigger images are sized as 840 × 756 pixels.

Due to potential variations in light sources and intensities when users review AR e-books, as well as potential color differences produced by AR devices during the review, we conducted experiments to evaluate our authentication scheme's performance across different brightness levels and color systems. The relevant experimental findings are presented in Tables 1–4. Note that Tables 1 and 2 present the image quality of the stego images and the image quality of the extracted object images with the relative reference hiding strategy described in Sections 2.1 and 2.2.

Table 1. Detection results and image quality of the stego images and tampered stego images using the relative reference strategy with the relative reference hiding strategy and LSB substitution (B: brightness; C: contrast).

Stego Image		Parameters	Similarity (%)	PSNR (dB)
Stego image 1	Untampered	B: 0 C: 0	100	59.87
	Tampered 1	B: −40 C: −40	64	31.74
	Tampered 2	B: −20 C: −40	57	34.94
	Tampered 3	B: 0 C: −20	68	36.71
Stego image 2	Untampered	B: 0 C: 0	100	59.58
	Tampered 1	B: −40 C: −40	58	28.40
	Tampered 2	B: −20 C: −40	66	29.49
	Tampered 3	B: 0 C: −20	66	37.82

Table 2. The PSNRs and MSEs of the extracted trigger image (obj image1 and obj image2) from the stego image under different parameters using the relative reference strategy and LSB substitution (B: brightness; C: contrast).

Extracted Obj Image		Parameters	PSNR (dB)	MSE
Stego image 1	Tampered 1	B: −40 C: −40	52.15	0.39
	Tampered 2	B: −20 C: −40	50.85	0.53
	Tampered 3	B: 0 C: −20	49.47	0.73
Stego image 2	Tampered 1	B: −40 C: −40	48.13	0.99
	Tampered 2	B: −20 C: −40	48.13	1.0
	Tampered 3	B: 0 C: −20	45.89	1.67

Table 3. Detection results and image qualities of stego images and tampered stego images using DWT-based data hiding (B: brightness; C: contrast, $\alpha = 0.01$).

Extracted Obj Images		Parameters	Similarity (%)	PSNR (dB)
Stego image 1	Untampered	B: 0 C: 0	100.0	62.91
	Tampered 1	B: −40 C: −40	70.55	55.00
	Tampered 2	B: −20 C: −40	81.09	57.93
	Tampered 3	B: 0 C: −20	83.00	58.00
Stego image 2	Untampered	B: 0 C: 0	100.0	57.95
	Tampered 1	B: −40 C: −40	71.73	50.00
	Tampered 2	B: −20 C: −40	72.64	52.00
	Tampered 3	B: 0 C: −20	75.58	54.00

Table 4. The PSNRs and MSEs of the extracted trigger image (obj image1 and obj image2) from the stego images and tampered stego images under different parameters using DWT-based data hiding (B: brightness; C: contrast).

Extracted Obj Images		Parameters	PSNR (dB)	MSE
Stego image 1	Untampered	B: 0 C: 0	53.68	0.27
	Tampered 1	B: −40 C: −40	50.00	0.35
	Tampered 2	B: −20 C: −40	51.00	0.30
	Tampered 3	B: 0 C: −20	52.00	0.28
Stego image 2	Untampered	B: 0 C: 0	49.66	0.70
	Tampered 1	B: −40 C: −40	46.00	0.75
	Tampered 2	B: −20 C: −40	47.00	0.73
	Tampered 3	B: 0 C: −20	48.00	0.72

Table 1 shows that the similarity rate of the trigger image (obj image1 or obj image2) with the relative reference strategy with the relative reference hiding strategy remains 100% and the visual quality of the stego image retains 59.87 dB and 58.58 dB for stego image 1 and stego image 2, respectively. However, when the brightness and contrast are justified accordingly, the detection performance is below 70% and the PSNRs are below 38 dB. However, even though the brightness and contrast of the stego images have been modified, Table 2 confirms that the image quality of the two extracted trigger images (obj image 1 and obj image2) still ranged between 45.89 dB and 52.15 dB. According to general image quality evaluation standards [14], if the PSNR exceeds 30 dB, the human visual system has difficulty distinguishing between the original image and the stego image. For example, as

shown in Tables 1 and 2, the image quality of the two extracted trigger images (obj image 1 and obj image 2) ranges from 57 dB to 62 dB regardless of whether the LSB-based or DWT-based data-hiding strategy is used. This is well above the 30 dB threshold. Therefore, it can be concluded that both proposed data-hiding strategies effectively maintain image quality.

Table 3 demonstrates that the similarity rate of the trigger image (obj image1 or obj image2) with the relative reference strategy with DWT-based data hiding remains 100% and the visual quality of the stego image retains 62.91 dB and 57.95 dB for stego image 1 and stego image 2, respectively. However, when the brightness and contrast are justified accordingly, the detection performance will remain above 70% and the PSNRs are above 50 dB. The similarity rate becomes less than 100%; this is because the image has been modified, therefore, the extracted results can not be the same as the original one. However, comparing the results of the two proposed data-hiding strategies shown in Tables 1 and 3, and Tables 2 and 4, it is noted that the average visual quality of the stego images with DWT is always higher than that using the relative reference strategy and LSB substitution. Moreover, the average PSNRS of the extracted trigger images (obj image1 and obj image2) is almost above 46 dB, which is higher than that with the relative reference strategy and LSB substitution.

To test the authentication performance and tamper detection performance of our AR content authentication with different data-hiding strategies, some modifications are made to the stego image of Figure 13a,b. Take Figure 14a for example—the original content of the stego image remained the same but two extra objects were illegally added and each tamper area is framed with the blue line. By contrast, in Figure 14b, one object of the stego image is replaced with a new object, and an extra object is also added to the stego image. Each tampered area of Figure 13b is framed with a red line as shown in Figure 14b.

Figure 14. Examples of two tampered stego images. (**a**) Original stego image 1 vs. tempered stego image 1; (**b**) Original stego image 2 vs. tempered stego image 2.

From Tables 5 and 6, we can see that if there is no attack, the hidden authentication data can be 100% extracted. However, when the extra object image is added to the stego image or the original trigger image is replaced with a new image, the data extraction rate will be changed and cannot remain 100%. In this case, the stego image will be determined as unverified. Take into account that those who plagiarize might attempt to alter the hue of an image page of an AR e-book, such as converting it into a blue-toned version, in order to avoid detection by the proposed AR content authentication scheme. Hence, we modified stego image1 and stego image2 to consist of only six individual colors each shown in Figures 15 and 16, aiming to test the efficacy of our proposed content authentication scheme in detecting such manipulations. The outcomes of our experiments are detailed in Tables 7 and 8. These findings indicate that the embedded authentication data can

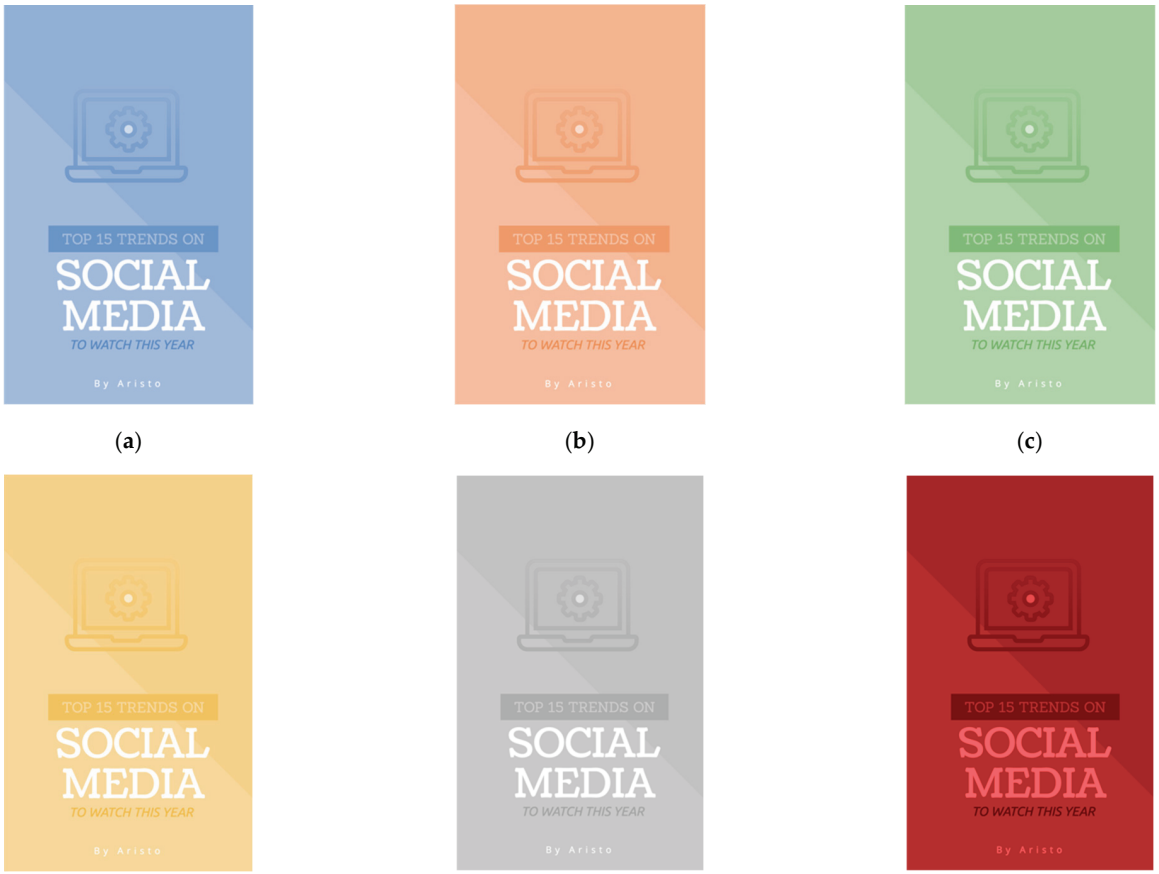
only be reliably extracted within a range of 40% to 80%, indirectly implying the potential manipulation of the image data.

Table 5. Data extraction rates and PSNRs of stego images and tampered stego images with the relative reference strategy and LSB substitution.

Extracted Obj Images		Similarity (%)	PSNR (dB)
Stego image 1	Untampered	100	59.87
	Tampered	74	39.84
Stego image 2	Untampered	100	59.58
	Tampered	52	32.37

Table 6. Data extraction rates and PSNRs of stego images and tampered stego images with DWT-based data hiding (α = 0.01).

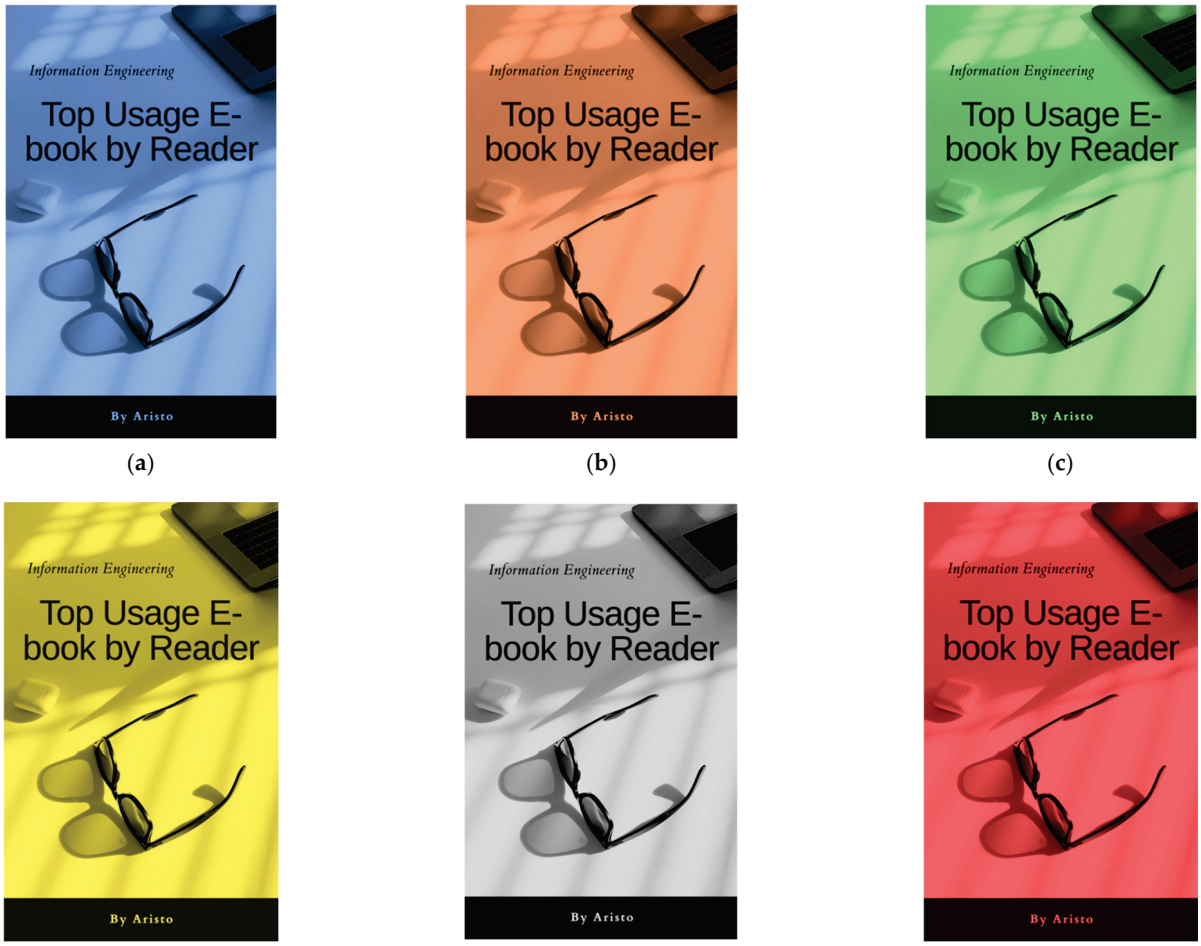
Extracted Obj Images		Similarity (%)	PSNR (dB)
Stego image 1	Untampered	100	62.91
	Tampered	94	55.00
Stego image 2	Untampered	100	57.95
	Tampered	70	50.00



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 15. Examples of converting the original stego image1 to monochrome with tone-mapping: (**a**) tone-mapping to blue color, (**b**) tone-mapping to orange color, (**c**) tone-mapping to green color, (**d**) tone-mapping to yellow color, (**e**) tone-mapping to gray color, and (**f**) tone-mapping to red color.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 16. Examples of converting the original stego image2 to monochrome with tone-mapping: (**a**) tone-mapping to blue color, (**b**) tone-mapping to orange color, (**c**) tone-mapping to green color, (**d**) tone-mapping to yellow color, (**e**) tone-mapping to gray color, and (**f**) tone-mapping to red color.

Table 7. Detection results of tempered SI under six different tones (color tones).

Image	Tone-Mapping	Similarity (%)
Tempered stego image 1	Blue	72.00
	Orange	57.04
	Green	57.82
	Yellow	40.00
	Gray	55.12
	Red	45.29

In addition to showing detection rates (similarity) with our proposed authentication framework and two data-hiding strategies, Tables 9 and 10 present the execution times for the embedding phase and the data extraction and tamper detection phase, with attacks depicted in Figures 14 and 15. The average execution time for both embedding and data extraction/tamper detection phases is approximately 22 s. The extraction and tamper detection phase takes slightly longer than the embedding phase because, after extracting the hidden authentication code, tamper detection must be performed to verify the SI.

Table 8. Detection results of tempered SI under six different tones (color tones).

Image	Tone-Mapping	Similarity (%)
Tempered stego image 2	Blue	47.39
	Orange	55.28
	Green	76.30
	Yellow	79.78
	Gray	69.52
	Red	64.23

Table 9. The execution time of the two data-hiding strategies for attacks demonstrated in Figure 14 (unit: seconds).

Data-Hiding Strategies	Image	Tamper	Embedding Phase	Data Extraction and Tamper Detection Phase
The relative reference strategy and LSB substitution	Stego image 1	Untampered	19.32	19.554
		Tampered	19.32	19.555
	Stego image 2	Untampered	20.10	20.331
		Tampered	20.10	20.373
DWT data hiding	Stego image 1	Untampered	21.30	21.765
		Tampered	21.30	21.768
	Stego image 2	Untampered	26.13	26.571
		Tampered	26.13	26.604

Table 10. The execution time of the two data-hiding strategies for attacks demonstrated in Figure 15 (unit: seconds).

Data-Hiding Strategies	Image	Tamper	Embedding Phase	Data Extraction and Tamper Detection Phase
The relative reference strategy and LSB substitution	Stego image 1	Untampered	19.32	19.554
		Tampered	19.32	19.619
	Stego image 2	Untampered	20.10	20.303
		Tampered	20.10	20.317
DWT data hiding	Stego image 1	Untampered	22.20	22.599
		Tampered	22.20	22.602
	Stego image 2	Untampered	26.40	26.796
		Tampered	26.40	26.799

At present, any mobile device with an AR app and camera can read the AR e-book and implement the authentication framework proposed in this study. To ensure the widespread applicability of AR applications, the two authentication code embedding and verification methods designed in this paper do not require high computational power. Tables 9 and 10 confirm that both our data-hiding strategies maintain an average computational time of around 22 s, making them well-suited for real-time AR e-book applications.

6. Conclusions

Augmented reality (AR) is an excellent way to bring people to a truly interactive experience with computer-generated components adding on people's view of the real world. AR not only visually changes natural environments but also provides additional information to users. Establishing the authenticity and integrity of AR content is thus very essential when problems are raised about the origin of AR content. In this paper, an AR content authentication scheme is proposed with two data-hiding strategies: the relative reference strategy with LSB substitution and DWT-based data hiding. The experimental results confirm that DWT-based data hiding offers more robustness of the hidden authentication data under either different tone-converting attacks or extra object insertion or image replacement. With the extraction rate of the hidden authentication, the image page of an AR e-book can be judged as verified or unverified. Once the image page is authenticated, the retrieved 3D objects pop up according to the extracted coordinates of the ROI. The experimental results show that the proposed method can verify the integrity of AR contents while sustaining superior tamper localization accuracy.

In this paper, we do not consider external lighting conditions when verifying the image pages of AR e-books, so our proposed method can effectively detect content attacks targeting hue adjustment. However, external lighting does affect the image tone during image capture, and thieves can manipulate the tone of the image page to evade content detection attacks. Balancing these conflicting concerns will be our next challenge. Furthermore, in the design of our current approach, we initially employed re-rotation techniques to address the non-forward image acquisition problem. Identifying more adaptive approaches in the future is another challenge that needs to be addressed.

Author Contributions: Conceptualization, C.-C.L.; software, A.N.; validation, C.-C.L., M.S. and E.E.; writing—original draft preparation, A.N.; writing—review and editing, C.-C.L.; visualization, A.N.; supervision, C.-C.L.; project administration, E.E.; C.-C.L. reviewed and edited and analyzed the data, performed the experiments, and wrote the paper; M.S. reviewed the paper; and E.E. supported the funding. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science and Technology Council NSC112-2634-F-005-001-MBK, 111-2410-H-167-005-MY2.

Data Availability Statement: Data available on request.

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

References

1. Feiner, S.; Macintyre, B.; Seligmann, D. Knowledge-based augmented reality. *Commun. ACM* **1993**, *36*, 53–62. [CrossRef]
2. Gaebel, E.; Zhang, N.; Lou, W.; Hou, Y.T. Looks good to me: Authentication for augmented reality. In Proceedings of the 6th International Workshop on Trustworthy Embedded Devices, Vienna, Austria, 28 October 2016; pp. 57–67. [CrossRef]
3. Wazir, W.; Khattak, H.A.; Almogren, A.; Khan, M.A.; Din, I.U. Doodle-Based Authentication Technique Using Augmented Reality. *IEEE Access* **2020**, *8*, 4022–4034. [CrossRef]
4. Bhalla, A.; Sluganovic, I.; Krawiecka, K.; Martinovic, I. MoveAR: Continuous biometric authentication for augmented reality headsets. In Proceedings of the CPSS '21: Proceedings of the 7th ACM on Cyber-Physical System Security Workshop, Virtual, 7 June 2021; pp. 41–52. [CrossRef]
5. Stephenson, S.; Pal, B.; Fan, S.; Fernandes, E.; Zhao, Y.; Chatterjee, R. Sok: Authentication in augmented and virtual reality. In Proceedings of the IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 22–26 May 2022. [CrossRef]
6. Schneier, B. *Applied Cryptography: Protocols, Algorithms and Source Code in C*; Wiley: Hoboken, NJ, USA, 2015.
7. Wu, M.; Liu, B. Data hiding in image and video. I. Fundamental issues and solutions. *IEEE Trans. Image Process.* **2003**, *12*, 685–695. [PubMed]
8. Lin, P.L.; Hsieh, C.-K.; Huang, P.-W. A hierarchical digital watermarking method for image tamper detection and recovery. *Pattern Recognit.* **2005**, *38*, 2519–2529. [CrossRef]
9. Zhang, X.; Wang, S.; Qian, Z.; Feng, G. Reference Sharing Mechanism for Watermark Self-Embedding. *IEEE Trans. Image Process.* **2011**, *20*, 485–495. [CrossRef] [PubMed]
10. Chang, Y.; Tai, W. A block-based watermarking scheme for image tamper detection and self-recovery. *Opto-Electron. Rev.* **2013**, *21*, 182–190. [CrossRef]

11. Chang, C.-C.; Fan, Y.-H.; Tai, W.-L. Four-scanning attack on hierarchical digital watermarking method for image tamper detection and recovery. *Pattern Recognit.* **2008**, *41*, 654–661. [CrossRef]
12. Sarreshtedari, S.; Akhaee, M.A. A Source-Channel Coding Approach to Digital Image Protection and Self-Recovery. *IEEE Trans. Image Process.* **2015**, *24*, 2266–2277. [CrossRef] [PubMed]
13. Qin, C.; Wang, H.; Zhang, X.; Sun, X. Self-embedding fragile watermarking based on reference-data interleaving and adaptive selection of embedding mode. *Inf. Sci.* **2016**, *373*, 233–250. [CrossRef]
14. Lin, C.-C.; Huang, Y.; Tai, W.-L. A novel hybrid image authentication scheme based on absolute moment block truncation coding. *Multimedia Tools Appl.* **2017**, *76*, 463–488. [CrossRef]
15. Lin, C.-C.; Liu, X.-L.; Tai, W.-L.; Yuan, S.-M. A novel reversible data hiding scheme based on AMBTC compression technique. *Multimedia Tools Appl.* **2015**, *74*, 3823–3842. [CrossRef]
16. Tai, W.-L.; Liao, Z.-J. Image self-recovery with watermark self-embedding. *Signal Process. Image Commun.* **2018**, *65*, 11–25. [CrossRef]
17. Yu, Z.; Lin, C.-C.; Chang, C.-C. ABMC-DH: An Adaptive Bit-Plane Data Hiding Method Based on Matrix Coding. *IEEE Access* **2020**, *8*, 27634–27648. [CrossRef]
18. Nazir, H.; Ullah, M.S.; Qadri, S.S.; Arshad, H.; Husnain, M.; Razzaq, A.; Nawaz, S.A. Protection-Enhanced Watermarking Scheme Combined With Non-Linear Systems. *IEEE Access* **2023**, *11*, 33725–33740. [CrossRef]
19. Li, F.-Q.; Wang, S.-L.; Liew, A.W.-C. Linear Functionality Equivalence Attack Against Deep Neural Network Watermarks and a Defense Method by Neuron Mapping. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1963–1977. [CrossRef]
20. Tang, Y.; Wang, S.; Wang, C.; Xiang, S.; Cheung, Y.-M. A Highly Robust Reversible Watermarking Scheme Using Embedding Optimization and Rounded Error Compensation. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 1593–1609. [CrossRef]
21. Anand, A.; Singh, A.K. Dual Watermarking for Security of COVID-19 Patient Record. *IEEE Trans. Dependable Secur. Comput.* **2023**, *20*, 859–866. [CrossRef]
22. Chang, C.C.; Liu, Y.; Nguyen, T.S. A novel turtle shell based scheme for data hiding. In Proceedings of the 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Kitakyushu, Japan, 27–29 August 2014; pp. 89–93.
23. Chen, C.-C.; Chang, C.-C.; Lin, C.-C.; Su, G.-D. TSIA: A Novel Image Authentication Scheme for AMBTC-Based Compressed Images Using Turtle Shell Based Reference Matrix. *IEEE Access* **2019**, *7*, 149515–149526. [CrossRef]
24. Lee, H.R.; Shin, J.S.; Hwang, C.J. Invisible marker tracking system using image watermarking for audgmented reality. In Proceedings of the 2007 Digest of Technical Papers International Conference on Consumer Electronics, Las Vegas, NV, USA, 10–14 January 2007.
25. Li, C.; Sun, X.; Li, Y. Information hiding based on Augmented Reality. *Math. Biosci. Eng.* **2019**, *16*, 4777–4787. [CrossRef] [PubMed]
26. Bhattacharya, P.; Saraswat, D.; Dave, A.; Acharya, M.; Tanwar, S.; Sharma, G.; Davidson, I.E. Coalition of 6G and Blockchain in AR/VR Space: Challenges and Future Directions. *IEEE Access* **2021**, *9*, 168455–168484. [CrossRef]

symmetry

MDPI

Article

AntiPhishStack: LSTM-Based Stacked Generalization Model for Optimized Phishing URL Detection

Saba Aslam [1,2], Hafsa Aslam [3,4], Arslan Manzoor [5], Hui Chen [6,*] and Abdur Rasool [1,*]

[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China; aslam@siat.ac.cn
[2] Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen 518055, China
[3] Cybex IT Group, Faisalabad 38000, Pakistan; hafsaaslam@cybex.com.pk or 6262103@studenti.unige.it
[4] Dibris Polytechnic Interschool Section, University of Genoa, 16126 Genoa, Italy
[5] Department of Mathematics and Informatics, University of Catania, 95123 Catania, Italy; arslan.manzoor@phd.unict.it
[6] School of Artificial Intelligence, Shenzhen Polytechnic University, Shenzhen 518055, China
* Correspondence: huichen@szpu.edu.cn (H.C.); rasool@siat.ac.cn or abdurrasoolphd@gmail.com (A.R.)

Abstract: The escalating reliance on revolutionary online web services has introduced heightened security risks, with persistent challenges posed by phishing despite extensive security measures. Traditional phishing systems, reliant on machine learning and manual features, struggle with evolving tactics. Recent advances in deep learning offer promising avenues for tackling novel phishing challenges and malicious URLs. This paper introduces a two-phase stack generalized model named AntiPhishStack, designed to detect phishing sites. The model leverages the learning of URLs and character-level TF-IDF features symmetrically, enhancing its ability to combat emerging phishing threats. In Phase I, features are trained on a base machine learning classifier, employing K-fold cross-validation for robust mean prediction. Phase II employs a two-layered stacked-based LSTM network with five adaptive optimizers for dynamic compilation, ensuring premier prediction on these features. Additionally, the symmetrical predictions from both phases are optimized and integrated to train a meta-XGBoost classifier, contributing to a final robust prediction. The significance of this work lies in advancing phishing detection with AntiPhishStack, operating without prior phishing-specific feature knowledge. Experimental validation on two benchmark datasets, comprising benign and phishing or malicious URLs, demonstrates the model's exceptional performance, achieving a notable 96.04% accuracy compared to existing studies. This research adds value to the ongoing discourse on symmetry and asymmetry in information security and provides a forward-thinking solution for enhancing network security in the face of evolving cyber threats.

Keywords: phishing detection; stack generalization; LSTM networks; anti-phishing; malicious URLs

1. Introduction

Phishing, a deceptive method through social and technical engineering, poses a severe threat to online security, aiming to obtain illicit user identities, personal account details, and bank credentials [1]. It is a primary concern within criminal activity, with phishers pursuing objectives such as selling stolen identities, extracting cash, exploiting vulnerabilities, or deriving financial gains [2,3]. The nuanced landscape of phishing techniques showcasing symmetry and asymmetry includes algorithms, domain spoofing, HTTPS phishing, SMS phishing, link handling, email phishing, and pop-ups. Attributes such as prefixes, suffixes, subdomains, IP addresses, URL lengths, '@' symbol, spear phishing, dual-slash attributes, ports, HTTPS tokens, request URLs, URL anchors, tag links, and domain age contribute to the multifaceted nature of phishing attacks [4]. Phishing perpetrators adeptly mimic

legitimate websites, particularly those related to online banking and e-commerce. This creates a symmetrical illusion that induces users to unwittingly divulge sensitive information, leading to various fraudulent actions [5,6].

A phishing attacker's role involves three specific duties: influencing target selection, sociological aspects, and technological infiltration [7]. As of March 2006, the Anti-Phishing Working Organization reported 18,480 significant phishing assaults and 9666 distinct phishing domains, resulting in substantial financial repercussions for businesses and affecting billions of site visitors [8]. Microsoft estimates the potential cost of computerized offenses on the global network to be a staggering USD 500 billion, underscoring the symmetrical impact of cyber threats on the financial ecosystem [9]. A single data breach could incur an average cost of approximately USD 3.8 million for organizations in 2018, highlighting the symmetrical consequences of security lapses. Data from the Anti-Phishing Working Group (APWG) reveal a notable increase in attack networks, with 180,768 identified during the first quarters of 2019, up from 138,328 in the fourth quarter of 2018 and 151,014 in the third quarter of 2018 [10]. The visual symmetry between benign and deceptive websites challenges human perception, making it difficult to distinguish between them. When visitors access these mimicked sites, critical information is stolen through scripting, underscoring the symmetrical vulnerability in human–computer interaction. The exponential growth in e-commerce consumers contributes to the escalating frequency of phishing attacks, carried out through various means such as malware, online platforms, and emails, creating a symmetrical escalation in cyber threats [11].

Researchers propose varied solutions to enhance symmetry in phishing detection. Some use a blacklist for identifying phishing sites [12]. However, this method fails to detect non-blacklisted phishing websites, introducing asymmetry, such as zero-day attacks. Heuristic-based detection analyzes website content and third-party service features, but potential service restrictions create asymmetry. Simultaneously, exploring online content and third-party features introduces temporal asymmetry due to its time-consuming nature [13]. Similarly, a hierarchical clustering method groups DOM vectors based on distance, limiting detection efficiency and suggesting a need for symmetrical analysis of URL features to enhance throughput [14].

URLs play a pivotal role in phishing attacks transmitted to users through various channels like emails and social media, presenting a facade of symmetry by appearing as a genuine URL [15]. Machine learning-based techniques emerge as symmetrical solutions among the available approaches for evaluating URLs. By familiarizing malicious URLs with categorization algorithms, these techniques effectively differentiate between phishing and benign URLs, introducing a symmetrical balance in the categorization process [16]. URL-based studies leverage a phishing tank database, a comprehensive collection tracking reported phishing URLs by various online security companies. While this database offers organized data categorization patterns, asymmetries arise when using categorization algorithms or machine learning for URL data, necessitating additional symmetrical URL management techniques [17]. Standard techniques like blacklisting, regular expression, and signature matching, although employed to identify phishing attempts, exhibit asymmetry by falling short in detecting unfamiliar URLs [4]. Continuous updating of database signatures to detect unexpected patterns in malicious URLs underscores the need for applying symmetrical machine learning-based research, particularly with deep learning models, for robust and symmetrical identification of malicious URLs [18].

Machine learning and deep neural networks have been pivotal in various research endeavors, showcasing substantial performance improvements [19–22]. In the context of phishing detection, the authors of [19] proposed a multidimensional feature engineering approach, harnessing a deep learning model (CNN-LSTM) and machine learning algorithms. This method integrated predictions using the XGBoost (eX-treme Gradient Boosting) algorithm, offering a solution to extract features from diverse dimensions for swiftly effective attack detection. However, the reported results indicated a decline in the false positive rate to 59%, signaling a reduction in the level of attack prediction. Another

study [20] introduced an end-to-end deep learning architecture grounded in natural language processing techniques to combat malicious URL phishing. The model aimed to classify benign and malicious URLs using character-level and word-level embedding in CNN networks. However, the model exhibited a lack of generalization on test data, indicating a need for improved accuracy and malicious URL detection ability. Wang et al. [21] presented the PDRCNN approach, designed to enhance phishing-detection efficiency by eliminating reliance on feature crawling from third-party services. Based on the LSTM network, this approach selects optimal features from the URL, employs CNN to distinguish characters influencing phishing, and predictions with machine learning classifiers. While reporting efficient performance, the mechanism's dependency on existing knowledge of phishing detection raises concerns about its susceptibility to errors in identifying the latest vulnerabilities.

In contrast to traditional machine learning methods that implicitly extract handcrafted features, deep learning approaches prove advantageous when faced with the challenge of professional phishers exploiting the multilayer features of URLs. To address this, stacking, an ensemble learning methodology integrating various machine learning algorithms and deep learning models, employs a metamodel to amalgamate predictions, enhancing overall performance. Initially employed for malware identification on mobile devices, the stacking approach demonstrated improved accuracy and the F measure [23]. We extended this stacking mechanism by designing two distinct phases, leveraging the symmetrical integration of other methods to enhance detection impact.

This paper leverages a deep learning neural network, long short-term memory (LSTM), introducing a novel stack generalization model named AntiPhishStack. The proposed model employs five optimizers in two phases to detect phishing URLs effectively. In the first phase, machine learning classifiers, coupled with k-fold cross-validation to mitigate overfitting, generate a mean prediction. The second phase utilizes a two-layered LSTM-based stack generalized model optimized for premier prediction in phishing site detection. Merging the mean prediction from Phase I with the premier prediction from Phase II, meta-classifiers, specifically XGBoost, deliver the final prediction. This stacking model significantly enhances phishing-detection accuracy by learning URL and character-level TF-IDF features, showing symmetrical capabilities. The AntiPhishStack model intelligently identifies new phishing URLs previously unidentified as fraudulent. Experimental evaluations on two benchmark datasets ([24,25]) for benign and phishing sites demonstrate robust performance, assessed through various matrices, including AUC-ROC curve, precision, recall, F1, mean absolute error (MAE), mean square error (MSE), and accuracy. Comparative analysis with baseline models and traditional machine learning algorithms, such as support vector machine, decision tree, naïve Bayes, logistic regression, k-nearest neighbor, and sequential minimal optimization, highlights the AntiPhishStack model's superior phishing-detection efficiency. Notably, this model offers the following significant advantages in achieving symmetrical advancements in cybersecurity:

i. Prior feature knowledge independence: The approach taken in this work embraces the concept of symmetry by treating URL strings as character sequences, serving as natural features that require no prior feature knowledge for our proposed model to learn effectively.
ii. Strong generalization ability: The URL character-based features are utilized for more robust generalization and check-side accuracy, and the multi-level or low-level features are combined in the hidden layers of the neural network to attain effective generalization.
iii. Independence of cybersecurity experts and third-party services: Our proposed stack generalization model autonomously extracts necessary URL features, eliminating the reliance on cybersecurity experts. Additionally, the AntiPhishStack model, reliant on URLs and character-level TF-IDF features, demonstrates independence from third-party features such as page rank or domain age.

The significant contributions of this paper are:

- Presentation of a two-phase stacked-based generalization model (AntiPhishStack) that breaks free from the necessity of prior feature knowledge for phishing site detection. The model achieves this by learning URL and character-level TF-IDF features.
- In Phase I, features are trained on the base machine learning classifier to generate the mean prediction. Meanwhile, Phase II employs two-layered stacked-based LSTM networks and five adaptive optimizers for premier prediction detection.
- The final prediction is established by developing a meta-classifier (XGBoost) classifying URLs into benign and phishing categories. Experimental results showcase the AntiPhishStack model's noteworthy performance on baseline models, utilizing symmetrically structured Alexa and PhishTank datasets.

The structure of the rest of the article is as follows: Section 2 deliberates the background research work of phishing detection; Section 3 introduces the AntiPhishStack proposed model; Section 4 delivers the experiments, and Section 5 presents the results and its evaluations, and Section 6 elaborates the conclusion and future work.

2. Literature Review

2.1. Phishing Detection

There are three different methods for phishing detection that are often utilized [26]. First and foremost, web-based phishing refers to imitating a legitimate web interface. Phishers trick users into providing credential information, believing it to be genuine. Second, attackers transmit phishing material via email using web-based methods. The third is a malware-based phishing assault in which attackers insert a harmful code into the user's system [16]. Adebowale [27] suggested a common approach in which particular users steal private information from websites and are called phishing users. This behavior is usually carried out through phony websites or malicious URLs, which are called fraudulent enterprises. Cybercriminals develop a well-planned phishing assault by engaging in fraudulent activities. After gaining access to the victim's computers, hackers may install malware or insufficiently secure users' systems. Acquisti [28] proposed that several techniques are advised to prepare and teach end users to detect phishing URLs to decrease the potential of phishing attacks. El-Alfy [29] suggested using the architecture of the node to train unsupervised and supervised algorithms. Phishing sites rely on feasibility neural networks and K-medoid clustering. The K-medoid method uses feature selection and modules to minimize storage capacities. The required technique achieves 96.79% accuracy on thirty characteristics. PHISH-SAFE, an anti-phishing solution, has suggested employing an SVM classifier to recognize phishing websites with greater than 90% accuracy [30].

One research study offered another anti-phishing approach based on a weighted URL token system that extracts identification keywords from a query webpage. A search engine was used to locate the target domain name by identifying keywords as search phrases and validating the query web page. Tan et al. presented an anti-phishing approach that collects keywords from a website and then uses a weighted URL token-based system [31].

2.2. Machine Learning-Based Detection

In the last couple of decades, machine learning has been vigorously applied for phishing detection through different models and methods for various purposes [32–34]. For instance, Wang [35] proposed ensemble classifiers for email filtering that eliminated five algorithms: Support Vector Machines, K-Nearest Neighbor, Gaussian Naive Bayes, Bernoulli Naive Bayes, and Random Forest Classifier. Finally, Random Forest enhanced its accuracy from 94.09 percent to 98.02 percent.

Rahman et al. [36] utilized six machine learning classifiers (KNN, DT, SVM, RF, ERT, and GBT). They applied three publicly accessible datasets with multidimensional attributes that could also be used to detect phishing attacks in several anti-phishing systems due to a lack of proper selection of machine learning classifiers. To quantify the classifier performance, they used the confusion matrix, precision, recall, F1-score, accuracy, and misclassification rates. It finds greater performance than Random Forest and Exceptionally

Randomized Tree, which achieved 97% and 98% accuracy rates for detecting phishing URLs, respectively. Gradient Boosting Tree performed best for the multiclass feature set, with a 92% accuracy.

Mogimi et al. [37] proposed a phishing detector that exhibits a lack of symmetry in its approach. The support vector machine (SVM) method was initially employed to train a phishing-detection model, followed by the decision tree (DT) approach to uncover concealed phishing. Although the suggested method achieves high true-positive rates (0.99) and low false-negative rates (0.001) in a large dataset, it operates under the assumption that phishing web pages exclusively utilize innocuous page content. This assumption lacks symmetry with real-world scenarios, where phishing pages may employ deceptive elements. Rao et al. [25] recently presented CatchPhish, a lightweight program that predicts URL validity without examining the website's content. The suggested framework uses the random forest classifier to retrieve the suspicious URL's hand-crafted and term frequency-inverse document frequency (TF-IDF) characteristics.

2.3. Deep Learning-Based Detection

Machine learning and deep learning are both types of AI. Machine learning, in essence, is AI that uses algorithms to read data, learn from those data, and make decisions based on what it has learned, allowing it to adapt automatically with little to no human intervention. However, deep learning uses artificial neural networks, a specialized branch of machine learning, to simulate how the human brain learns. It layers algorithms to build an "artificial neural network" to learn and decide for itself.

In phishing detection, deep learning provides an automatic, accurate, and fast means (artificial neural network) to identify the URLs as benign or legitimate. It uses different layers and weights assigned by the optimization functions to learn the huge complex dataset, which traditional machine learning algorithms find difficult to process. The development of deep learning algorithms, e.g., recurrent neural networks (RNNs), recurrent convolutional neural networks, and deep neural networks (DNNs), have lately been used for phishing detection. Though deep learning approaches are not often used in phishing detection due to the lengthy training period, they frequently give higher accuracy and automatically retrieve characteristics from raw data with no background experience [11,38]. Neural networks typically include one to two hidden layers. The number of layers varies in various deep learning applications. However, it needs almost 150 layers [20]. There are several guidelines for determining the number of layers, including two or fewer layers for basic datasets and additional layers for computer vision, time series, or complicated datasets [4,39,40].

Wang et al. [21] presented a rapid phishing website detection approach dubbed accurate phishing detection with recurrent convolutional neural networks (PDRCNNs) based solely on the website's URL. It converts URL information into a two-dimensional tensor and feeds the tensor to a deep learning neural network to categorize the original URL. They first employed a bidirectional long short-term memory (LSTM) network to extract global and local URL characteristics, followed by a convolutional neural network (CNN). YANG et al. [19] created a two-step multidimensional feature phishing-detection technique. The character sequence characteristics of the provided URL were retrieved and utilized for categorization by LSTM-CNN deep learning networks in the first phase. In the second phase, URL statistics data, webpage content characteristics, and deep learning classification results were merged to form multidimensional features. Yuan et al. [41] and Yuan et al. [36] developed a method introducing symmetry by combining character embedding (word2vec) with a URL structure to create vector representations of URLs. The URL was systematically divided into five components: the URL protocol, sub-domain name, domain name, domain suffix, and URL path. This approach enhances existing classification methods, training vector representations to identify phishing URLs symmetrically. Huang et al. [1] presented a deep learning-based approach for detecting phishing URLs called PhishingNet. They utilized a CNN network to extract character-level URL characteristics and an attention-

based hierarchical recurrent neural network (RNN) to retrieve word-level URL features. The features were then fused and trained using three convolutional layers and two fully connected layers.

The summary of these studies is categorized (C) in Table 1.

Table 1. The literature review summary compares phishing-detection studies utilizing machine and deep learning techniques.

C	Ref.	Method	Datasets	Findings	Limitations and Future Gaps
Phishing detection	[42]	Character embedding CNN and RF to classify phishing websites based on multi-level features.	PhishTank (47,210) and Alex (83,857)	95.49% on dataset D2	Reliance on URL features only and limited to character-level features.
	[29]	PNN and K-medoid clustering are combined and trained on pre-classified websites and relevant features.	11,055 phishing and benign websites	87% using address bar-related features	The potential impact of the Gaussian smoothing parameter on performance and the limited effectiveness of HTML- and JavaScript-based features.
	[43]	Two-level filtering mechanism using lightweight visual similarity and heuristic filtering to detect phishing.	PhishTank and Google, 100 search results	Matthew's correlation coefficient is 97%	Inability to detect variations in blacklisted sites and the generation of false negatives when encountering out-of-list legitimate sites.
ML-based detection	[37]	Rule-based system that extracts hidden knowledge to detect phishing attacks.	Dataset3 (103 phishing and 73 legitimate)	Average accuracy 90.51% with SVM and 1.35% error rate	Reliance on webpage content for feature sets may not account for attackers redesigning phishing web pages.
	[24]	Whitelists and blacklists for classifying legitimate and phishing web pages.	Ebbu2017 contains 73,575 URLs	Accuracy rate of 10.86% with DT	Limited dataset size (1400 items) and high acceptance for noisy data.
	[25]	Detects phishing sites using client-side, URL-based features, independent of third-party services, and fast computation.	Common crawl, Alexa database, PhishTank, total samples: 85,409	CatchPhish accuracy is 94.26% with Random Forest	Model misclassified some phishing sites hosted on free or compromised hosting servers.
DL-based detection	[40]	Mapas detects malware using API call graph patterns with a CNN model.	9000 malicious apps, 9000 benign from Google Play	Accuracy 91.27% with CNN	Excludes obfuscated apps that cannot extract API call graphs with FlowDroid.
	[21]	Uses RNN to extract global features and CNN for local features from URLs for phishing detection.	Alexa and PhishTank, 500,000 sample	Accuracy 93.48% (RNN) 95.03% (CNN)	Training time was too long, and unable to classify URLs if it is not semantics.
	[19]	Utilizes a CNN-LSTM with feature extraction for phishing detection.	PhishTank and dmoztools.net (989,021 URLs)	Accuracy is 94.41%	It is not supported for webpage code and webpage text detection.

2.4. Stack Generalization-Based Detection

Deep learning learns representation at several levels of abstraction by using layers of layered nonlinear projections. It has demonstrated superior performance in various applications, including natural language processing, computer vision, speech recognition, and so on [44–47].

The DNNs recommended [20] are trained with inferred deep stacking. The analyzed covers of previous outlines are updated as they had been at the end of each DNN training epoch, and the upgraded evaluated veils then provide further inputs to train the DNN in the

subsequent epoch. During the testing phase, the DNN generates expectations sequentially and repeatedly. In addition, it suggests using the L1 loss for training.

3. AntiPhishStack Proposed Model

The primary purpose of this model is to determine the best output through evaluation by applying the stacking technique and deep neural network to the processed dataset and to propose an optimized model based on that output. The AntiPhishStack model of stack generalization is illustrated in Figure 1.

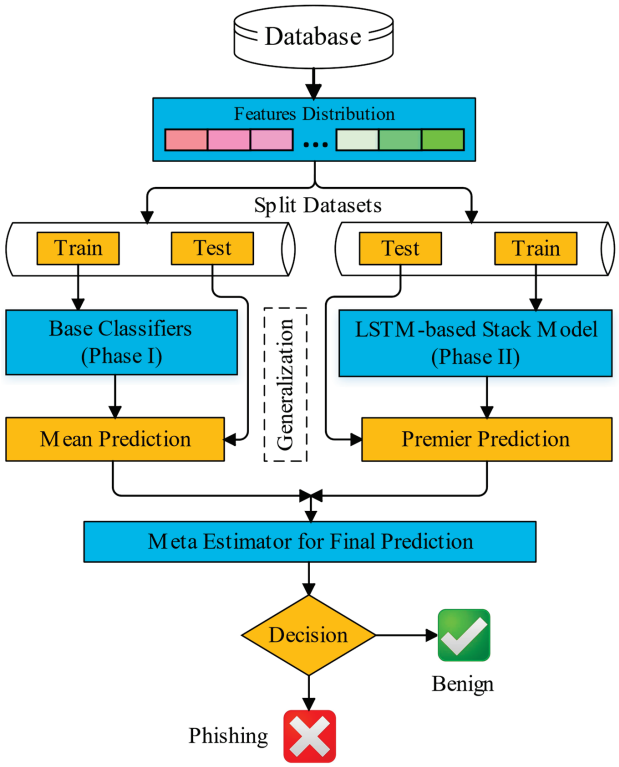


Figure 1. AntiPhishStack: proposed LSTM-based stock generalization model's flow.

Our model's flow has a five-level approach. The key steps are as follows:

i. Collection of datasets and feature distribution into URL features and character-level features.
ii. Dataset division into training and testing by 70:30 ratio, respectively.
iii. Construct the stack generalization model's first phase (Phase I) based on the machine learning-based model and calculate the mean prediction with the test dataset.
iv. Construct the second phase (Phase II) of the stack generalization model with the LSTM model based on adaptive optimizers and compute the performance evaluation with the test set.
v. Merge predictions and evaluations from both Phase I and Phase II for the ultimate prediction, enhancing symmetrically the recognition and determination of the phishing web pages.

The notations and meanings used in this paper are described in Table 2.

Table 2. Notations and their meanings.

Notations	Meanings
W_i	Weight factor of URLs
h_{t-1}	Hidden state of the $t-1$ instant
bb	Bias of each gate
i_t, f_t, o_t, and C_t	Input gate, forget gate, output gate, and unit status, respectively.
W_f, W_i, and W_o	Weight matrix of forget gate, input gate, and output gate, respectively.
x_t	Current input
f_l	Training loss function
γ	Complexity of each leaf
T	Number of leaves nodes

3.1. Datasets

The URLs were collected from a variety of sources (Alexa and PhishTank) [24,25]. URLs that were duplicated or did not survive were deleted before they were used to create a dataset. The typical URL elements, such as "http://", "https://", and "www.", were deleted. Inconsistent URL forms can easily impair the model's quality during training if the prefixes are not trimmed. The database management system (pgAdmin) was utilized in conjunction with Python to import the preprocessed data, and then the dataset was divided into two parts: 70% for training and 30% for testing. The distribution of legitimate and phishing URLs was as follows:

i. *Dataset 1 (DS1)*: Benign Yandex sites (https://yandex.com.tr/dev/xml/ (accessed on 3 December 2023)) and PhishTank phishing sites [24].
ii. *Dataset 2 (DS2)*: Benign sites from Common Crawl, the Alexa database, and phishing sites from PhishTank [25].

The datasets were selected for their diverse and current mix of benign and phishing URLs, ensuring robust model training. DS1 and DS2 offer a balanced representation of typical internet environments and specialized sources, respectively. This variety enhances the model's applicability and accuracy in real-world phishing detection. Meanwhile, the feature dataset was divided into 70% training and 30% testing datasets to ensure a balanced setup: 70% for training our AntiPhishStack model and 30% for robust testing on unseen data, aligning with standard machine learning practices.

3.2. Feature Distribution

Features and the capacity to use these features must be examined before examining the features selection section [48]. There are four major features and a total of 30 sub-features. Based on the details, each characteristic provides information on whether the website is phishing, legitimate, or suspect. This section contains the plans for highlighting the characteristics.

3.2.1. URL Features

A Uniform Resource Locator (URL) provides the location of online resources such as pictures, files, hypertext, and videos. In general, attackers attempt to build phishing URLs that look like reputable websites to users. Attackers use URL jamming tactics to mislead users into disclosing personal information that can be exploited against them. This research aims to detect phishing websites quickly, utilizing lightweight characteristics, i.e., the weight factor URL token system, inspired by [49]. For example, the segmentation of a URL (Figure 2) provides the different tokens and their final weights W_i for i-th distinct words and can be calculated as:

$$W_i = \frac{h_i}{n} \sum_{x=1}^{S} \frac{N_x}{x^2} \quad (1)$$

where h_i indicates the length of i-th distinct word, S denotes the total steps available for tokens, n shows the number of URLs from webpages, and N_x is the total number of i-th word occurrences in step S with respect to the x level.

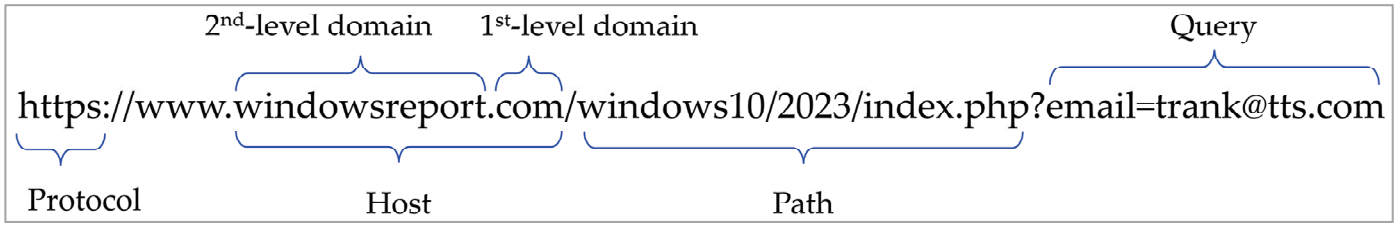


Figure 2. Tokenization of URL characteristics and components for the weight calculation.

Calculating this weight delivers the weight value of each URL assigned to neural network gates for phishing prediction. This was accomplished by extracting only characteristics from the URL rather than accessing the website's content. Figure 2 shows an example of URL characteristics for the weight.

The first component of the URL is a protocol (https, http, ftp, etc.), which is a set of rules that regulates how data are transported from data transmission. The second component is the location of the host IP address or resource. The hostname is separated into two parts: major domains and top-level domains (TLDs). The URL's hostname is comprised of the principal domain and the TLD. The hostname is followed by a port number, which is optional. The third component uses the path to identify the specific resource inside the domain accessed by a user. An optional field, such as inquiry, follows the path. The protocol, hostname, and URL path are appended to the base URL. The combination of the second domain and top-level domain names, known as the host domain, makes the URL unique. As a result, cybersecurity firms are working hard to identify the fraudulent websites used for phishing offenses by name. If a hostname is designated as phishing, an IP address can be banned to prevent it from accessing the web pages included within it.

It has the following sub-features, according to the dataset:

- IP Address: If an IP address is used instead of a domain name in the URL of a phishing website, the client may virtually be certain that someone is attempting to steal their credentials. From this dataset, 570 URLs with an IP address were discovered, accounting for 22.8 percent of the dataset, and a rule IP address is in the URL that is termed phishing; otherwise, it was suggested to be legitimate.
- Operate the @ Symbol: Web browsers usually ignore the section preceded by the @ sign. Because it is maintained separately from real-world addresses, finding 90 URLs with the '@' sign will provide just 3.6 percent of the total, according to the dataset.
- Operate the "//" symbol: As valid URLs, the "//" sign is used after HTTP or HTTPS. If the URL changes after the initial protocol declaration, it is called a phishing URL. The "//" sign is used to redirect to other websites.
- Domain name prefixes and suffixes separated by the "-" sign: A URL with the "-" sign in its domain name is a phishing URL. In general, verified URLs do not include the "-" sign.
- Use the "." sign in the domain: Use the "." sign in the domain. Adding a sub-domain with the domain name must include the dot. Consider it suspect if you drop out more than one subdomain, and anything greater than that will indicate phishing.
- HTTPS (secure socket layer): The majority of legal sites use the HTTPS protocol. Therefore, the age of the certificate is quite important when utilizing HTTPS. This necessitates the use of a trustworthy certificate.
- Favicon: A favicon might redirect clients to dubious sites when layered from an outside space. It is mainly used on websites and is a graphic picture.

3.2.2. Character-Level Features

Term Frequency-Inverse Document Frequency is abbreviated as TF-IDF. The TF-IDF score indicates a term's relative significance in the document and throughout the whole corpus. The TF-IDF score is made up of two terms: the first computes the normalized Term Frequency (TF), and the second computes the Inverse Document Frequency (IDF), which is calculated as the logarithm of the number of documents in the corpus divided by the number of documents in which the specific term appears [25,50].

$$TF(t,d) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document\ d}{Total\ number\ of\ terms\ in\ the\ document}, \tag{2}$$

$$IDF(t,D) = log_e\left(\frac{Total\ number\ of\ documents\ D}{Number\ of\ documents\ with\ term\ t\ in\ it}\right), \tag{3}$$

$$TF - IDF(t,d,D) = TF(d) * IDF(t,D). \tag{4}$$

TF-IDF vectors may be produced at many levels of input tokens (words, characters, and n-grams):

- Word-level TF-IDF: A matrix indicating the TF-IDF scores of each term in distinct texts.
- Character-level TF-IDF: A matrix indicating the TF-IDF scores of character-level n-grams in the corpus.
- N-gram-level TF-IDF: N-grams are the collection of N terms. This matrix indicates the TF-IDF scores of N-grams.

It should be mentioned that TF-IDF has been used in numerous studies to identify website phishing by examining URLs [25] to obtain indirectly related connections, target websites, and the validity of suspicious websites [51]. TF-IDF retrieves prominent keywords from the textual content. However, it has certain limitations. One of the limitations is that the approach fails when extracting mis-spelled terms. Because the URL might contain nonsensical words, it used a character-level TF-IDF method with a maximum feature count of 5000.

Furthermore, we measured the URL strings as character sequences by employing the idea from the literature [52]. This idea provides the advantage that the proposed model can train the URL character sequences as natural features that do not need prior feature knowledge to be learned by our proposed model. Our proposed AntiPhishStack model uses the stack generalization model to extract the local URL features from the URL character sequences. Finally, the URL will be classified by designing a meta-classifier for final prediction.

3.3. Stack Generalization Model

The stack generalization model is divided into two phases, as illustrated in the flow model (Figure 1).

3.3.1. Phase I

Based on the abovementioned characteristics, existing machine learning models were utilized directly to distinguish phishing and legitimate web pages. This paper proposes a stacking model (illustrated in Figure 3) for this purpose by merging various machine learning models, including support vector machine (SVM), naïve Bayes (NB), decision tree (DT), logistic regression (LR), K-nearest neighbors (KNNs), sequential minimal optimization (SMO), and XGBoost.

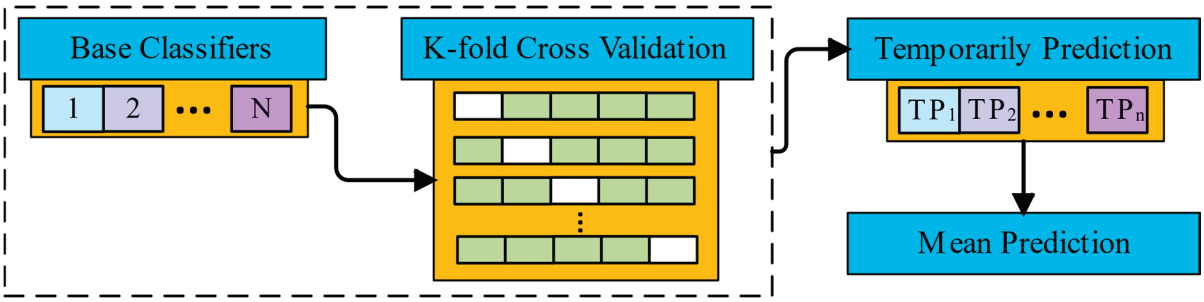


Figure 3. Phase I of the proposed stack generalization model.

The training set was split into Z copies, with $Z-1$ copies utilized for training and one copy used for testing. The training process was not terminated until each basic model had predicted the samples. This suggested system employs k-fold cross-validation to avoid overfitting for this training set and each fold of the train part that might be predicted using out-of-fold.

This suggested model uses a value of three to ten for k-fold cross-validation; after all, it delivers output using a test set. Following the temporary prediction (TP) acquisition, the mean prediction is obtained, which is strengthened by the test dataset validation. This time, it is comprehensive, with a fold approach required for estimating all figures on all folds utilized.

3.3.2. Phase II

The train segment was put to a two-layer neural network architecture of LSTM once the features from the training dataset were loaded. Because there are dependencies on immediately preceding entries in sequential phishing webpage data, LSTM is better suited to simulate phishing detection in this investigation. Meanwhile, it is explicitly designed to avoid the long-term dependency problem by storing the feature information in its memory cell. It can remove or add information to these call states and is regulated by structures called gates. These gates and corresponding operations/functions are presented in [53], while Phase II of the integrated stack generalized model is illustrated in Figure 4.

In the first gate (Forget gate), the information from the current input x_t and the previous hidden state h_t is passed through the sigmoid activation function. If the output value of the feature is closer to 0, it means forget, and closer to 1 means retain. The second gate, the input gate, decides what relevant feature (phishing or benign) can be added from the current step. The third gate, the control gate, decides which values will be updated (either 0 or 1), for which a *tanh* layer creates a vector of $\hat{C}_t$. The last gate, the output gate, determines the value of the next hidden state [54].

At time t, the LSTM cell's components are modified as follows:

i. Equation (5) represents the forgotten gate f_t with the sigmoid function σ. The weights W_f and bias b_f are applied to the concatenation of the previous layer's output h_{t-1} and the current layer's input x_t, represented as $[h_{t-1}, x_t]$. This concatenation forms a row vector, and Equation (5) describes the computation for f_t, considering the forgotten information from the cell state at time $t-1$.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{5}$$

ii. Save information in the cell state, which consists mostly of three parts:

   a. The Sigmoid layer's results are i_t entering the gate as information to be updated.

b. The *tanh* layer's freshly generated vector C_t is being added to the cell state. The previous cell state C_{t-1} is multiplied by f_t to forget the information, and the new party information $i_t * \hat{C}_t$ is totaled to create a cell state update.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\hat{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{7}$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \tag{8}$$

c. The output gate decides the output data. To process the cell state, first, the Sigmoid layer is used to identify which part of the information should be produced, and then *tanh* is used to process the cell state. The output value is the product of the two elements of the information.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t * tanh(C_t) \tag{10}$$

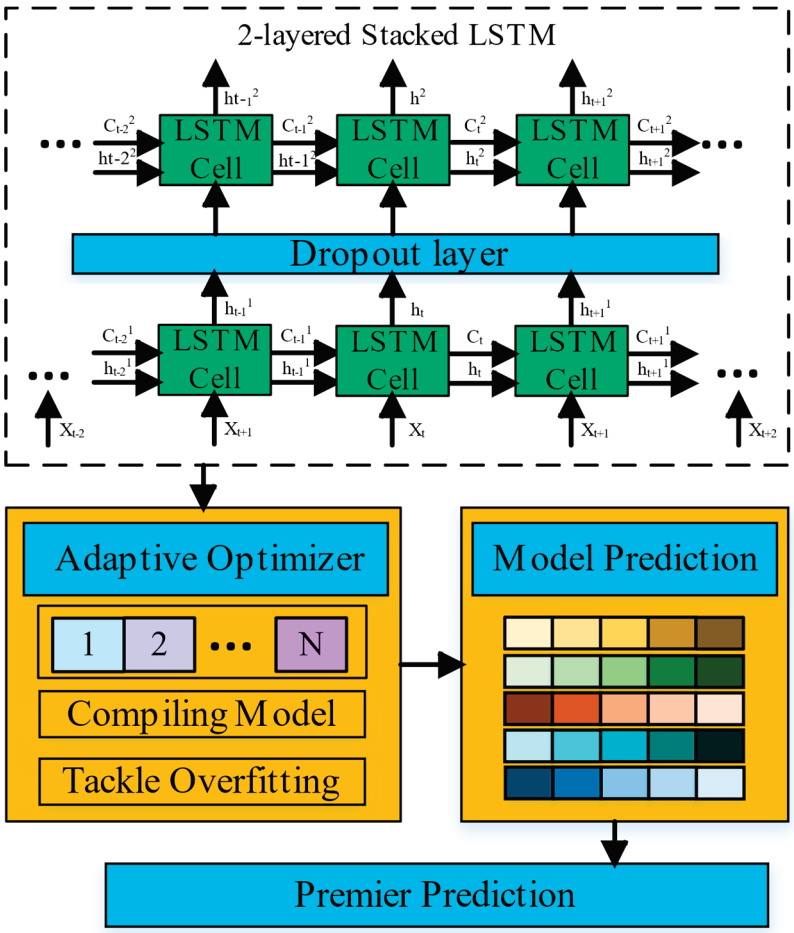


Figure 4. Phase II of the proposed stack generalization model.

The sigmoid function is one of them; h_{t-1} represents the hidden state of the $t-1$ instant; *bb* represents the bias of each gate; i_t, f_t, o_t, and C_t are the input gate, forget gate, output gate, and unit status, respectively. For the connection, W_f, W_i, and W_o are represented as a weight matrix. The three gates of LSTM cells govern the flow of information and hence define the cell's state. The gradient vanishing problem may be efficiently handled with LSTM [55].

The suggested model, in this instance, comprises two LSTM layers. The first LSTM layer outputs a sequence as one input above the LSTM layer. As explained previously, the internal design of both LSTM layers is the same. It also tried the LSTM cell rather than another GRU cell because the network with the LSTM cell outperformed the network with the GRU cell. This study constructs an LSTM network with a hidden vector of 128 elements. After the first LSTM layer, a dropout layer is added. Dropout reduces overfitting and enhances the model's generalization [56]. The LSTM's last layer generates a vector hi, which is supplied as the input to a fully linked multilayer network. Each layer has an activation function. The rectified linear unit (ReLU) activation function is used for each layer, and the exponential activation function is used for the output layer. Because the dataset is binary, a nonlinear activation function was used to solve the binary classification issue. For hidden layers of neurons, the ReLU function was employed, while for the output layer of neurons, the sigmoid function was used.

After the training process, the parameters were changed or tweaked to assess the wrong predictions and ensure the predictions are as correct as possible with optimization. We used an optimizer mold and designed the model for the most accurate and possible prediction with the parameters (or weights). The value that the weights were updated in the training process is called the learning rate, a configurable hyperparameter to train deep neural networks with a small value within the 0.0–1.0 range. However, the learning rate varies due to overfitting [53]; thus, our model can predict accurately with the given dataset. Nevertheless, it is not appropriate for new or real-world data. We used the regularization technique to overcome the overfitting errors by fitting the functions appropriately on the training sets. It helped to attain optimal optimization solutions. These optimizers modify the neural network's attributes, i.e., weights and learning rates, to improve the accuracy.

Thus, we utilized the following five adaptive optimizers to generalize the LSTM networks to overcome the overall loss and improve the accuracy. The selection of these optimizers is also given below:

- **AdaDelta:** This optimizer is based on the learning rate per dimension instead of the learning rate by parameter. It can solve the continual decay of learning rates by training and based on manually selected learning rates.
- **Adam:** This utilizes the prediction of the first and second moments to adapt the learning rate for the neural networks. It uses the momentum concept for adding a part of previous gradients to the current one. It is a faster optimizer and requires fewer parameters for tuning.
- **RMSprop:** Root means square propagation optimizer avoids the oscillations in the vertical direction and can increase the learning rate with feasible steps in the horizontal direction.
- **AdaGard:** This deals explicitly with individual features for different learning rates for different weights of sparse datasets to achieve a high learning rate. It can avoid the manual tuning of the learning rate for individual features.
- **SGD (Stochastic Gradient Descent):** Gradient descent optimizer has a drawback for large datasets. A variant of gradient descent, SGD, is generalized to make neural networks learn faster on the large-scale dataset.

These optimizers were implemented based on the packages and function calls in the Pytorch framework. For instance, we utilized *torch.optim.x*, where *x* indicates the name of the optimizer, i.e., Adam or SGD, etc. The model was then compiled using these adaptive optimizers. The model was trained to avoid overfitting by utilizing several epochs and early stopping strategies. By assessing the model using the test set, the output is now accessible. The stack generalization technique was used in the dataset after the strategy was implemented.

3.4. Final Prediction

Two outputs were generated using the aforementioned multilayer stacked methods, and a model was chosen depending on the decision based on the value of the initial

predictions. The mean prediction was considered and combined with the anticipated outcomes from the premier prediction. Finally, the outputs of the mean and premier prediction of the stacking models were combined as the final prediction using a meta-estimator classifier.

The meta-estimator involves constructing a robust classifier by applying the boosting method. Boosting combines multiple weak yet precise classifiers to create a powerful and resilient classifier for identifying phishing crimes. Additionally, boosting aids in integrating multiple features, resulting in improved classification performance. One notable boosting classifier is the XGBoost classifier, which transforms weak learners into potent contributors. It is well suited for our proposed stack generalization model for identifying phishing sites, introducing a sense of symmetry to the classification process. Implemented on integrated feature sets of URLs and character-level features, it acts as a robust classifier within our proposed AntiPhishStack model for phishing identification, emphasizing the importance of symmetry in enhancing detection capabilities.

Suppose there are n URLs in a set $\{(a_i, b_i)|i = 1, 2, \ldots, n\}$, where $a_i \in E^f$; represents a set of selected features corresponding to i-th URLs, while $b_i \in \{0, 1\}$ is a class label, e.g., $b_i = 1$ if the URLs are considered malicious or phishing websites. The final outcome of the XGBoost model was computed using the following equation [57]:

$$f_m(x) = f_l(b_i, f_m(a)) = \sum_{i=1}^{n} f_l(b_i, f_{m-1}(a_i) + G_m(a_i) + \Omega(G_m(a_i))) \tag{11}$$

where $f_m(x)$ is the model's prediction at step m, f_l represents the training loss function, and a represents the input features used in the XGBoost model. The regularization term $\Omega(G_m(a))$ is defined as $\gamma T + \frac{1}{2}\sum_{t=1}^{T} \omega_t^2$, where T is the number of leaf nodes in the base learner $G_m(a)$, γ is the complexity of each leaf, and λ represents the regularization parameter, controlling the strength of regularization in the XGBoost model, while ω_t is the output value at each final leaf node.

At step m, considering the base learners from previous steps $(m - 1)$ as fixed, the loss function can be expanded using Taylor's series [57,58]:

$$f_l(b, f_{m-1}(a) + G_m(a)) = \sum_{i=1}^{n} \left[g_i G_m(a_i) + \frac{1}{2} h_i G_m^2(a_i) \right] + \gamma T + \frac{1}{2} \sum_{t=1}^{T} \omega_t^2 \tag{12}$$

where g_i and h_i are the first and second derivatives of the loss function f_1 with respect to $f_{m-1}(a)$, computed as:

$$g_i = \frac{\partial f_l(b_i, f_{m-1}(a_i))}{\partial f_{m-1}(a)}$$

$$h_i = \frac{\partial^2 f_l(b_i, f_{m-1}(a_i))}{\partial f_{m-1}^2(a)}$$

This formulation defines the model's optimization process at each step, incorporating both the loss function and the regularization term to balance model complexity and fit. Then, the integrated features are categorized into phishing and benign, based on the weights by the meta-estimator for final prediction. Furthermore, XGBoost comes up with many advantages, some of which include (i) the power to fix missing values within the training set and, (ii) working with extensive data that do not fit into memory, and (iii) utilizing multiple cores on the CPU to achieve faster computing, .

Deep learning involves many datasets and a significant time for model training. The efficiency of these models depends on the system resource specifications and the complexity of datasets. In order to identify phishing assaults, the time complexity is a crucial factor [43]. The proposed method's computational cost is based on how the characteristics are generated and extracted. URL and character-level features extracted by our proposed method require logarithmic time complexity $O(\log(n))$. The extraction of

such features during the model training and time complexity depends on the number of samples n and dimensions d. Accordingly, the time complexity of our proposed work is $O(n\log(n)d)$.

4. Experiments

This paper utilized Python 2.7 to develop the suggested model and TensorFlow GPU v1.8.0 as a machine learning framework. The operating system was Windows 10 Pro Education, and the architecture was built using Python. This project's Python packages and libraries to detect phishing URLs included *Keras* (built-in with *TensorFlow*), *SciPy*, *Pandas*, *NumPy*, *Matplotlib*, and *Seaborn*.

Support Vector Machine (SVM), Decision Tree (DT), Naive Bayes (GNB), Logistic Regression (LR), Sequential Minimal Optimization (SMO), and K-Nearest Neighbor (KNN) algorithms were evaluated for stacking in this work. In the first stage, LSTM was employed as the basic classifier for stacked generalization, and further 10-fold cross-validation was utilized. In Phase II, the XGBoost classifier was utilized as a meta-estimator for the final prediction.

For the model's effectiveness, the following statistical metrics were used to assess the proposed work for different purposes [59].

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$Precision = \frac{TP}{(TP + FP)}$$

$$Recall = \frac{(TP)}{(TP + FN)}$$

$$F\_measure = \frac{2 * precision * recall}{(precision + recall)}$$

Precision–Recall Curve: A graph was utilized for the trade-off between the true positive rate and the true negative or vice versa for the predictive model assessment [59].

- For Positive Precision $P = \frac{TP}{TP+FP}$
- For Negative Precision $N = \frac{TN}{TN+FN}$
- For Positive Recall $PR = \frac{TP}{TP+FN}$
- For Negative Recall $NR = \frac{TN}{TN+FP}$

where TP indicates the true positive, which means the number of URLs is correctly classified as phishing; in contrast, the parameter TN indicates the true negative, which means the number of URLs is correctly determined as benign. FP is a false positive, which means the number of benign URLs is wrongly classified as phishing, and FN is a false negative, which shows the number of phishing URLs classified as benign.

Mean Absolute Error (MAE): The average value for all absolute errors [60].

$$MAE = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n} \tag{13}$$

Mean Square Error (MSE): The average value for all squared errors [60].

$$MSE = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{14}$$

5. Results and Evaluations

5.1. Feature Evaluation with Classifiers

This experiment evaluates both feature sets (URLF and CLF) from DS1 and DS2 by applying different machine learning classifiers. The fundamental goal of this experiment is

to determine the best classifiers for both features in Phase 1. The future evaluations of these classifiers are given in Figure 5 and Table 3. Imbalanced classes exist in real-time datasets, which creates the problem of classification. To tackle this problem, the F-Measure was utilized due to the crucial values of false negatives and false positives. We refrained from using oversampling techniques such as the Synthetic Minority Over-sampling Technique (SMOTE) due to the potential risk of overfitting.

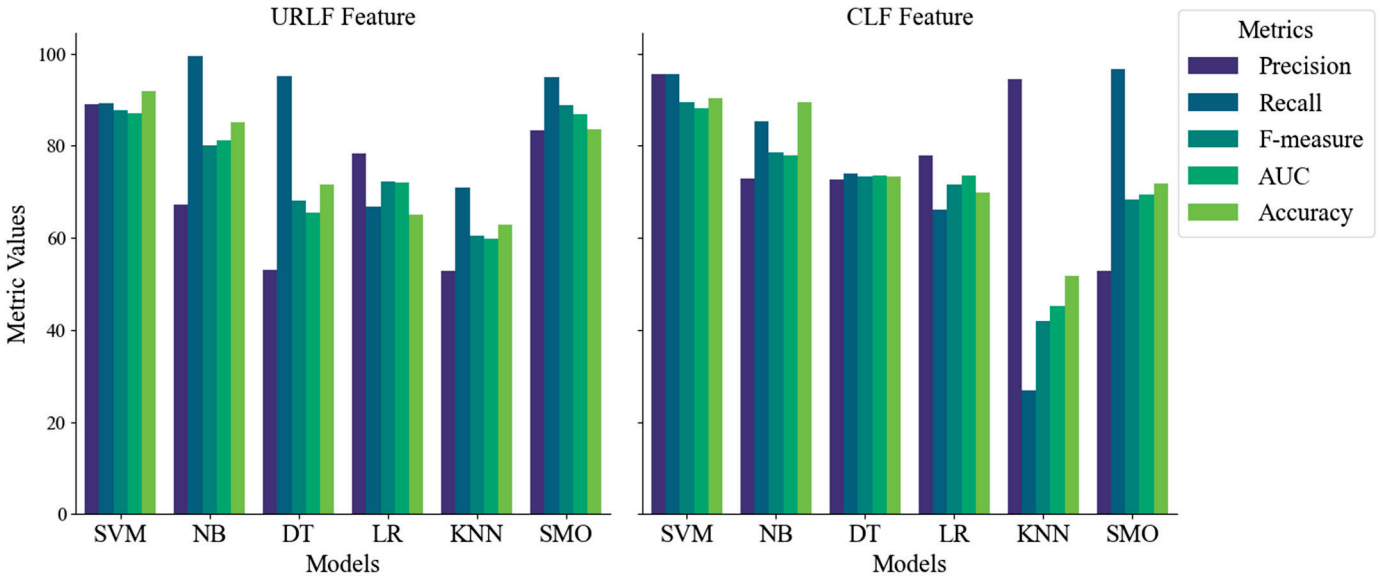


Figure 5. Classification results for proposed model on DS1.

Table 3. Classification results for proposed model on DS2.

Models	Features	Precision	Recall	F-Measure	AUC	Accuracy
SVM	URLF	88.72	98.62	93.4	91.34	92.46
	CLF	91.11	96.85	93.88	89.99	91.72
NB	URLF	86.26	98.18	91.84	91.07	90.08
	CLF	86.03	97.96	91.61	78.68	86.37
DT	URLF	74.58	73.66	74.12	71.89	71.23
	CLF	50.47	99.84	67.05	69.41	73.68
LR	URLF	80.61	99.92	89.23	85.37	76.24
	CLF	69.62	81.25	74.98	74.02	71.82
KNN	URLF	78.24	66.93	72.14	70.56	70.34
	CLF	78.38	66.86	72.16	71.21	68.91
SMO	URLF	86.03	97.21	91.28	91.4	89.64
	CLF	78.42	98.68	87.39	87.24	83.21

Figure 5 reveals a sense of symmetry, showcasing that SVM and NB exhibit maximum AUC, accuracy, precision, recall, and F-measure in DS1. These results show that SVM maintains an average accuracy of 91.1% and an 88.6% F-measure, while NB achieves an average accuracy of 87.295% and a 79.32% F-measure. However, some classifiers experience minimal accuracy. For instance, KNN records an average accuracy of 57.3% and a 51.25% F-measure, while LR demonstrates a 67.46% accuracy and a 71.87% F-measure. The observed lower accuracy is primarily attributed to the independent protectors employed by these classifiers. Beyond the classifiers, the URLF feature in SVM symmetrically stands out with the highest accuracy, reaching 91.78%.

Table 3 shows that the same classifiers have maximum accuracy to different extents in DS2. The results show SVM has an average of 92.09% accuracy and a 93.64% F-measure, while NB averaged an 88.225% accuracy and a 91.725% F-measure. Meanwhile, KNN and LR were found to have the lowest accuracy and F-measures. These lowest accuracy rates are due to the normal future distribution that is assumed by those classifiers.

As a comparative analysis of these features from both datasets (DS1 and DS2), Figure 6 presents the bar graph, indicating the accuracy of the comparison of all machine learning used in this work. It can be easily distinguished that the classifier performances on DS2 are slightly better than on DS1 for the given features. However, some classifiers have minimum accuracy due to those features that are inefficient enough to discriminate between phishing and benign features. It might be possible that phishers are using modern technologies to receive online users and websites.

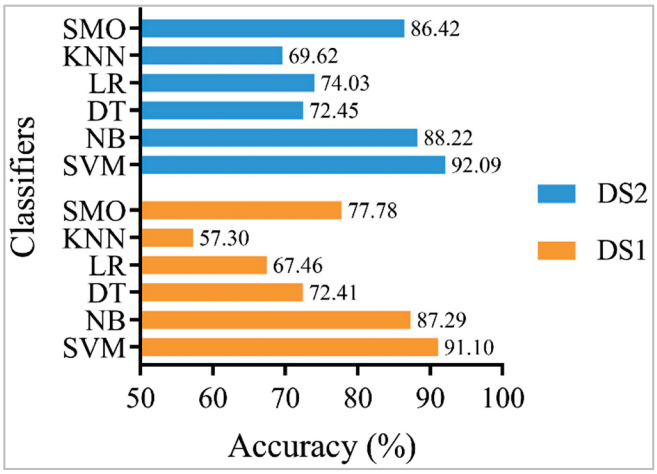


Figure 6. Comparison of machine learning classifier's performance for the proposed model on both datasets.

5.2. Optimizer Evaluation on LSTM

Optimizers deal with model accuracy, which is a key component of machine learning and artificial intelligence, and it is responsible for molding the model to acquire possible accurate results. In this experiment, different levels of epochs used in machine learning and artificial intelligence are considered to indicate the number of passes of the entries to train the dataset. The different numbers of epochs were adjusted to implement the two-layered LSTM with different optimizers. The sometimes-higher number of epochs can lead to overfitting issues, and a lower number of epochs may result in underfitting the model. The learning rate controls the speed at which the mold learns. It is a configurable hyperparameter to train the neural networks with a small positive value from 0.0 to 1.0. From the previous experiment, it was detected that features from D2 have maximum accuracy and F-measure values. Therefore, the optimizers on both datasets were evaluated to confirm the result's originality with the LSTM-based stack generalization model. Meanwhile, it used different deep learning-based adaptive optimizer programs from which the optimizer would be the best choice for the proposed anti-phishing model.

Table 4 shows that the AdaGard optimizer provider has the highest accuracy of 92.5%, a minimum mean squared error (MSE), and a mean absolute error (MAE) of 0.02 with CLF features in DS1. However, its learning rate is higher than other optimizers. Meanwhile, the performance of AdaDelta and the SGD optimizer is also significant, with the lowest learning rates of 0.0023 and 0.003, respectively. In contrast, the LSTM-based stack generalization model loses performance when predicting phishing features with other optimizers, such as Adam and RMSprop optimizers. The Adam optimizer has an average of an 89.6% accuracy and an RMSprop 89.7 at both feature sets.

Table 4. Optimizer evaluations with LSTM-based stack generalization for DS1.

Optimizer	Epochs	Features	Learning Rate	MSE	MAE	Accuracy
AdaDelta	200	URLF	0.019	0.04	0.06	91.6
		CLF	0.0023	0.08	0.05	91.1
Adam	100	URLF	0.007	0.09	0.07	89.3
		CLF	0.0016	0.07	0.08	89.9
RMSprop	150	URLF	0.025	0.03	0.09	89.7
		CLF	0.027	0.03	0.02	89.7
AdaGard	200	URLF	0.0048	0.06	0.03	91.8
		CLF	0.098	0.02	0.02	92.5
SGD	250	URLF	0.003	0.07	0.09	90.4
		CLF	0.003	0.05	0.06	90.2

In a symmetrical comparison, Table 5 reveals that the Adam optimizer achieved a maximum accuracy of 92.7%, MSE of 0.08, and MAE of 0.09 with CLF features in DS2. It is noteworthy that the learning rate of the Adam optimizer is 0.0194. Simultaneously, AdaDelta, AdaGard, and SGD optimizers exhibit highly sufficient accuracy, MSE, and MAE while maintaining a minimal learning rate. However, the RMSprop's performance is deficient, experiencing slightly lower accuracy with the proposed model, LSTM-based stack generalization. This decline stems mainly from the poor adaptive quality of those optimizers within the proposed model.

Table 5. Optimizer evaluations with LSTM-based stack generalization for DS2.

Optimizer	Epochs	Features	Learning Rate	MSE	MAE	Accuracy
AdaDelta	200	URLF	0.0029	0.03	0.04	91
		CLF	0.0017	0.06	0.05	90.6
Adam	100	URLF	0.096	0.07	0.08	91.3
		CLF	0.0194	0.08	0.09	92.7
RMSprop	150	URLF	0.056	0.06	0.06	90.1
		CLF	0.007	0.02	0.08	89.6
AdaGard	200	URLF	0.068	0.05	0.05	91.5
		CLF	0.007	0.01	0.03	90.8
SGD	250	URLF	0.001	0.06	0.04	91.2
		CLF	0.02	0.08	0.08	90.9

Furthermore, precision–recall curves are illustrated in Figure 7a,b for each future. These curves indicate the trade-off between precision and recall. A big area under the curve shows high precision and recall; high precision indicates a low false-positive rate, and high recall indicates a low false-negative rate. The analysis given in Tables 4 and 5 represents the learning rate that significantly contributes to the success of this proposed model with the adoptive optimizers. For example, the Adam optimizer has a maximum accuracy of 92.7, 0.08 MSE, and 0.09 MAE with CLF features for only a 0.0194 learning rate when the two-layered LSTM is employed with 100 epochs. For comparing optimizer evaluation with LSTM-based stack generalization on both datasets, the average performance on DS1 is a 90.62 accuracy, while a 91.24 accuracy is reported on DS2.

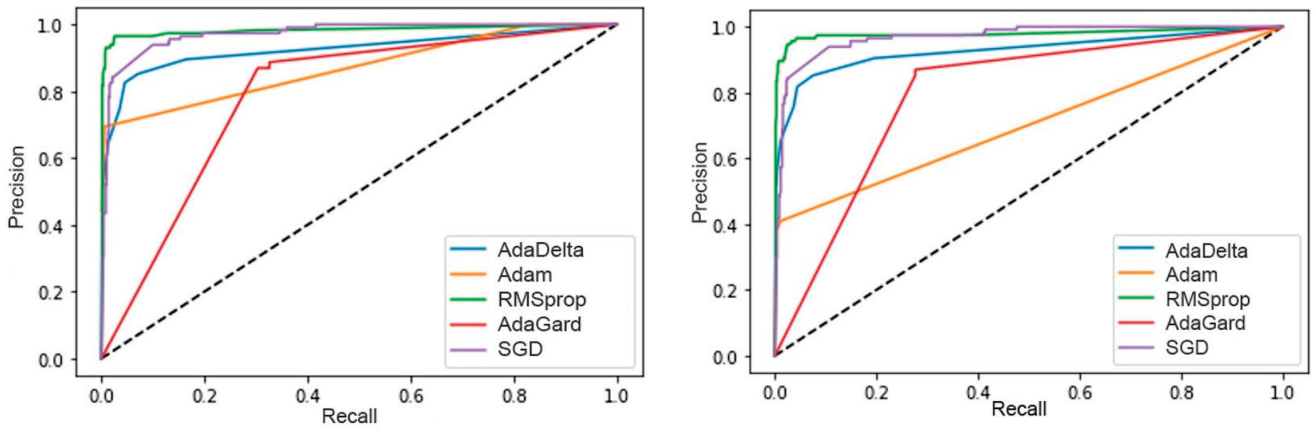


Figure 7. LSTM-based stack generalization model's precision–recall curve on DS1 with (**a**) URLF and (**b**) CLF.

In Figure 7a, the RMSprop optimizer has the highest precision–recall curve, while AdaGard has the lowest precision–recall curve with the URLF feature from DS1. Similarly, in Figure 7b, the RMSprop also has the highest precision–recall curve, while the Adam optimizer has the lowest precision–recall curve with the CLF feature from DS1. In Figure 8, almost all optimizers have an equivalent precision–recall curve with URLF and CLF features from DS2. However, there is a slight difference between the performances of both future sets with the proposed model, LSTM-based stack generalization.

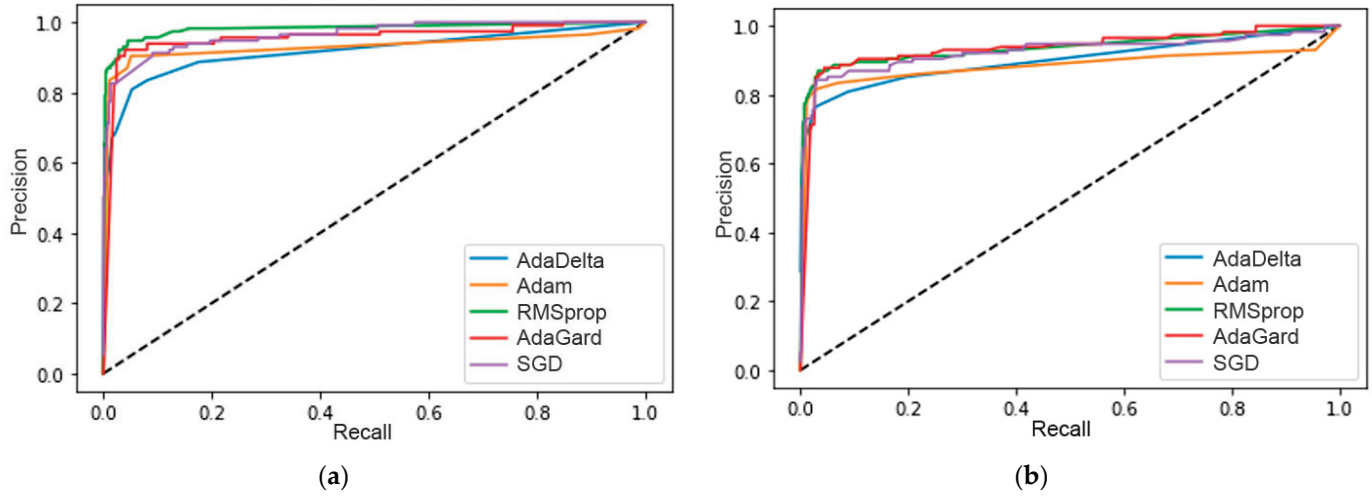


Figure 8. LSTM-based stack generalization model's precision–recall curve on DS2 with (**a**) URLF and (**b**) CLF.

5.3. AntiPhishStack Model's Evaluation

The major goal of stacked generalization is to employ a next-generation-based model that combines the previous models to attain higher prediction-based accuracy. Generally, the stacking method merges multiple models and learns them together for effective classification accuracy.

Initially, seven machine learning algorithms and one neural network were harnessed on the given datasets to ensure the highest accuracy on both future sets. These algorithms played a pivotal role in crafting the proposed stack model. Furthermore, five widely recognized optimizers were embraced to assess the LSTM-based stack generalization model, validating the symmetry between machine learning classifiers and a neural

network. Consequently, the proposed model symmetrically classified and deduced phishing and benign features from both datasets. The classifiers and neural network-based optimization have already acquired symmetry in the stacking process. The AntiPhishStack model was constructed by amalgamating features from both Phase I and Phase II. Accordingly, the previous temporary prediction underwent symmetrical filtering with a meta-estimator, XGBoost classifier, and LSTM model with its two hidden layers. Finally, the AntiPhishStack model deploys symmetrical detection criteria to distinguish the phishing and benign features.

For the performance evaluation of the AntiPhishStack model on given datasets, Table 6 presents the comprehensive measurements of the proposed model. The model training time was required to detect the features from the training dataset. The validation period presents the time required to classify each feature on the test datasets after the training is finished. The Keras callback approach was used to note the model training and testing times. This approach can save the model performance time when accuracy is no longer improving, and it interrupts the model on that particular task.

Table 6. Prediction of AntiPhishStack model with combined features on both datasets.

Dataset	Training Time (s)	Test Time (s)	MAE	MSE	Precision	Recall	F-Measure	AUC	Accuracy
DS1	8924.027	43.16	0.9	0.7	97.99	92.24	95.03	96.22	95.67
DS2	9706.15	57.23	0.5	0.4	98.01	92.3	95.91	95.81	**96.04**

It can be found that the AntiPhishStack model performed effectively on DS2 with a 96.04% accuracy compared to DS1, which has slightly less accuracy. Similarly, DS2 has a better F-measure score and MAE and MSE rates. However, the training time for both datasets was unexpectedly higher. For example, this model provided phishing detection in 161 min for DS2 and 148 min for DS1. Although in stacking models the training time is always higher than in basic or hybrid models, this timing is higher in this case due to two-layered LSTM and multiple stacks in both phases. Notably, in phases I and II, the average accuracy and F-measure values were less to a different extent than the final prediction with the AntiPhishStack generalization model. Furthermore, the training and validation accuracy rates were provided with individual feature sets of each dataset. Figure 9 presents the AntiPhishStack generalization model's training and validation accuracy rates for DS1, and Figure 10 indicates DS2.

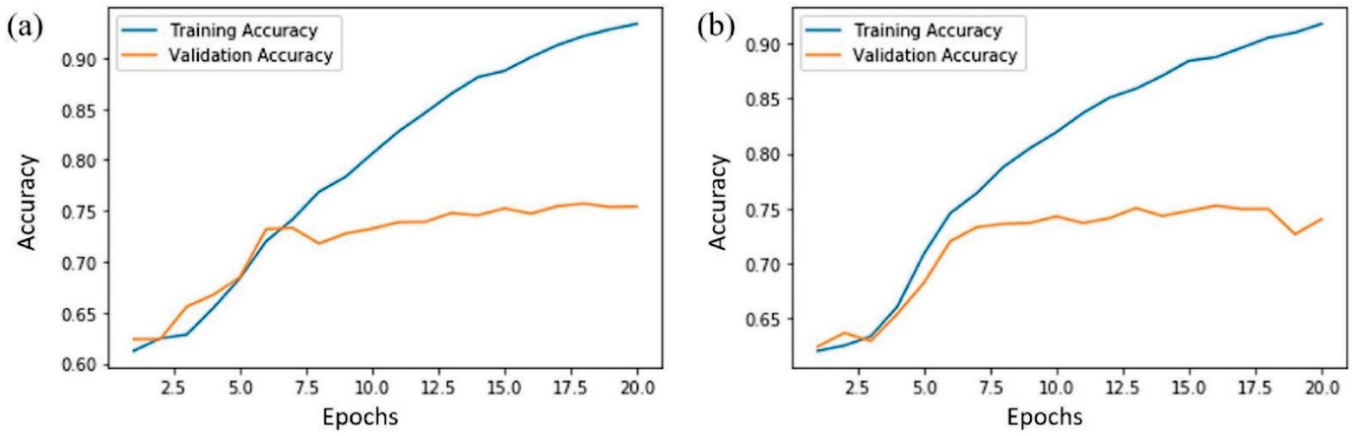


Figure 9. Training and validation accuracy of the AntiPhishStack model on DS1 with (**a**) URLF and (**b**) CLF.

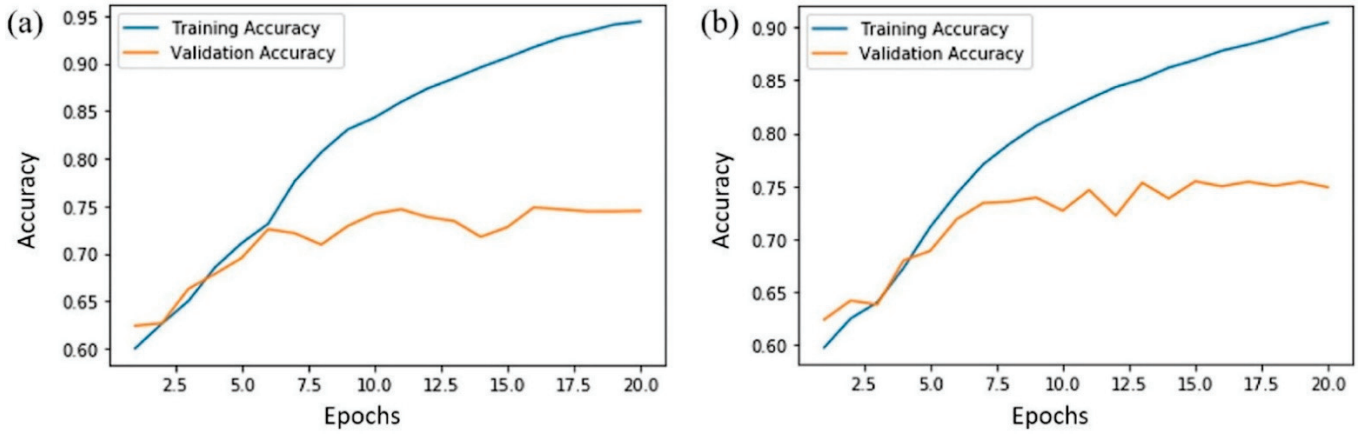


Figure 10. Training and validation accuracy of the AntiPhishStack model on DS2 with (**a**) URLF and (**b**) CLF.

5.4. Comparative Analysis

The significance of our proposed model was described and illustrated in previous subsections. Furthermore, we compared our model's performance with prior studies. Table 7 demonstrates the comparative analysis of existing studies by considering the applied approaches and algorithm WRT years.

Table 7. The proposed model's performance compared with the prior studies.

References	Year	Applied Approaches	Dataset	Algorithms/Networks	Observed Accuracy (%)
[26]	2018	Web-based phishing using URLs features	UCI phishing dataset	Random forest	92.9
[21]	2019	BiLSTM for URL feature extraction and CNN for classification	Alexa and PhishTank	R-CNN	95.6
[46]	2020	Hybrid multimodel solution for networks	UNSW NB-15, UGR'16	Ensemble four ML	92.9
[42]	2021	Integrated phishing URLs detection with CNN	PhishTank and Alex	CNN + XGB	89.31
Proposed model	2024	Stack generalized model for URLs and TF-IDF features.	Alexa, Yandex, PhishTank	LSTM and XGBoost	**96.04**

It should be noted that there is some existing literature on the topic with a higher performance than our proposed work; however, we only compared those works that were closely based on the stack generalization method, and these studies were tested in our same environment to ensure a fair comparison. The proposed model's performance outperformed the available studies to a different extent. For example, the current study [42] in 2021 proposed a generalized stack model for phishing detection by integrating URL features with CNN and XGB algorithms, and it reported an 89.31% accuracy. In contrast, our proposed AntiPhishStack model received a 96.04% accuracy by generalizing the URLs and character-level TF-IDF features.

5.5. Limitation of the Proposed Model

While our suggested method exhibits adequate accuracy, it does entail certain drawbacks. The first limitation is the asymmetry in our phishing-detection method's dependence on English-language textual properties. In cases where the suspected webpage employs a language other than English, inaccurate classification findings may result. This study high-

lights the best-case performance of our method on multiple datasets, acknowledging the limitation of not providing an average performance metric across various conditions. Future research should incorporate a more comprehensive evaluation with rigorous average performance metrics.

Additionally, our methodology overlooks assessing the website's URL status, whether active or not, affecting overall outcomes. To surmount this limitation, expediting the training process and refining feature engineering becomes essential. Addressing zero-day assaults swiftly is imperative due to the narrow timeframe of phishing attacks. A pragmatic phishing-detection system necessitates the inclusion of real-time detection as a crucial component. Capitalizing on big data technologies such as Apache Spark or Hadoop, which facilitate real-time processing, can reduce time complexity [61].

6. Conclusions

Phishing presents a symmetrical cybersecurity challenge. While machine learning techniques have played a pivotal role in phishing-detection systems, recent advancements in deep learning yield significant outcomes for addressing contemporary phishing issues, especially with malicious URLs. This paper introduces novelty through a two-layered deep neural network (LSTM) integrated into a proposed stack generalization model that eliminates the need for prior feature knowledge in phishing detection. The model undergoes direct training on processed features, optimizing phishing URL detection through two phases. The results from DS1 indicate SVM's average accuracy at 91.1%, slightly improving to 92.09% in DS2. Furthermore, the AdaGard optimizer exhibits peak accuracy at 92.5%, with minimum MSE and MAE of 0.02 for CLF features in DS1. Comparisons with existing baseline models and traditional ML algorithms underscore the importance of the proposed model with its symmetrical stack generalization technique. Experiments demonstrate the AntiPhishStack model achieving 95.67% and 96.04% accuracy rates for the final prediction on benchmarks DS1 and DS2, respectively. These results signify the model's intelligent detection of previously unidentified phishing URLs, positioning this paper as a symmetrical and advanced solution for establishing a phishing-detection mechanism through profound deep learning-based stacking methods.

To detect malicious and fraudulent contract accounts, the AntiPhishStack model might be implemented with different deep neural networks, i.e., gated recurrent unit (GRU) networks with cryptocurrency-based features [11].

Author Contributions: S.A.: Data curation, Writing—original draft, Writing—review and editing, Methodology, and Software. H.A. and A.M.: Visualization, Writing—review and editing, and Methodology. H.C. and A.R.: Conceptualization, Supervision, Methodology, Formal analysis, and Validation. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by Shenzhen Polytechnic Research Fund No. 6023310010K.

Data Availability Statement: The research data used are from https://yandex.com.tr/dev/xml/ (accessed on 3 December 2023).

Acknowledgments: The authors would like to thank all the anonymous reviewers for their insightful comments and constructive suggestions that have obviously upgraded the quality of this manuscript.

Conflicts of Interest: Author Hafsa Aslam was employed by the company; Cybex IT Group, Faisalabad 38000, Pakistan. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Huang, Y.; Yang, Q.; Qin, J.; Wen, W. Phishing URL Detection via CNN and Attention-Based Hierarchical RNN. In Proceedings of the 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science and Engineering (TrustCom/BigDataSE), Rotorua, New Zealand, 5–8 August 2019.
2. Dhamija, R.; Tygar, J.D.; Hearst, M.A. Why Phishing Works. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Montréal, QC, Canada, 22–27 April 2006.

3. Miao, Q.; Liu, J.; Cao, Y.; Song, J. Malware detection using bilayer behavior abstraction and improved one-class support vector machines. *Int. J. Inf. Secur.* **2016**, *15*, 361–379. [CrossRef]
4. Rahman, S.S.M.M.; Gope, L.; Islam, T.; Alazab, M. IntAnti-Phish: An Intelligent Anti-Phishing Framework Using Backpropagation Neural Network. In *Machine Intelligence and Big Data Analytics for Cybersecurity Applications*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 217–230.
5. Abutair, H.Y.; Belghith, A. Using Case-Based Reasoning for Phishing Detection. *Procedia Comput. Sci.* **2017**, *109*, 281–288. [CrossRef]
6. Jeet, R.; Kumar, P.A.R. A survey on interest packet flooding attacks and its countermeasures in named data networking. *Int. J. Inf. Secur.* **2022**, *21*, 1163–1187. [CrossRef]
7. Pompon, R.; Walkowski, D.; Boddy, S.; Levin, M. 2018 Phishing and Fraud Report: Attack Speak during the Holidays. 2018. Available online: https://www.f5.com/labs/articles/threat-intelligence/2018-phishing-and-fraud-report--attacks-peak-during-the-holidays (accessed on 15 November 2023).
8. Oleg Viktorov, S.i.A.A.-S. Detecting Phishing Emails Using Machine Learning Techniques. Ph.D. Thesis, Middle East University, Amman, Jordan, 2017.
9. Microsoft Corporate Blogs, New Research Forecasts the Staggering Cost of Cybercrime. 18 March 2014. Available online: https://blogs.microsoft.com/on-the-issues/2014/03/18/new-research-forecasts-the-staggering-cost-of-cybercrime/ (accessed on 7 August 2023).
10. APWG, Phishing Activity Trends Reports, Phishing Attack Trends Report—1Q 2019. 15 May 2019. Available online: https://apwg.org/trendsreports/. (accessed on 7 August 2023).
11. Do, N.Q.; Selamat, A.; Krejcar, O.; Yokoi, T.; Fujita, H. Phishing Webpage Classification via Deep Learning-Based Algorithms: An Empirical Study. *Appl. Sci.* **2021**, *11*, 9210. [CrossRef]
12. Ozawa, S.; Ban, T.; Hashimoto, N.; Nakazato, J.; Shimamura, J. A study of IoT malware activities using association rule learning for darknet sensor data. *Int. J. Inf. Secur.* **2020**, *19*, 83–92. [CrossRef]
13. Mimura, M.; Ito, R. Applying NLP techniques to malware detection in a practical environment. *Int. J. Inf. Secur.* **2022**, *21*, 279–291. [CrossRef]
14. Cui, Q.; Jourdan, G.-V.; Bochmann, G.V.; Couturier, R.; Onut, I.-V. Tracking Phishing Attacks Over Time. In Proceedings of the 26th International Conference on World Wide Web 2017, Perth, Australia, 3–7 April 2017; International World Wide Web Conferences Steering Committee: Perth, Australia, 2017; pp. 667–676.
15. Shirazi, H.; Bezawada, B.; Ray, I. "Kn0w Thy Doma1n Name": Unbiased Phishing Detection Using Domain Name Based Features. In Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies, Indianapolis, IN, USA, 13–15 June 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 69–75.
16. Dong, Z.; Kapadia, A.; Blythe, J.; Camp, L.J. Beyond the lock icon: Real-time detection of phishing websites using public key certificates. In Proceedings of the 2015 APWG Symposium on Electronic Crime Research (eCrime), Barcelona, Spain, 26–29 May 2015.
17. Mohammad, R.M.A. An Ensemble Self-Structuring Neural Network Approach to Solving Classification Problems with Virtual Concept Drift and Its Application to Phishing Websites. Ph.D. Thesis, University of Huddersfield, Huddersfield, UK, 2016.
18. Woogue, P.D.P.; Pineda, G.A.A.; Maderazo, C.V. Automatic web page categorization using machine learning and educational-based corpus. *Int. J. Comput. Theory Eng.* **2017**, *9*, 427–432. [CrossRef]
19. Yang, P.; Zhao, G.; Zeng, P. Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning. *IEEE Access* **2019**, *7*, 15196–15209. [CrossRef]
20. Le, H.; Pham, Q.; Sahoo, D.; Hoi, S.C.H. URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection. *arXiv* **2018**, arXiv:1802.03162.
21. Wang, W.; Zhang, F.; Luo, X.; Zhang, S. PDRCNN: Precise Phishing Detection with Recurrent Convolutional Neural Networks. *Secur. Commun. Netw.* **2019**, *2019*, 2595794. [CrossRef]
22. Raghunath, K.M.K.; Kumar, V.V.; Venkatesan, M.; Singh, K.K.; Mahesh, T.R.; Singh, A. XGBoost Regression Classifier (XRC) Model for Cyber Attack Detection and Classification Using Inception V4. *J. Web Eng.* **2022**, *21*, 1295–1321. [CrossRef]
23. Zhang, W.; Ren, H.; Jiang, Q.; Zhang, K. Exploring Feature Extraction and ELM in Malware Detection for Android Devices. In Proceedings of the Advances in Neural Networks—ISNN 2015, Jeju, South Korea, 15–18 October 2015; Springer International Publishing: Cham, Switzerland, 2015.
24. Sahingoz, O.K.; Buber, E.; Demir, O.; Diri, B. Machine learning based phishing detection from URLs. *Expert Syst. Appl.* **2019**, *117*, 345–357. [CrossRef]
25. Rao, R.S.; Vaishnavi, T.; Pais, A.R. CatchPhish: Detection of phishing websites by inspecting URLs. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 813–825. [CrossRef]
26. Hutchinson, S.; Zhang, Z.; Liu, Q. Detecting Phishing Websites with Random Forest. In Proceedings of the Machine Learning and Intelligent Communications, Hangzhou, China, 6–8 July 2018; Springer International Publishing: Cham, Switzerland, 2018.
27. Adebowale, M.A.; Lwin, K.T.; Hossain, M.A. Deep Learning with Convolutional Neural Network and Long Short-Term Memory for Phishing Detection. In Proceedings of the 2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Island of Ulkulhas, Maldives, 26–28 August 2019.

28. Acquisti, A.; Adjerid, I.; Balebako, R.; Brandimarte, L.; Cranor, L.F.; Komanduri, S.; Leon, P.G.; Sadeh, N.; Schaub, F.; Sleeper, M.; et al. Nudges for Privacy and Security: Understanding and Assisting Users' Choices Online. *ACM Comput. Surv.* **2017**, *50*, 44. [CrossRef]
29. El-Alfy, E.-S.M. Detection of Phishing Websites Based on Probabilistic Neural Networks and K-Medoids Clustering. *Comput. J.* **2017**, *60*, 1745–1759. [CrossRef]
30. Jain, A.K.; Gupta, B.B. PHISH-SAFE: URL Features-Based Phishing Detection System Using Machine Learning. In *Cyber Security*; Springer: Singapore, 2018.
31. Tan, C.L.; Chiew, K.L.; Sze, S.N. Phishing Webpage Detection Using Weighted URL Tokens for Identity Keywords Retrieval. In *9th International Conference on Robotic, Vision, Signal Processing and Power Applications*; Springer: Singapore, 2017.
32. Aamir, M.; Zaidi, S.M.A. DDoS attack detection with feature engineering and machine learning: The framework and performance evaluation. *Int. J. Inf. Secur.* **2019**, *18*, 761–785. [CrossRef]
33. Ahmad, I.; Yao, C.; Li, L.; Chen, Y.; Liu, Z.; Ullah, I.; Shabaz, M.; Wang, X.; Huang, K.; Li, G.; et al. An efficient feature selection and explainable classification method for EEG-based epileptic seizure detection. *J. Inf. Secur. Appl.* **2024**, *80*, 103654. [CrossRef]
34. Rasool, A.; Tao, R.; Marjan, K.; Naveed, T. Twitter Sentiment Analysis: A Case Study for Apparel Brands. *J. Phys. Conf. Ser.* **2019**, *1176*, 022015. [CrossRef]
35. Wang, Z.; Wang, D. Recurrent deep stacking networks for supervised speech separation. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017.
36. Rahman, S.S.M.M.; Rafiq, F.B.; Toma, T.R.; Hossain, S.S.; Biplob, K.B.M.B. Performance Assessment of Multiple Machine Learning Classifiers for Detecting the Phishing URLs. In *Data Engineering and Communication Technology*; Springer: Singapore, 2020.
37. Moghimi, M.; Varjani, A.Y. New rule-based phishing detection method. *Expert Syst. Appl.* **2016**, *53*, 231–242. [CrossRef]
38. Hussain, M.; Cheng, C.; Xu, R.; Afzal, M. CNN-Fusion: An effective and lightweight phishing detection method based on multi-variant ConvNet. *Inf. Sci.* **2023**, *631*, 328–345. [CrossRef]
39. Javeed, D.; Gao, T.; Khan, M.T.; Ahmad, I. A Hybrid Deep Learning-Driven SDN Enabled Mechanism for Secure Communication in Internet of Things (IoT). *Sensors* **2021**, *21*, 4884. [CrossRef]
40. Kim, J.; Ban, Y.; Ko, E.; Cho, H.; Yi, J.H. MAPAS: A practical deep learning-based android malware detection system. *Int. J. Inf. Secur.* **2022**, *21*, 725–738. [CrossRef]
41. Yuan, H.; Yang, Z.; Chen, X.; Li, Y.; Liu, W. URL2Vec: URL Modeling with Character Embeddings for Fast and Accurate Phishing Website Detection. In Proceedings of the 2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom), Melbourne, Australia, 11–13 December 2018.
42. Yang, R.; Zheng, K.; Wu, B.; Wu, C.; Wang, X. Phishing Website Detection Based on Deep Convolutional Neural Network and Random Forest Ensemble Learning. *Sensors* **2021**, *21*, 8281. [CrossRef] [PubMed]
43. Rao, R.S.; Pais, A.R. Two level filtering mechanism to detect phishing sites using lightweight visual similarity approach. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 3853–3872. [CrossRef]
44. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
45. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
46. Rajagopal, S.; Kundapur, P.P.; Hareesha, K.S. A Stacking Ensemble for Network Intrusion Detection Using Heterogeneous Datasets. *Secur. Commun. Netw.* **2020**, *2020*, 4586875. [CrossRef]
47. Riyaz, S.; O'la, A.-L. A Modified Stacking Ensemble Machine Learning Algorithm Using Genetic Algorithms. In *Artificial Intelligence: Concepts, Methodologies, Tools, and Applications*; Information Resources Management Association, Ed.; IGI Global: Hershey, PA, USA, 2017; pp. 395–405.
48. Dhull, A.; Singh, A.; Singh, K.K. An intelligent technique for pattern-based clustering of continuous-valued datasets. *Clust. Comput.-J. Netw. Softw. Tools Appl.* **2022**, *25*, 3231–3248. [CrossRef]
49. Tang, Y.; Chen, Y.; Zhou, D. Measuring Uncertainty in the Negation Evidence for Multi-Source Information Fusion. *Entropy* **2022**, *24*, 1596. [CrossRef]
50. Kamyab, M.; Tao, R.; Mohammadi, M.H. Sentiment Analysis on Twitter: A text Mining Approach to the Afghanistan Status Reviews. In Proceedings of the 2018 International Conference on Artificial Intelligence and Virtual Reality, Taichung, Taiwan, 10–12 December 2018; Association for Computing Machinery: Nagoya, Japan, 2018; pp. 14–19.
51. Xiang, G.; Hong, J.; Rose, C.P.; Cranor, L. CANTINA+: A Feature-Rich Machine Learning Framework for Detecting Phishing Web Sites. *ACM Trans. Inf. Syst. Secur.* **2011**, *14*, 21. [CrossRef]
52. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; MIT Press: Montreal, QC, Canada, 2015; Volume 1, pp. 649–657.
53. Ahmad, I.; Wang, X.; Javeed, D.; Kumar, P.; Samuel, O.W.; Chen, S. A Hybrid Deep Learning Approach for Epileptic Seizure Detection in EEG signals. *IEEE J. Biomed. Health Inform.* **2023**, 1–12. [CrossRef]
54. Kamyab, M.; Liu, G.; Rasool, A.; Adjeisah, M. ACR-SA: Attention-based deep model through two-channel CNN and Bi-RNN for sentiment analysis. *Peerj Comput. Sci.* **2022**, *8*, e877. [CrossRef] [PubMed]
55. Aslam, S.; Rasool, A.; Wu, H.; Li, X. CEL: A Continual Learning Model for Disease Outbreak Prediction by Leveraging Domain Adaptation via Elastic Weight Consolidation. *arXiv* **2024**, arXiv:2401.08940.

56. Wang, Z.; Kim, S.; Joe, I. An Improved LSTM-Based Failure Classification Model for Financial Companies Using Natural Language Processing. *Appl. Sci.* **2023**, *13*, 7884. [CrossRef]
57. Available online: https://github.com/YC-Coder-Chen/Tree-Math/blob/master/XGboost.md (accessed on 15 November 2023).
58. Chen, T.Q.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, 13–17 August 2016.
59. Rasool, A.; Tao, R.; Kamyab, M.; Hayat, S. GAWA–A Feature Selection Method for Hybrid Sentiment Classification. *IEEE Access* **2020**, *8*, 191850–191861. [CrossRef]
60. Indrasiri, P.L.; Halgamuge, M.N.; Mohammad, A. Robust Ensemble Machine Learning Model for Filtering Phishing URLs: Expandable Random Gradient Stacked Voting Classifier (ERG-SVC). *IEEE Access* **2021**, *9*, 150142–150161. [CrossRef]
61. Haggag, M.; Tantawy, M.M.; El-Soudani, M.M.S. Implementing a Deep Learning Model for Intrusion Detection on Apache Spark Platform. *IEEE Access* **2020**, *8*, 163660–163672. [CrossRef]

symmetry

MDPI

Article

RG-Based Region Incrementing Visual Cryptography with Abilities of OR and XOR Decryption

Yu-Ru Lin and Justie Su-Tzu Juan *

Department of Computer Science and Information Engineering, National Chi Nan University, Puli, Nantou 545, Taiwan; s111321509@mail1.ncnu.edu.tw
* Correspondence: jsjuan@ncnu.edu.tw

Abstract: Visual cryptography (VC) is a cryptographic technique that allows the encryption of a secret image into multiple shares. When the shares of a qualified subset are superimposed, the original secret image can be visually recovered. Region incremental visual cryptography (RIVC) is a class of visual cryptography; it encrypts a single image into a shared image with multiple levels of secrecy, and when decrypted, the secret image of each region can be gradually recovered. Traditional VC encrypts two black-and-white images, and its recovery method is equivalent to a logical OR operation. To obtain a better recognizability of the restored image, the XORoperator becomes a simple and efficient method of encryption and decryption. Because the XOR operation needs extra cost or equipment, if the equipment cannot be obtained, the scheme can be more flexible if the secret can still be restored by using OR decryption (superimpose). In this paper, we propose a novel RIVC that allows encoding multiple secret regions of a secret image into n random grids. Both the OR operation and the XOR operation can be used as operations during decryption. The proposed scheme is evaluated by simulation, and the experimental result shows its correctness, effectiveness and practicability.

Keywords: visual cryptography; region incrementing; secret sharing; XOR operation; random grid

1. Introduction

Introduced in 1995 [1], the visual cryptography scheme (VCS for short) was a new technique that involves encrypting a secret image into n random grid images (also called "shares") that can be stacked together to disclose the original secret image. To enhance this approach, a (k, n)-threshold visual cryptography scheme ((k, n) VCS for short) was presented. This method entails the encryption of a binary secret image into n shares, with the condition that at least k shares (where $k \leq n$) must be combined to reconstruct the original image. If fewer than k shares are utilized, no clue about the original image can be discerned.

Generally speaking, cryptography systems in information security can be divided into symmetric cryptography systems and asymmetric cryptography systems. For VCS, if shares are regarded as the "key" and "ciphertext" generated by the encryption algorithm, then because at least k shares need to be collected to recover the "plaintext", the visual cipher can be regarded as a type of symmetric cryptosystem. Compared with traditional cryptography systems, VCS usually does not pursue the ability to completely restore the original secret. Instead, it focuses on the ability to use human vision to identify the original secret during restoration.

Looking back to 1987, Kafri and Keren [2] were pioneers in the field of visual secret sharing (VSS for short) through the use of random grids (RG for short). In the field of visual cryptography, random grids are a technique used to divide a secret image into irregular grids or patterns. Each grid or pattern is designed to appear random when viewed individually and does not contain the original secret information, as shown in Figure 1. The original secret image can only be reconstructed when these random grids or

patterns are combined or superimposed. This innovative approach rectified the limitation of secret pixel expansion and eliminated the necessity for codebook designs, which were shortcomings in Naor and Shamir's method [1]. Although Kafri and Keren [2] initially introduced the construction of (2, 2) RG-based VCS, it was not until 2009 that Shyu [3], as well as Chen and Tsao [4] independently proposed the first (n, n) RG-based VCS. In 2011, Chen and Tsao [5] were pioneers in introducing (k, n) RG-based VCS.

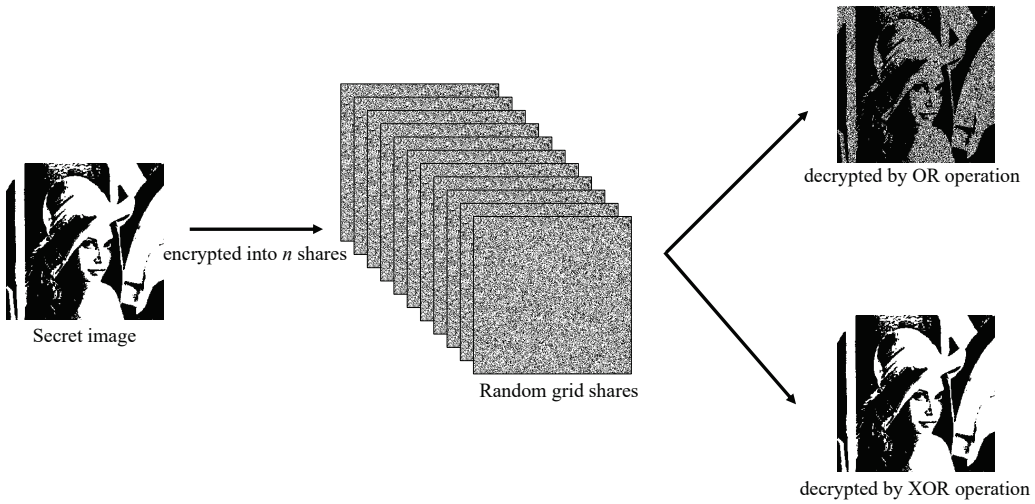


Figure 1. The structure of OR ($\otimes$) and XOR ($\oplus$) decryption.

Traditional VCS involves directly stacking two binary (black-and-white) images, which is equivalent to performing a logical OR operation just like the human visual system (HVS); we treat black pixels as 1 and white pixels as 0. In other words, it treats white pixels as transparent and black pixels as black. When the two images are superimposed, what the human eye perceives is the content of the OR-reconstructed image. In practice, it is equivalent to printing the share on a transparency separately. You can simply overlay those transparencies when restoring. However, this approach results in the image becoming progressively darker and makes it difficult to identify the original secret as more shares are stacked. Furthermore, it is impossible to completely restore the original secret. Therefore, a method for recovering the secret using an XOR operation was developed. By assigning 0 (white) when both pixels are the same and 1 (black) when they differ, as shown in Table 1, this operation makes it possible to fully recover the secret, as shown in Figure 1. In practice, it can be completed by using additional equipment to perform XOR operations. For example, using a copier that can print out black-and-white reverse images (inverting color, that is, a NOT operation logically) and a formula that converts the XOR operation into OR and NOT operations. Because the XOR operation needs extra cost or equipment, if the equipment cannot be obtained, the scheme can be more flexible if the secret can still be restored by using the OR operation (direct stacking). Currently, some scholars have studied a VCS that can restore the original image using both OR and XOR decryptions [6–8].

Table 1. Stacking result by OR ($\otimes$) and XOR ($\oplus$) calculation.

Pixel 1 p_1	**Pixel 2** p_2	$p_1 \otimes p_2$	$p_1 \oplus p_2$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	0

In 2009, Wang proposed a region incrementing visual cryptography scheme (RIVCS for short) [9], which enables a dealer to divide the content of a secret image S into multiple

regions and assign a level of secrecy to each region. In this q-level scheme, the secrecy level of a region ranges from 1 to q, with the first-level secret being the least significant and the q-level secret being the most significant. The dealer has the flexibility to assign each region of S a secrecy level that aligns with the specific needs of their application, reflecting the degree of secrecy required for that region. This level of regulation allows for more precise control over the distribution of secrets, ensuring that sensitive information remains highly protected while still allowing for the sharing of less sensitive information. Wang's scheme [9] utilizes the basic matrix proposed by Naor and Shamir [1] for encryption. However, the encryption process results in pixel expansion and can only be restored using an OR operation. For the purpose of addressing the limitations of pixel expansion and the reliance on a single OR-based decryption, we have made improvements accordingly and propose an RG-based RIVCS with the abilities of OR and XOR decryption. While visual cryptography is considered symmetric due to the division between public and private keys, it is important to note that VC, including RIVCS, often does not rely on complex mathematical operations or key pairs like traditional symmetric encryption (e.g., AES or DES). Instead, it uses visual properties and simple pixel stacking techniques for decryption. After Wang, many scholars have continued research on RIVCS. Wang et al. [10] proposed a (2, 3) RIVCS using random grids in 2010. In 2012, Shyu and Jiang [11] focused on minimizing pixel expansion of basis matrices (2, n) RIVCS, and Yang et al. [12] concentrated on applying (k, n) RIVCS to color images. Kumar and Sharma [13] proposed a (k, n) RIVCS using random grids in 2015. They gave three construction methods using the ideas of Kafri and Keren [2], and the number of the secrecy levels is always equal to $n - k + 1$. Furthermore, in 2019, Yang et al. [14] achieved progressive restoration in a (k, n) RIVCS.

Currently, many scholars are actively engaged in research on visual cryptography schemes. Each researcher brings a unique focus to their exploration of VCS. For example, Huang et al. emphasize generating meaningful encrypted images and utilizing the XOR operator for restoration [15]. Chen and Juan specialize in encrypting grayscale and color images into meaningful encrypted images [16]. Panchbhai and Varade focus on the research of a block-based progressive secret sharing scheme (BPVSS for short) for both grayscale and color images [17]. Zhang et al. concentrate on a group secret sharing scheme [18], where collaborative efforts within a group enable the recovery of secret information.

Therefore, the main contribution of this paper is to propose a new method for RIVCS. Compared with existing RIVCS, the proposed method not only has better visual quality in OR-based recovery in most cases (especially when stacking all shares), but also incorporates the function of XOR-based restoration. It supports OR and XOR operation decryption, making the decryption process flexible. This design allows users to use OR operations for decryption without the need for any specialized equipment, and users with additional equipment can use XOR operations to enhance the visual quality of the recovered images. To the best of our knowledge, this paper is the first to propose an innovative RIVCS that can use both OR and XOR operations for decryption.

The paper follows this structure: the subsequent section covers related work, Section 3 introduces the main result of this paper, which is (k, n) 2D_RIVCS, and Section 4 provides an analysis of this scheme, while Sections 5 and 6 showcase the experimental results and comparison with other similar works, respectively. Finally, Section 7 provides a summary and introduces some innovative ideas for future work.

2. Related Work

In this section, we will provide an overview of the relevant literature that has informed and influenced the research presented in this paper. These foundational references serve as the basis for the exploration conducted in this study. Section 2.1 will introduce the variables and definitions utilized in this paper. Section 2.2 will explore Kafri and Keren's pioneering work on (2, 2) RG-based VCS [2], a method grounded in random grids for encryption. Section 2.3 will present the (k, n) RG-based VCS proposed by Chen and Tsao in 2011 [5]. Section 2.4 will introduce Wang's innovative contribution to the field, known

as region incrementing visual cryptography (RIVCS) [9]. Lastly, Section 2.5 will provide insights into Lin and Juan's RG-based (k, n)-threshold visual cryptography with abilities of OR and XOR decryption (2D_RIVCS) [6]. These foundational studies lay the groundwork for our investigation and contribute to the understanding of the developments in visual cryptography.

2.1. Variables and Definitions

First, the variables used in this paper will be introduced, as shown in Table 2.

Table 2. Notations used in this paper.

Notation	Definition
0	White (transparent) pixel
1	Black (opaque) pixel
$\otimes$	OR operation
$\oplus$	XOR operation
S	The secret image
$S(0)/S(1)$	All of S's transparent/opaque pixels
s	A pixel of S
W, H	The width and height of the image
B	The encrypted image
b	The encrypted image before rearrangement
R	The reconstructed image
(i, j)	The position of the pixel in the image, $1 \leq i \leq W, 1 \leq j \leq H$
q	The number of secret levels
Sl	Secret level of secret image

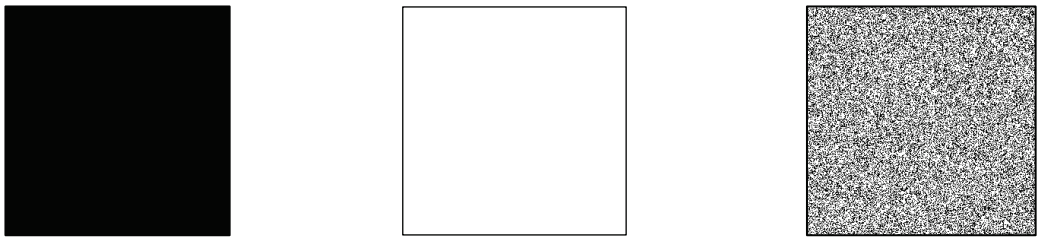
After defining the variables, the next step involves introducing some tools we will employ to assess the quality of the reconstructed images: average light transmission (called T) and contrast (called α). These tools are essential in evaluating the fidelity and visual clarity of the reconstructed images. Average light transmission allows us to compute how well the secret information is revealed, while contrast measures the sharpness and distinction between the secret content and the reconstructed image. These metrics play a crucial role in objectively quantifying the effectiveness of our approach in achieving high-quality reconstructions. Below are the definitions of average light transmission and contrast.

Definition 1. *Average Light Transmission (T) [19].*

The likelihood of a specific pixel, denoted as x, being transparent in the binary image X is represented as Prob(x = 0). This probability is indicated by the symbol $T[x]$, which expresses the light transmission of pixel x. The average light transmission of image X is expressed as

$$T[X] = \frac{\sum_{i=1}^{W} \sum_{j=1}^{H} T\left[x_{i,\ j}\right]}{W \times H}, \tag{1}$$

where $x_{i,\ j}$ is the pixel of X at the position (i, j). Therefore, when $T[X]$ is equal to 0, the entire image will consist of black pixels, as shown in Figure 2a. Conversely, when $T[X]$ is equal to 1, the entire image will be completely white, as shown in Figure 2b. When $T[X]$ is set to $\frac{1}{2}$, the entire image will appear as a half-black and half-white composition, as depicted in Figure 2c. Note that a white pixel is denoted by 0, but the light transmission of a white pixel $T[0] = 1$; a black pixel is denoted by 1, and the light transmission of a black pixel $T[1] = 0$.



(**a**) (**b**) (**c**)

Figure 2. Examples of different average light transmission (T) (**a**) Average light transmission is 0. (**b**) Average light transmission is 1. (**c**) Average light transmission is 0.5

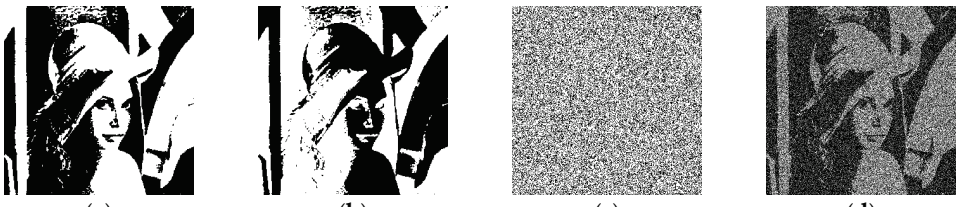
Definition 2. *Contrast (α) [19].*

Contrast serves as a critical criterion for assessing the fidelity of the reconstructed images and evaluating the effectiveness of the method proposed in this paper. The higher the contrast, the closer the visual resemblance between the reconstructed image and the original secret image. Contrast will be calculated using the previously mentioned average light transmission (T) and can be determined using the following formula:

$$\alpha = \frac{T[R[S(0)]] - T[R[S(1)]]}{1 + T[R[S(1)]]} \tag{2}$$

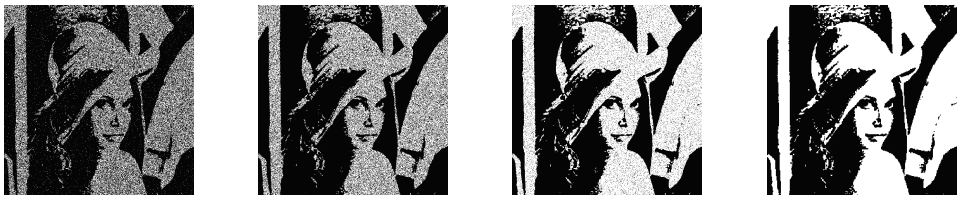
where $S(0)$ represents the white pixels in the secret image S, while $S(1)$ represents the black pixels in the same image. $T[R[S(0)]]$ signifies the average light transmission of the reconstructed image B corresponding to the white area of the secret image S, and $T[R[S(1)]]$ represents the average light transmission of the reconstructed image B corresponding to the black area of the secret image S. By utilizing the difference between $T[R[S(0)]]$ and $T[R[S(1)]]$ and dividing it by 1 + $T[R[S(1)]$, the contrast can be calculated. The contrast value falls within the range of -1 to 1.

When the contrast is -1, the secret image and the reconstructed image are exact opposites, with black and white colors inverted, as shown in Figure 3b. When the contrast is 0, the secret image and the reconstructed image lack any discernible correlation, and no traces of the secret image are visible in the reconstructed image, as shown in Figure 3c. When the contrast is greater than 0, details related to the secret will gradually become apparent. As the contrast approaches 1, the content of the secret becomes clearer, as shown in Figure 3d–g. Figure 3d–g are generated based on the contrast formula. Figure 3d,e fix the average light transmission of $S(0)$ at $\frac{1}{2}$ (as in Figure 3c), and the calculated average light transmissions for $S(1)$ are $\frac{1}{4}$ and $\frac{1}{14}$, respectively. In Figure 3f,g, the average light transmission of $S(1)$ is fixed at 0 (as in Figure 3h, because it must be greater than 1/2 and for ease of comparison with each other), and the calculated average light transmissions for $S(1)$ are $\frac{3}{5}$ and $\frac{4}{5}$, respectively. At a contrast value of 1, the secret image and the reconstructed image match perfectly, representing a perfect reconstruction, as shown in Figure 3h. This relationship is illustrated in Figure 3.



(**a**) (**b**) (**c**) (**d**)

Figure 3. *Cont.*



(**e**) (**f**) (**g**) (**h**)

Figure 3. Examples of different contrast (α) (**a**) Secret image S. (**b**) R with $\alpha = -1$. (**c**) R with $\alpha = 0$. (**d**) R with $\alpha = 0.2$. (**e**) R with $\alpha = 0.4$. (**f**) R with $\alpha = 0.6$. (**g**) R with $\alpha = 0.8$. (**h**) R with $\alpha = 1$.

Contrast (α) provides an objective measure of how distinct and well-defined the hidden information appears within the reconstructed image. It allows us to quantitatively gauge the quality of the restoration process and its success in revealing the concealed content.

Definition 3. *Security [20].*

After proposing a scheme, it is essential to validate the security and feasibility of the scheme. In threshold-based visual cryptography, achieving security conditions involves two critical aspects. Firstly, when provided with a single share (encrypted image), no information regarding the secret image should be discernible. Secondly, when collecting fewer than the threshold value k of shares, no details about the secret image should be revealed. Notably, based on the concept of contrast, an image exhibits a contrast of 0 with the secret image S when they are entirely unrelated. Hence, when introducing a novel approach, it is important to demonstrate that both individual shares and collections of fewer than k shares are secure, resulting in a contrast of 0. To achieve a contrast of 0, it is necessary to ensure that

$$T[R[S(0)]] = T[R[S(1)]], \tag{3}$$

as dictated by the contrast formula.

Security is predicated on the concept that examining an individual encrypted image (share) or even a subset of images falling short of the threshold k should yield no information about the secret image. Consequently, ensuring that the contrast is zero in these scenarios is a key indicator of security in threshold-based visual cryptography schemes.

Definition 4. *Visually recognizable [20].*

In addition to verifying its security, the visual recognizability of a method also requires validation. In the context of visual cryptography, usability refers to the ability to reveal information about the secret image S when a sufficient number of encrypted images are collected, specifically, when more than or equal to k images are gathered. The range of contrast, as defined, is between -1 and 1. A contrast of 0 indicates that the two images are entirely unrelated, while a contrast greater than 0 theoretically implies the potential to unveil information about the secret image S. Therefore, based on the contrast formula, it is necessary to achieve

$$T[R[S(0)]] > T[R[S(1)]], \tag{4}$$

to confirm usability.

This approach ensures that when a certain number of encrypted images, equivalent to or exceeding the threshold k, are collected, the contents related to the secret image S can be revealed. This is a fundamental aspect of visual recognizability in the context of visual cryptography. Contrast, a measure that ranges between -1 and 1, plays a crucial role in determining the degree to which the secret image's content can be observed. Achieving a contrast greater than 0 is essential for usability, suggesting the potential for discerning details about the secret image S.

2.2. (2, 2) RG-Based VCS

In 1987, Kafri and Keren [2] made a groundbreaking contribution to the field of visual cryptography by introducing the concept of (2, 2) RG-based VCS. Their innovative approach involved encrypting a binary image, denoted as S, and generating two shares, B_1 and B_2. These shares resemble seemingly random grids when viewed individually. However, when these shares were superimposed, they allowed the secret image S to be seamlessly reconstructed. What set their method apart was its simplicity and accessibility, as it required no specialized computational or cryptographic knowledge; it just needed to stack by OR operation ($\otimes$) like the human visual system (HVS). The Algorithm 1 they devised is depicted below:

Algorithm 1: (2, 2) RG-Based VCS [2]

Input:	A $W \times H$ secret binary image S.
Output:	Two shares B_1, B_2.
Step 1.	Generate the first share B_1 randomly selecting 0 or 1 for each pixel of B_1.
Step 2.	For each pixel $B_2[i, j]$ of share B_2, based on the pixel $S[i, j]$ of S and the pixel $B_1[i, j]$ of B_1, $B_2[i, j]$ is calculated by $B_2[i,j] = \begin{cases} B_1[i,j], & if\ S[i,j] = 0 \\ \overline{B_1[i,j]}, & if\ S[i,j] = 1 \end{cases}$.
Step 3.	Output (B_1, B_2).

The algorithm of (2, 2) RG-based VCS begins by inputting a binary secret image S with dimensions $M \times N$. The ultimate goal of this algorithm is to produce two random encrypted images, denoted as B_1 and B_2, which, when viewed individually reveal no discernible clues about the original secret content. The algorithm operates on a pixel-by-pixel basis and follows these key steps: first, it initiates the process by generating the content of B_1. In this step, each pixel is randomly assigned a value of 0 or 1, creating B_1 as a pattern that appears random and devoid of any identifiable structure; subsequently, we generate B_2, the content of which is determined based on the pixel values of S and B_1. When a pixel in S is 0, corresponding to a white region, B_2 mirrors the content of B_1. Conversely, when a pixel in S is 1, representing a black region, B_2 adopts the opposite value to that of B_1. Each pixel is processed according to the above steps, resulting in the generation of two encrypted images, B_1 and B_2.

These encrypted images form the basis of (2, 2) RG-based VCS, offering a secure and visually decipherable means of sharing confidential data without necessitating complex computations or cryptographic expertise. Kafri and Keren's pioneering work laid the foundation for subsequent advancements in visual cryptography, offering a user-friendly approach to protect and share sensitive information that remains relevant and influential to this day.

2.3. (k, n) RG-Based VCS

In 2009, Chen and Tsao conducted research on Kafri and Keren's (2, 2) RG-based VCS [2]. They expanded the original method, which could only encrypt into two encrypted images, to allow encryption into n encrypted images. These n encrypted images could then be directly superimposed using the OR operation ($\otimes$) to reconstruct the original secret, a scheme referred to as (n, n) RG-based VCS [4]. Two years later, in 2011, Chen and Tsao achieved another significant breakthrough by transforming (n, n) RG-based VCS into (k, n) RG-based VCS [5], introducing a threshold value k, where k must be less than or equal to n. This adaptation made (k, n) RG-based VCS more flexible. Unlike the previous method, which required the collection of all encrypted images for reconstruction, (k, n) RG-based VCS only necessitates gathering k encrypted images to restore the secret, offering users greater convenience. Below is the Algorithm 2 for (k, n) RG-based VCS.

Algorithm 2: (k, n) RG-Based VCS [5]

Input:	A $W \times H$ secret binary image S, a threshold value k and a total number n.
Output:	n shares ($B_1, B_2, \ldots, B_n$).
Step 1.	For every pixel $S[i, j] = s$, iterate through Steps 2 to 4.
Step 2.	Randomly select from {0, 1} to generate n random pixels $b_1, b_2, \ldots, b_n$.
Step 3.	$b_k = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1}$.
Step 4.	Randomly rearrange the above n bits $b_1, b_2, \ldots, b_n$ into $B_1[i, j], B_2[i, j], \ldots, B_n[i, j]$.
Step 5.	Output ($B_1, B_2, \ldots, B_n$).

In the algorithm for (k, n) RG-based VCS [5], we perform encryption on each pixel, employing the (2, 2) RG-based VCS method. Here is how it works: we take a single pixel s from the secret image and encrypt it into two pixels, b_1 and $\widetilde{b_2}$. We then treat $\widetilde{b_2}$ as the secret pixel and encrypt it to obtain b_2 and $\widetilde{b_3}$. We continue this process iteratively until $\widetilde{b_k}$ is generated, as illustrated in the flowchart in Figure 4. Subsequently, b_{k+1} to b_n are randomly assigned values of 0 or 1. At this point, we have obtained $b_1, b_2, \ldots, b_n$. From the flowchart in Figure 4, it is evident that b_1 to b_{k-1} and b_{k+1} to b_n are randomly generated, similar to Step 2 in the (k, n) RG-based VCS algorithm. Only b_k is computed and is the result of repeatedly applying the (2, 2) RG-based VCS process. Consequently, we can generalize b_k as the result of $s \oplus b_1 \oplus b_2 \ldots \oplus b_{k-1}$, just as in Step 3 of the (k, n) RG-based VCS algorithm. Once this process is completed for each pixel [i, j] of the secret image S, these n pixels are randomly distributed across $B_1, B_2, \ldots, B_n$, finalizing the encryption in (k, n) RG-based VCS.

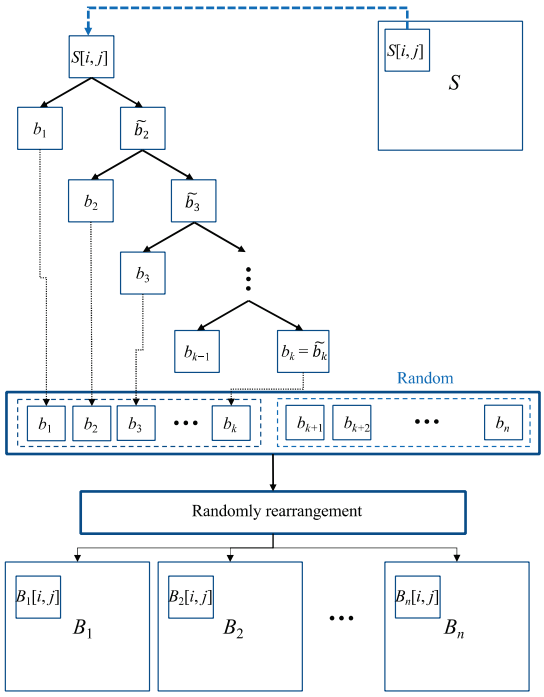


Figure 4. The flowchart of (k, n) RG-VCS.

The decryption process for (k, n) RG-based VCS is straightforward. To visually reconstruct the secret, one can directly stack k or more encrypted images together, achieving visual recovery through an OR operation ($\otimes$).

2.4. Region Incrementing Visual Cryptography (RIVCS)

Wang introduced a region incrementing visual cryptography scheme (RIVCS) in 2009 [9] that marked a significant advancement in the field. This innovative approach empowered the dealer to divide the content of the secret image S into multiple regions, referred to as secret levels. This paper represents the number of secret levels using the variable q. The (k, n)-threshold visual cryptography method further enhanced this concept,

enabling the segmentation of the secret image S into a maximum of $n - k + 1$ secret levels, denoted as Sl_1, Sl_2, ..., Sl_{n-k+1}. So, the range of q is from 1 to $n - k + 1$. These regions follow a hierarchical order where $R_i \subset R_{i+1}$, ensuring that each subsequent level contains increasingly significant secret information. This hierarchical structure allows for the flexible allocation of varying degrees of secrecy to different regions, ensuring that the most sensitive information remains well-protected while allowing for the sharing of less critical data.

The content of lower secret levels is nested within higher secret levels, as illustrated in Figure 5. Figure 5a,e represent the original secret images, Figure 5b–d depict the secret levels when $q = 3$ for Figure 5a, and Figure 5f–i represent the secret levels when $q = 4$ for Figure 5e. It is evident from the above explanation that when referring to the secret level Sl_q, the secret level Sl_q is equal to the secret image S.

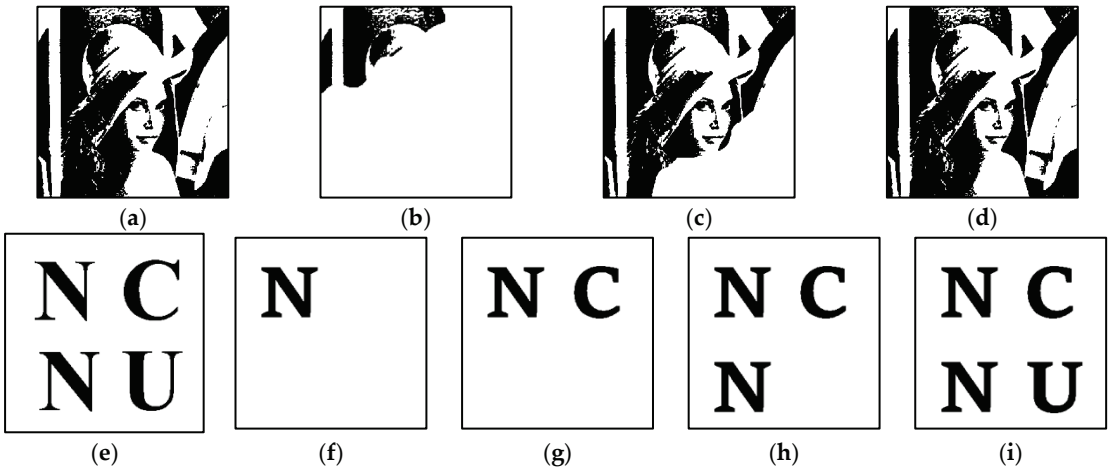


Figure 5. Example of different secret levels (Sl). (**a**) S_1. (**b**) Sl_1 for S_1. (**c**) Sl_2 for S_1. (**d**) Sl_3 for S_1. (**e**) S_2. (**f**) Sl_1 for S_2. (**g**) Sl_2 for S_2. (**h**) Sl_3 for S_2. (**i**) Sl_4 for S_2.

This level structure allows for a hierarchical organization of secret information, with each secret level contributing to the complete reconstruction of the original secret image. Figure 5 visually demonstrates how different secret levels at varying q values can be utilized to reveal different portions of the secret image, with the completeness of the reconstruction being dependent on the specific secret level chosen.

2.5. (k, n) 2D_VCS

In 2023, Lin and Juan introduced a new scheme known as (k, n) 2D_VCS [6]. This scheme presents a novel method for visual cryptography by incorporating random grids and introducing adaptable decryption choices for binary images, employing both OR and XOR operations. Unlike the work by Chen and Tsao [5] discussed in Section 2.3, which solely focused on stacking k shares, our approach broadens the scope to accommodate the stacking of more than k shares, thus offering a higher degree of flexibility. Even it can perfectly recover when stacking n encrypted images by XOR decryption. In addition, to ensure the complete restoration of the original secret image even when all shares are collected, we have incorporated two crucial steps, enhancing the algorithm's efficiency and practicality.

The schematic representation of (k, n) 2D_VCS is illustrated in Figure 2, and the subsequent paragraph outlines the algorithmic details. In the context of Algorithm 3, n denotes the total count of shares, k denotes the threshold value for decryption, b refers to the encrypted pixels before shuffling in the algorithm, and B represents the final output share formed by the shuffled pixels.

Algorithm 3: (*k*, *n*) 2D_VCS [6]

Input:	A $W \times H$ secret binary image S, a threshold value k and a total number n.
Output:	n shares $(B_1, B_2, \ldots, B_n)$.
Step 1.	For each pixel $S[i, j] = s$, repeat Steps 2–6.
Step 2.	Randomly select from {0, 1} to generate n random pixels $b_1, b_2, \ldots, b_n$.
Step 3.	$b_k = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1}$.
Step 4.	Select a number t randomly from $\{k + 1, \ldots, n\}$. Then, $b_t = s \oplus b_1 \oplus b_2 \ldots \oplus b_{t-1}$.
Step 5.	$b_n = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{n-1}$.
Step 6.	Randomly rearrange the above n bits $b_1, \ldots, b_n$ into $B_1[i, j]$, $B_2[i, j]$, $\ldots$, $B_n[i, j]$.
Step 7.	Output n share images $(B_1, B_2, \ldots, B_n)$

Algorithm 3 for (*k*, *n*) 2D_VCS differs only in Steps 4 and 5 from Chen and Tsao's (*k*, *n*) VCS. These two steps lead to important consequences. Step 4 of Algorithm 3 makes the reconstructed image clearer when stacking t ($k < t \leq n$) shares. Step 5 of Algorithm 3 makes it possible to fully recover the secret image after collecting all n shares. Because we do not change the first k shares constructed by (*k*, *n*) VCS, when Algorithm 3 stacks k shares, the quality of the reconstructed image is similar to that yielded by Chen and Tsao's approach. When stacking t shares, the probability of recovering the secret image is $\frac{1}{C_t^n \times (n-k)}$, which makes the proposed scheme more clearly recover the secret using the XOR operator for stacking $k < t \leq n$ shares. Therefore, compared to earlier studies, the method has a higher quality of the recovered image and improved ability to recover the secret image with both XOR and OR operations. Figure 6 depicts the suggested scheme's schematic.

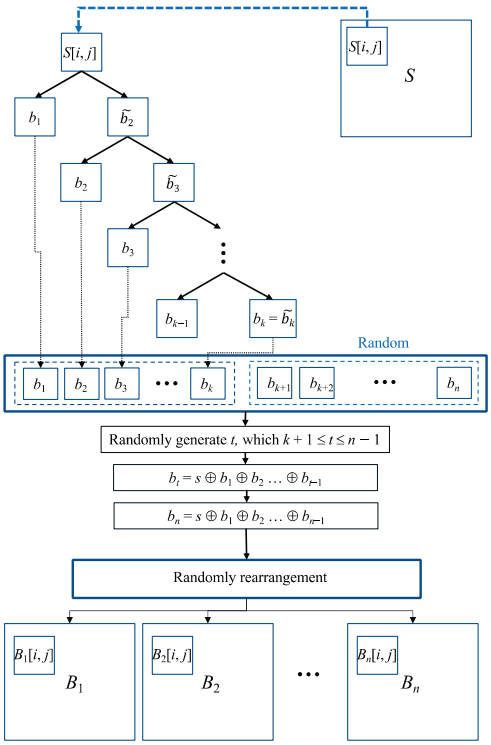


Figure 6. Schematic diagram of Algorithm 3 (*k*, *n*) 2D_VCS.

3. Methods

A new (*k*, *n*) RG-based RIVCS used with two decryption methods (OR and XOR) for binary images is presented in this section, called (*k*, *n*) 2D_RIVCS. The basic idea of this scheme is to encrypt each level of secrets on average, so that when a suitable number of shares are gathered, the original image of the corresponding level can be revealed. The detailed Algorithm 4 is illustrated as follows.

Algorithm 4: (k, n) 2D_RIVCS

Input:	A $W \times H$ secret image S, a number q, where $1 \leq q \leq n - k + 1$, and q secret level images $Sl_1, Sl_2, \ldots, Sl_q$.
Output:	n shares $(B_1, B_2, \ldots, B_n)$.
Step 1.	For every position $[i, j]$, iterate through Steps 2 to 4.
Step 2.	Randomly pick a numerical value d from $\{1, 2, \ldots, q\}$ with equal probability.
Step 3.	Encrypt secret pixel $Sld[i, j]$ by $(k - 1 + d, n)$ 2D_VCS Steps 2–5.
Step 4.	Randomly rearrange these n bits $b_1, b_2, \ldots, b_n$ into $B_1[i, j], B_2[i, j], \ldots, B_n[i, j]$.
Step 5.	Output n share images $(B_1, B_2, \ldots, B_n)$.

Within Algorithm 4, the (k, n) 2D_VCS encryption method is invoked regularly. To begin, a random value d is selected from the range $\{1, 2, \ldots, q\}$ for each pixel. Subsequently, the $(k + d - 1, n)$ 2D_VCS encryption method is applied to the corresponding Sl_d level. The schematic diagram of Algorithm 4 is shown in Figure 7. It will begin by providing an input of a secret image, denoted as S, and a set of q secret levels $\{Sl_1, Sl_2, \ldots, Sl_q\}$. When it comes to decryption, stacking k shares will allow you to recover Sl_1, while stacking $k + 1$ shares will recover Sl_2, and so forth. This implies that the secret image can be divided into a maximum of $n - k + 1$ levels, where $1 \leq q \leq n - k + 1$. The partitioning of secret levels is determined by the user based on the importance of different portions of the secret image. In addition, due to secret levels $Sl_1, Sl_2, \ldots, Sl_q$ satisfying $Sl_i \subset Sl_{i+1}$, when we recover the secret image, it can achieve a progressive effect.

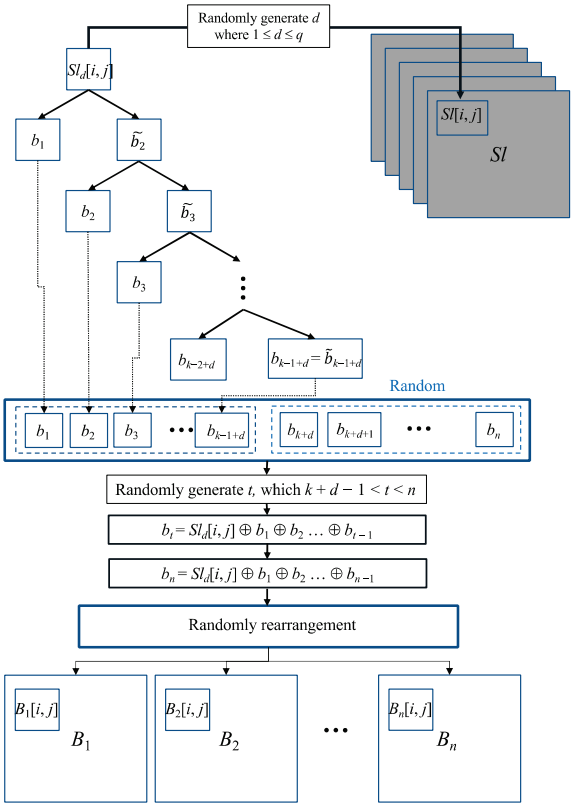


Figure 7. Schematic diagram of Algorithm 4 (k, n) 2D_RIVCS.

We have the option to combine a sufficient quantity of shares ($\geq k$) to restore the original secret image through the OR ($\otimes$) operator, similar to the human visual system (HVS) perception (normal human sight). Leveraging computers provides the capability to achieve more precise secret image reconstruction ($\oplus$). Nevertheless, if you gather fewer than k shares, it becomes infeasible to extract any information about the original secret image.

4. Theoretical Analyses

In this section, we offer a theoretical discussion of the security and visual recognizability of the proposed scheme.

First, this paper will establish the security of Algorithm 3, verifying that each individual encrypted image is secure and that collecting fewer than k encrypted images remains secure. Note that, to facilitate this proof, we will utilize b_i to represent the ith image within the algorithm before rearrangement (Step 4 of Algorithm 4), while B_i will denote the ith image after rearrangement, with $1 \leq i \leq n$. As used in the literature [5], we also define that $b_{1\oplus 2\oplus \ldots \oplus i} = b_1 \oplus b_2 \oplus \ldots \oplus b_i$ and $b_{1\otimes 2\otimes \ldots \otimes i} = b_1 \otimes b_2 \otimes \ldots \otimes b_i$, with $1 \leq i \leq n$.

Lemma 1. *For each grid pixel b_1, b_2, …, b_n generated based on the corresponding secret pixel of S in Algorithm 3,*

$$\text{Prob}(b_i = 0) = \text{Prob}(b_i = 1) = \frac{1}{2}. \tag{5}$$

Proof of Lemma 1. In Step 2 of Algorithm 3, we generate the n grid pixels b_1, b_2, …, b_n with a value of 0 or 1 with equal probability. So, after executing Step 2, we obtain

$$\text{Prob}(b_i = 0) = \text{Prob}(b_i = 1) = \frac{1}{2}, \text{for } 1 \leq i \leq n.$$

Then, b_k, b_t and b_n will be changed in Steps 3, 4 and 5 of Algorithm 3 according to conditions for each corresponding secret pixel s. For Step 3, because $\text{Prob}(b_{1\oplus 2\oplus \ldots \oplus k-1} = 0)$ = $\text{Prob}(b_{1\oplus 2\oplus \ldots \oplus k-1} = 1) = \frac{1}{2}$ and $b_k = b_1 \oplus b_2 \ldots \oplus b_{k-1} \oplus s$, we have

$$\text{Prob}(b_k = 0) = \text{Prob}(b_k = 1) = \frac{1}{2}$$

After executing Step 3. For the same reason, we know $\text{Prob}(b_t = 0) = \text{Prob}(b_t = 1) = \frac{1}{2}$, with $k \leq t \leq n$, after executing Steps 4 and 5. So, $\text{Prob}(b_i = 0) = \text{Prob}(b_i = 1) = \frac{1}{2}$ with $1 \leq i \leq n$, in the end. □

In the following, we define $b[S(0)]$ (and $b[S(1)]$) to denote a pixel b in the recovered image corresponding to a transparent (opaque) pixel in the image S.

Lemma 2. *For each grid pixel b_1, b_2, …, b_n generated based on the corresponding secret pixel of S in Algorithm 3, stacking any p grid pixel $\{b_{i1}, b_{i2}, \ldots, b_{ip}\}$ from $\{b_1, b_2, \ldots, b_n\}$, for which $p < k$, we have*

$$T[b_{i1} \otimes b_{i2} \otimes \ldots \otimes b_{ip}[S(0)]] = T[b_{i1} \otimes b_{i2} \otimes \ldots \otimes b_{ip}[S(1)]] \tag{6}$$

$$T[b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{ip}[S(0)]] = T[b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{ip}[S(1)]]. \tag{7}$$

Proof of Lemma 2. According to Lemma 1, $\text{Prob}(b_i = 0) = \frac{1}{2} = \text{Prob}(b_i = 1)$, where $1 \leq i \leq n$. However, based on the calculations of the algorithm and the properties of the XOR operation, when $b_1 \oplus b_2 \oplus \ldots \oplus b_k = s$ and $b_1 \oplus b_2 \oplus \ldots \oplus b_t = s$, it results in $b_{k+1} \oplus \ldots \oplus b_t = 0$ for some $k < t < n$. Similarly, when $b_1 \oplus b_2 \oplus \ldots \oplus b_t = s$ and $b_1 \oplus b_2 \oplus \ldots \oplus b_n = s$, it leads to $b_{t+1} \oplus \ldots \oplus b_n = 0$. Additionally, when $b_1 \oplus b_2 \oplus \ldots \oplus b_k = s$ and $b_1 \oplus b_2 \oplus \ldots \oplus b_n = s$, it causes $b_{k+1} \oplus \ldots \oplus b_n = 0$. This situation occurs regardless of whether the secret is white or black, allowing us to determine that no matter the stacking of pixels by OR operation or XOR operation, $T[b_{i1} \otimes b_{i2} \otimes \ldots \otimes b_{ip}[S(0)]] = T[b_{i1} \otimes b_{i2} \otimes \ldots \otimes b_{ip}[S(1)]]$, and $T[b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{ip}[S(0)]] = T[b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{ip}[S(1)]]$. □

Theorem 1. *Algorithm 3 (k, n) 2D_VCS is secure.*

Proof of Theorem 1. According to Lemma 1 and Lemma 2, we meet the security condition $T[B[S(0)]] = T[B[S(1)]]$ when stacking any p shares for $1 \leq p < k$. Algorithm 3 guarantees the security of each individual encrypted image and ensures the maintenance of security in the scenario where fewer than k encrypted images are collected. □

Next, we will show the visual recognizability of Algorithm 3, which offers two decryption methods. We will separately prove the visual recognizability based on whether OR or XOR decryption is applied, theoretically ensuring that the contrast is visually recognizable in both cases. First, let us begin with the XOR part.

Lemma 3. *Stacking k grid pixels $\{b_1, b_2, \ldots, b_k\}$, where $b_1, b_2, \ldots, b_k$ are generated after executing Step 2 and Step 3 of Algorithm 3, we have*

$$T[b_1 \oplus b_2 \oplus \ldots \oplus b_k[S(0)]] = 1, \tag{8}$$

$$T[b_1 \oplus b_2 \oplus \ldots \oplus b_k[S(1)]] = 0. \tag{9}$$

Proof of Lemma 3. Based on Step 3 of Algorithm 3, we know that $s = b_1 \oplus b_2 \oplus \ldots \oplus b_k$. So, $b_1 \oplus b_2 \oplus \ldots \oplus b_k = b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1} \oplus (b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1} \oplus s) = s$. Therefore, $T[b_1 \oplus b_2 \oplus \ldots \oplus b_k[S(0)]] = 1$, and $T[b_1 \oplus b_2 \oplus \ldots \oplus b_k[S(1)]] = 0$. □

Lemma 4. *Stacking p ($k < p \leq n$) grid pixels $\{b_1, b_2, \ldots, b_p\}$, where $b_1, b_2, \ldots, b_p$ are generated after executing Step 4 of Algorithm 3, we have*

$$T[b_1 \oplus b_2 \oplus \ \ldots \oplus b_p[S(0)]] = \frac{1}{n-k} + (1 - \frac{1}{n-k}) \cdot \frac{1}{2}, \tag{10}$$

$$T[b_1 \oplus b_2 \oplus \ \ldots \oplus b_p[S(1)]] = (1 - \frac{1}{n-k}) \cdot \frac{1}{2}. \tag{11}$$

Proof of Lemma 4. For (10) and (11), according to Step 4 of Algorithm 3, we know the probability is $\frac{1}{n-k}$ for choosing b_p from $\{b_{k+1}, \ldots, b_n\}$ to reset $b_1 \oplus b_2 \oplus \ldots \oplus b_p = s$. That is, the probability that the pixel is restored is $\frac{1}{n-k}$; otherwise, the pixel will be randomly turned transparent or opaque. □

Lemma 5. *Stacking p ($k < p \leq n$) grid pixels $\{b_1, b_2, \ldots, b_k, b_{n-p+k+1}, b_{n-p+k+2}, \ldots, b_n\}$, where $b_1, b_2, \ldots, b_n$ are generated after executing Step 5 of Algorithm 3, we have*

$$T[b_1 \oplus \ \ldots \oplus b_k \oplus b_{n-p+k+1} \oplus \ \ldots \ \oplus b_n[S(0)]] = \frac{1}{n-k} + (1 - \frac{1}{n-k}) \cdot \frac{1}{2}, \tag{12}$$

$$T[b_1 \oplus \ \ldots \oplus b_k \oplus b_{n-p+k+1} \oplus \ \ldots \ \oplus b_n[S(1)]] = (1 - \frac{1}{n-k}) \cdot \frac{1}{2}. \tag{13}$$

Proof of Lemma 5. In Step 3 of Algorithm 3, we are aware that $b_k = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1}$. Furthermore, in Step 4 and Step 5 we have $b_t = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{t-1}$ for some $k < t \leq n$, and $b_n = s \oplus b_1 \oplus b_2 \oplus \ldots \oplus b_{n-1}$. It implies $s = b_1 \oplus b_2 \oplus \ldots \oplus b_n = b_1 \oplus b_2 \oplus \ldots \oplus b_t \oplus b_{t+1} \oplus \ldots \oplus b_n = s \oplus b_{t+1} \oplus \ldots \oplus b_n$, so $b_{t+1} \oplus \ldots \oplus b_n = 0$ for some $k < t \leq n$ with probability $\frac{1}{n-k}$. When $t = n - p + k$, $b_1 \oplus b_2 \oplus \ldots \oplus b_k \oplus b_{n-p+k+1} \oplus \ldots \oplus b_n = s$. Hence, $T[b_1 \oplus b_2 \oplus \ldots b_{n-p+k+1} \oplus \ldots \oplus b_n\,[S(0)]] = \frac{1}{n-k} + (1 - \frac{1}{n-k}) \cdot \frac{1}{2}$, and $T[b_1 \oplus b_2 \oplus \ldots b_{n-p+k+1} \oplus \ldots \oplus b_n\,[S(0)]] = (1 - \frac{1}{n-k}) \cdot \frac{1}{2}$. □

Lemma 6. *Stacking p ($1 \leq p < n-k$) grid pixels $\{b_{k+1}, b_{k+2}, \ldots, b_{k+p}\}$, where $b_1, b_2, \ldots, b_n$ are generated after executing Step 5 of Algorithm 3, we have*

$$T[b_{k+1} \oplus b_{k+2} \oplus \ \ldots \oplus b_{k+p}[S(0)]] = T[b_{k+1} \oplus b_{k+2} \oplus \ \ldots \oplus b_{k+p}[S(1)]] = \frac{1}{n-k} + (1 - \frac{1}{n-k}) \cdot \frac{1}{2}. \tag{14}$$

Proof of Lemma 6. We know $b_1 \oplus b_2 \oplus \ldots \oplus b_k = s$, and when $t = k + p$ in Step 4 of Algorithm 3, $b_1 \oplus b_2 \oplus \ldots \oplus b_{k+p} = s$. So, $b_{k+1} \oplus b_{k+2} \ldots \oplus b_{k+p} = 0$ when $t = k + p$. □

Lemma 7. *When $p = n - k$, stacking p grid pixels $\{b_{k+1}, b_{k+2}, \ldots, b_{k+p} = b_n\}$, where $b_1, b_2, \ldots, b_n$ are generated after executing Step 5 of Algorithm 3, we have*

$$T[b_{k+1} \oplus b_{k+2} \oplus \ \ldots \oplus b_n[S(0)]] = T[b_{k+1} \oplus b_{k+2} \oplus \ \ldots \oplus b_n[S(1)]] = 1. \tag{15}$$

Proof of Lemma 7. Because $b_1 \oplus b_2 \oplus \ldots \oplus b_k = s$, $b_1 \oplus b_2 \oplus \ldots \oplus b_n = s$ (by Step 5 of Algorithm 3), and $p = n - k$, $b_{k+1} \oplus b_{k+2} \ldots \oplus b_n = 0$. □

Lemma 8. *Stacking p ($k < p < n$) grid pixels $\{b_{n-p+1}, b_{n-p+2}, \ldots, b_n\}$ where $b_1, b_2, \ldots, b_n$ are generated after executing Step 5 of Algorithm 3, we have*

$$T[b_{n-p+1} \oplus \ldots \oplus b_n[S(0)]] = T[b_{n-p+1} \oplus \ldots \oplus b_n\ [S(1)]] = \frac{1}{n-k} + (1 - \frac{1}{n-k}) \cdot \frac{1}{2}. \tag{16}$$

Proof of Lemma 8. We know $b_1 \oplus b_2 \oplus \ldots \oplus b_n = s$, and when $t = n - p$ in Step 4 of Algorithm 3, $b_1 \oplus b_2 \oplus \ldots \oplus b_{n-p} = s$. Hence, $b_{n-p+1} \oplus b_{\ n-p+2} \ldots \oplus b_n = 0$ when $t = n - p$. □

Lemma 9. *In Algorithm 3, stacking any k shares $\{B_{i1}, B_{i2}, \ldots, B_{ik}\}$ from $\{B_1, B_2, \ldots, B_n\}$, we have*

$$T[B_{i1} \oplus B_{i2} \oplus \ldots \oplus B_{ik}[S(0)]] = \begin{cases} \frac{1+(C_k^n - 1) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k > n-k; \\ \frac{2+(C_k^n - 2) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k = n-k; \\ \frac{1+\frac{2}{n-k}+(C_k^n - 1 - \frac{2}{n-k}) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k < n-k \end{cases} \tag{17}$$

$$T[B_{i1} \oplus B_{i2} \oplus \ldots \oplus B_{ik}[S(1)]] \begin{cases} \frac{(C_k^n - 1) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k > n-k; \\ \frac{(C_k^n - 2) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k = n-k; \\ \frac{\frac{2}{n-k}+(C_k^n - 1 - \frac{2}{n-k}) \cdot \frac{1}{2}}{C_k^n}, & \text{if } k < n-k. \end{cases} \tag{18}$$

Proof of Lemma 9. For $k > n - k$, we use Lemma 3; for $k = n - k$, apply Lemmas 3 and 7; and for $k < n - k$, Lemmas 3, 6 and 8 are corresponding. □

Lemma 10. *In Algorithm 3, stacking any p ($k < p < n$) shares $\{B_{i1}, B_{i2}, \ldots, B_{ip}\}$ from $\{B_1, B_2, \ldots, B_n\}$ we have*

$$T[B_{i1} \oplus B_{i2} \oplus \ldots \oplus B_{\mathrm{ip}}[S(0)]] = \begin{cases} \frac{\frac{2}{n-k}+(C_p^n - \frac{2}{n-k}) \cdot \frac{1}{2}}{C_p^n}, & \text{if } p > n-k; \\ \frac{1+\frac{2}{n-k}+(C_p^n - 1 - \frac{2}{n-k}) \cdot \frac{1}{2}}{C_p^n}, & \text{if } p = n-k; \\ \frac{\frac{4}{n-k}+(C_p^n - \frac{4}{n-k}) \cdot \frac{1}{2}}{C_p^n}, & \text{if } p < n-k. \end{cases} \tag{19}$$

$$T[B_{i1} \oplus B_{i2} \oplus \ldots \oplus B_{ip}[S(1)]] \begin{cases} \frac{\left(C_p^n - \frac{2}{n-k}\right) \cdot \frac{1}{2}}{C_p^n}, \text{ if } p > n-k; \\ \frac{1+\left(C_p^n - 1 - \frac{2}{n-k}\right) \cdot \frac{1}{2}}{C_p^n}, \text{ if } p = n-k; \\ \frac{\frac{2}{n-k}+\left(C_p^n - \frac{4}{n-k}\right) \cdot \frac{1}{2}}{C_p^n}, \text{ if } p < n-k. \end{cases} \tag{20}$$

Proof of Lemma 10. When we stack any p shares, we will meet one of three difference cases: (1) $p > n-k$; (2) $p = n-k$; and (3) $p < n-k$. In the first case we use Lemma 4 and Lemma 5 to prove it. In the second case we meet Lemmas 4, 5 and 7. When the third case happens, we use Lemmas 4, 5, 6 and 8. □

Theorem 2. *Algorithm 3 (k, n) 2D_VCS has visual recognizability when using XOR decryption.*

Proof of Theorem 2. To achieve visual recognizability as defined in Definition 4, it is necessary that $T[B[S(0)]] - T[B[S(1)]]$ be greater than 0. According to Lemma 9 and Lemma 10, we can deduce the light transmission for restoring p encrypted images using XOR, where $k \leq p < n$. Therefore, we can calculate the following:

$$T[B[S(0)]] - T[B[S(1)]] = \begin{cases} \frac{1}{C_p^n}, \; if \; p = k \; and \; p \neq n-k; \\ \frac{2}{C_p^n}, \; if \; p = k \; and \; p = n-k; \\ \frac{2}{C_p^n \times (n-k)}, \; if \; k < p < n; \\ 1, \; if \; p = n. \end{cases} \tag{21}$$

Based on the formula above, it is evident that Algorithm 3 for (k, n) 2D_VCS meets the condition of being visually recognizable when collecting k shares or more. □

The contrast under various cases for (k, n) 2D_VCS for $2 \leq k \leq p \leq n \leq 6$ using XOR decryption can be deduced from Lemma 9 and Lemma 10, as presented in Table 3.

Table 3. The α of the restored image in (k, n) 2D_VCS by using XOR operation.

(*k*, *n*)	***p* = 2**	***p* = 3**	***p* = 4**	***p* = 5**	***p* = 6**
(2, 4)	0.1111	0.1818	1	-	-
(3, 4)	0	0.1818	1	-	-
(4, 4)	0	0	1	-	-
(2, 5)	0.0674	0.0440	0.0930	1	-
(3, 5)	0	0.0690	0.1429	1	-
(4, 5)	0	0	0.1429	1	-
(5, 5)	0	0	0	1	-
(2, 6)	0.0449	0.0167	0.0220	0.0571	1
(3, 6)	0	0.0333	0.0301	0.0769	1
(4, 6)	0	0	0.0455	0.1176	1
(5, 6)	0	0	0	0.1176	1
(6, 6)	0	0	0	0	1

With the XOR decryption ($\oplus$) part thoroughly addressed, this paper will proceed to discuss the visual recognizability when the OR decryption ($\otimes$) method is applied. Before proceeding with the proof, let us introduce a variable s, for which $1 \leq s \leq n-k$, representing the number of encrypted images (out of the total p) that are chosen for restoration and fall within the range of the $(k+1)$th to the nth encrypted image. For instance, if we are restoring from 3 encrypted images and s is equal to 2, this implies that 1 encrypted image will be selected from $\{b_1, b_2, \ldots, b_k\}$, and 2 encrypted images will be chosen from $\{b_{k+1}, b_{k+2}, \ldots, b_n\}$. The following lemmas will be categorized based on the different cases of these s selected encrypted images.

Lemma 11. *Stacking k grid pixels* $\{b_1, b_2, \ldots, b_k\}$*, where* $b_1, b_2, \ldots, b_k$ *are generated after executing Step 2 and Step 3 of Algorithm 3, we have*

$$T[b_1 \otimes b_2 \otimes \ \ldots \otimes b_k[S(0)]] = \frac{1}{2^{k-1}}, \tag{22}$$

$$T[b_1 \otimes b_2 \otimes \ldots \otimes b_k[S(1)]] = 0. \tag{23}$$

Proof of Lemma 11. According to Lemma 1, we know that $b_1, b_2, \ldots, b_{k-1}$ are created randomly and independently; thus,

$$T[b_1 \otimes b_2 \otimes \ \ldots \otimes b_{k-1}[S(0)]] = T[b_1 \otimes b_2 \otimes \ \ldots \otimes b_{k-1}[S(1)]] = \frac{1}{2^{k-1.}}$$

The corresponding pixel b_k is generated by $b_k = b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1} \oplus s$. So, we have $b_1 \otimes b_2 \otimes \ldots \otimes b_k = b_1 \otimes b_2 \otimes \ldots \otimes b_{k-1} \otimes (b_1 \oplus b_2 \oplus \ldots \oplus b_{k-1} \oplus s)$. Hence,

$$T[b_1 \otimes b_2 \otimes \ \ldots \otimes b_k[S(0)]] = \frac{1}{2^{k-1}},$$

$$T[b_1 \otimes b_2 \otimes \ldots \otimes b_k[S(1)]] = 0. \ \square$$

When stacking p encrypted images using the OR decryption method, for which $k \leq p < n$, the value of s will gradually increase from 1 to $\min\{n - k - 1, p\}$. These s images will be consecutive encrypted images starting either from b_{k+1} exclusive-or counting backward from b_n. For instance, in the case of (3, 6) 2D_VCS, when p is 3 and s is 1, the set of s images will be $\{b_4\}$ or $\{b_6\}$, and when s is 2, the set of s images will be $\{b_4, b_5\}$ or $\{b_5, b_6\}$. The maximum value for s is $\min\{n - k - 1, p\}$, which is $\min\{2, 3\}$, which is equal to 2. Based on the aforementioned conditions, when decrypting p encrypted images using the OR decryption method, the set of p images containing s consecutive encrypted images from $\{b_{k+1}, b_{k+2}, \ldots, b_n\}$ starting either from b_{k+1} exclusive-or counting backward from b_n, let the recovered image be referred to as $R_{p,1}$, and let the set of all possible $R_{p,1}$ be SR_1.

Lemma 12. *The light transmission of* $R_{p,1}$ *in* SR_1 *can be derived as follows:*

$$\begin{aligned}
&\Sigma_{R_{p,1} \in SR_1} T\left[R_{p,1}[S(0)]\right] \\
&\quad = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} \frac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2 \right) \\
&\quad + \left(\frac{p+n-2k}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-p-2,0\}} + 1 \right) \times 2 \right) \\
&\quad + \left(\sum_{s=1}^{p-k-1} \frac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} + C_{p-s-k}^{\max\{n-k-2-s,0\}} \right) \times 2 \right)
\end{aligned} \tag{24}$$

$$\begin{aligned}
&\Sigma_{R_{p,1} \in SR_1} T\left[R_{p,1}[S(1)]\right] \\
&\quad = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} \frac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2 \right) \\
&\quad + \left(\frac{p+n-2k}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-p-2,0\}} - 1 \right) \times 2 \right) \\
&\quad + \left(\sum_{s=1}^{p-k-1} \frac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}} \right) \times 2 \right)
\end{aligned} \tag{25}$$

Proof of Lemma 12. In Lemmas 5, 6 and 7, it can be observed that when selecting s consecutive pixels from $\{b_{k+1}, b_{k+2}, \ldots, b_n\}$ starting either from b_{k+1} or counting backward from b_n, there is a $\frac{1}{n-k}$ probability of obtaining white pixels. Next, based on the difference

between p and s, we can divide the possible combination of the remaining $p - s$ pixels of $R_{p,1}$ into three cases:

(1) $p - s < k$: In this case, for each of the remaining $p - s$ pixels, the probability of being black or white is $\frac{1}{2}$, resulting in $C_{p-s}^{k+\max\{n-k-s-2,\,0\}} \times 2$ combinations. Here, $k + \max\{n - k - s - 2,\, 0\}$ counts the number of pixels that can be selected from the initial k and the remaining $n - k$ pixels, except the s consecutive pixels and the 2 pixels that could potentially continue the sequence or be against the definition of the exclusive-or. We multiply by 2 because there are two cases: starting from b_{k+1} and counting backward from b_n. Hence, we can derive:

$$T\left[R_{p,1}[S(0)]\right] = \left(\frac{s}{n-k} \times \frac{1}{2^{p-1}} + \left(1 - \frac{s}{n-k}\right) \times \frac{1}{2^p}\right) \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2$$

This simplifies to:

$$T\left[R_{p,1}[S(0)]\right] = \frac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2$$

(2) $p - s = k$: In this case, for the remaining $p - s$ pixels that exactly match the initial k pixels, there are $C_{p-s}^{k} \times 2 = 2$ combinations. Similarly, there are $(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s}^{k}) \times 2$ combinations for the remaining pixels that do not select the initial k pixels completely. Hence, we can derive:

$$\begin{aligned} T\left[R_{p,1}[S(0)]\right] &= \left(\tfrac{s}{n-k} \times \tfrac{1}{2^{p-2}} + \left(1 - \tfrac{s}{n-k}\right) \times \tfrac{1}{2^{p-1}}\right) \times C_{p-s}^{k} \times 2 \\ &+ \left(\tfrac{s}{n-k} \times \tfrac{1}{2^{p-1}} + \left(1 - \tfrac{s}{n-k}\right) \times \tfrac{1}{2^{p}}\right) \times \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s}^{k}\right) \times 2 \end{aligned}$$

This simplifies to:

$$T\left[R_{p,1}[S(0)]\right] = \frac{p+n-2k}{(n-k)\times 2^p} \times \left(C_{k}^{k+\max\{n-k-2-s,0\}} + 1\right) \times 2$$

(3) $p - s > k$: In this case, for the remaining $p - s$ pixels that are greater than k, there are $C_{p-s-k}^{\max\{n-k-2-s,0\}} \times 2$ combinations which choose all of the first k pixels completely. In addition, for the remaining cases, there are $(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}}) \times 2$ combinations. Hence, we can derive:

$$\begin{aligned} T\left[R_{p,1}[S(0)]\right] &= \left(\tfrac{s}{n-k} \times \tfrac{1}{2^{p-2}} + \left(1 - \tfrac{s}{n-k}\right) \times \tfrac{1}{2^{p-1}}\right) \times C_{p-s-k}^{\max\{n-k-2-s,0\}} \times 2 \\ &+ \left(\tfrac{s}{n-k} \times \tfrac{1}{2^{p-1}} + \left(1 - \tfrac{s}{n-k}\right) \times \tfrac{1}{2^{p}}\right) \times \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times 2 \end{aligned}$$

This simplifies to:

$$T\left[R_{p,1}[S(0)]\right] = \frac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} + C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times 2$$

Combining the three cases with variations in s, we can obtain:

$$\begin{aligned} &\Sigma_{R_{p,1}\in SR_1} T\left[R_{p,1}[S(0)]\right] \\ &\quad = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} \tfrac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2\right) \\ &\quad + \left(\tfrac{p+n-2k}{(n-k)\times 2^p} \times \left(C_{k}^{k+\max\{n-p-2,0\}} + 1\right) \times 2\right) \\ &\quad + \left(\sum_{s=1}^{p-k-1} \tfrac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} + C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times 2\right) \end{aligned}$$

In the part of $T[R_{p,1}[S(1)]]$, combinations of light transmission including the first k shares will all be 0. Therefore, we can obtain:

(1) when $p - s < k$, we have $T\left[R_{p,1}[S(1)]\right] = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} \frac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2\right)$

(2) when $p - s = k$, we have $T\left[R_{p,1}[S(1)]\right] = \left(\frac{p+n-2k}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-k-2-s,0\}} - 1\right) \times 2\right)$

(3) when $p - s > k$, we have $T\left[R_{p,1}[S(1)]\right] = \left(\sum_{s=1}^{p-k-1} \frac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times 2\right)$

Combining the three cases with variations in s, we can obtain:

$$\begin{aligned}
&\Sigma_{R_{p,1}\in SR_1} T\left[R_{p,1}[S(1)]\right] \\
&\quad = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} \frac{s+n-k}{(n-k)\times 2^p} \times C_{p-s}^{k+\max\{n-k-2-s,0\}} \times 2\right) \\
&\quad + \left(\frac{p+n-2k}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-p-2,0\}} - 1\right) \times 2\right) \\
&\quad + \left(\sum_{s=1}^{p-k-1} \frac{s+n-k}{(n-k)\times 2^p} \left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times 2\right)
\end{aligned}$$

According to the above proof, (24) and (25) are established. □

Based on these conditions, when stacking p encrypted images using the OR decryption method, for which $k \le p < n$, the value of s will gradually increase from 2 to $\min\{n - k - 1, p\}$. These s shares will be the sum of two sets of consecutive encrypted images, one counted from b_{k+1} forward and the other counted from b_n backward. Let the recovered image be referred to $R_{p,2}$, and let the set of all possible $R_{p,2}$ be SR_2. For example, in the (3, 7) 2D_VCS, when p is 3 and s is 2, these s shares will be $\{b_4, b_7\}$, and when s is 3, these s shares will be $\{b_4, b_5, b_7\}$ or $\{b_4, b_6, b_7\}$.

Lemma 13. *The light transmission of $R_{p,2}$ in SR_2 can be derived as follows:*

$$\begin{aligned}
&\Sigma_{R_{p,2}\in SR_2} T\left[R_{p,2}[S(0)]\right] \\
&\quad = \left(\left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} (s-1)\left(\frac{s+n-k}{(n-k)\times 2^p}\right) \times C_{p-s}^{k+\max\{n-k-2-s,0\}}\right)\right) \\
&\quad + \left((p-k-1) \times \frac{(n+p-2k)}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-p-2,0\}} + 1\right)\right) \\
&\quad + \left(\sum_{s=2}^{p-k-1} 2(s-1)\left(\frac{s+n-k}{(n-k)\times 2^p}\right)\left(C_{p-s}^{k+\max\{n-k-2-s,0\}} + C_{p-s-k}^{\max\{n-k-2-s,0\}}\right)\right)
\end{aligned} \tag{26}$$

$$\begin{aligned}
&\Sigma_{R_{p,2}\in SR_2} T\left[R_{p,2}[S(1)]\right] \\
&\quad = \left(\sum_{s=p-k+1}^{\min\{n-k-1,p\}} (s-1)\left(\frac{s+n-k}{(n-k)\times 2^p}\right) \times C_{p-s}^{k+\max\{n-k-2-s,0\}}\right) \\
&\quad + \left((p-k-1) \times \frac{(n+p-2k)}{(n-k)\times 2^p} \times \left(C_k^{k+\max\{n-p-2,0\}} - 1\right)\right) \\
&\quad + \left(\sum_{s=2}^{p-k-1} 2(s-1)\left(\frac{s+n-k}{(n-k)\times 2^p}\right)\left(C_{p-s}^{k+\max\{n-k-2-s,0\}} - C_{p-s-k}^{\max\{n-k-2-s,0\}}\right)\right)
\end{aligned} \tag{27}$$

Proof of Lemma 13. The proof process is similar to Lemma 12, as Lemmas 5, 6 and 7 have shown that there is a $\frac{1}{(n-k)}$ probability of white in $\{b_{k+1}, \ldots, b_n\}$ when counting from b_{k+1} or from b_n backward. Unlike Lemma 12, where one set of s shares is chosen from $\{b_{k+1}, \ldots, b_n\}$, Lemma 13 selects two sets that together add up to s shares. Similar to Lemma 12, it is divided into three cases, although the combination here does not need to be multiplied by 2. Instead, a variable i is used, and its range is from 1 to s. i represents the shares counted

from b_{k+1}, and the remaining $s-i$ shares are counted from b_n backward. Therefore, the three cases are as follows:

(1) when $p-s<k$, we have $T[R_{p,2}[S(0)]] = \left(\frac{s}{n-k}\times\frac{1}{2^{p-1}}+\left(1-\frac{s}{n-k}\right)\times\frac{1}{2^p}\right)\times C_{p-s}^{k+\max\{n-k-2-s,0\}}$, which simplifies to $\left(\frac{s+n-k}{(n-k)\times 2^p}\right)\times C_{p-s}^{k+\max\{n-k-2-s,0\}}$.

(2) when $p-s=k$, we have $T[R_{p,2}[S(0)]] = \left(\frac{s}{n-k}\times\frac{1}{2^{p-2}}+\left(1-\frac{s}{n-k}\right)\times\frac{1}{2^{p-1}}\right)\times C_{p-s}^{k}+\left(\frac{s}{n-k}\times\frac{1}{2^{p-1}}+\left(1-\frac{s}{n-k}\right)\times\frac{1}{2^p}\right)\times\left(C_{p-s}^{k+\max\{n-k-2-s,0\}}-C_{p-s}^{k}\right)$, which simplifies to $\left(\frac{p+n-2k}{(n-k)\times 2^p}\right)\times\left(C_{p-s}^{k+\max\{n-k-2-s,0\}}+1\right)$.

(3) when $p-s>k$, we have $T[R_{p,2}[S(0)]] = \left(\frac{s}{n-k}\times\frac{1}{2^{p-2}}+\left(1-\frac{s}{n-k}\right)\times\frac{1}{2^{p-1}}\right)\left(C_{p-s-k}^{\max\{n-k-2-s,0\}}\right)+\left(\frac{s}{n-k}\times\frac{1}{2^{p-1}}+\left(1-\frac{s}{n-k}\right)\times\frac{1}{2^p}\right)\times\left(C_{p-s}^{k+\max\{n-k-2-s,0\}}-C_{p-s-k}^{\max\{n-k-2-s,0\}}\right)$, which simplifies to $\left(\frac{s+n-k}{(n-k)\times 2^p}\right)\times 2\left(C_{p-s}^{k+\max\{n-k-2-s,0\}}+C_{p-s-k}^{\max\{n-k-2-s,0\}}\right)$.

Combining the three cases with variations in s, we can obtain (26). The proof for (27) $\Sigma_{R_{p,2}\in SR_2}\,T[R_{p,2}[S(1)]]$ is similar to the proof of (26); the results can be concluded. □

There is another possible combination when stacking p encrypted images using the OR decryption method for $n-k\le p<n$. That is, s equals $n-k$, indicating that $\{b_{k+1},\ldots,b_n\}$ are all included in the p shares. Let the recovered image be referred to $R_{p,3}$ and the set of all possible $R_{p,3}$ be SR_3.

Lemma 14. *The light transmission of $R_{p,3}$ in SR_3 can be derived as follows:*

$$\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(0)]\right]=\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(1)]\right]=\left(\frac{2n-2k-1}{(n-k)\times 2^{p-1}}\right)\times C_{p-n+k}^{k}. \quad (28)$$

Proof of Lemma 14. In the case of $R_{p,3}$, the proof process is similar to Lemmas 12 and 13. When s equals $n-k$, we have $b_{k+1}\oplus b_{k+2}\oplus\ldots\oplus b_n=0$. In this case, there is a probability of $\frac{1}{n-k}$ that t equals n, and as a result, only b_n has been computed (not random) in b_{k+1} to b_n. The remaining probability of $1-\frac{1}{n-k}$ may select one pixel b_t from b_{k+1} to b_{n-1}, such that both b_t and b_n have been computed. Furthermore, here $s=n-k$ and $p-s=p-n+k<k$ because $p<n$. Therefore, we only need to consider $p-s<k$, leading to the following:

$$\begin{aligned}\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(0)]\right]&=\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(1)]\right]\\&=\left(\frac{1}{n-k}\times\frac{1}{2^{p-1}}+\left(1-\frac{1}{n-k}\right)\times\frac{1}{2^{p-2}}\right)\times C_{p-s}^{k}\end{aligned}$$

Substituting $s=n-k$ yields and simplifies to:

$$\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(0)]\right]=\Sigma_{R_{p,3}\in SR_3}T\left[R_{p,3}[S(1)]\right]=\left(\frac{2n-2k-1}{(n-k)\times 2^{p-1}}\right)\times C_{p-n+k}^{k}.\square$$

When stacking p encrypted images using the OR decryption method, for which $k\le p<n$, and it always includes the first k encrypted images but excludes the condition as $R_{p,1}$ or $R_{p,2}$, let the recovered image be referred to $R_{p,4}$. In addition, let the set of all possible $R_{p,4}$ be SR_4.

Lemma 15. *The light transmission of $R_{p,4}$ in SR_4 can be derived as follows:*

$$\Sigma_{R_{p,4}\in SR_4} T\left[R_{p,4}[S(0)]\right] = \begin{cases} \left(C_{p-k}^{n-k} - 2C_{p-1-k}^{\max\{n-k-3,0\}} - \sum_{s=2}^{p-k}(s+1)C_{p-s-k}^{\max\{n-k-2-s,0\}}\right) \times \frac{1}{2^{p-1}}, & if\ k < p < n \\ \frac{1}{2^{p-1}}, & if\ p = k \end{cases} \tag{29}$$

$$\Sigma_{R_{p,4}\in SR_4} T\left[R_{p,4}[S(1)]\right] = 0. \tag{30}$$

Proof of Lemma 15. There will be C_{p-k}^{n-k} combinations when taking p out of n, which are guaranteed to include the first k pixels. However, we need to subtract the combinations already calculated in $R_{p,1}$ and $R_{p,2}$. In the case of $R_{p,1}$ and when $p - s = k$ and $p - s > k$, there are $\sum_{s=1}^{p-k} 2C_{p-s-k}^{\max\{n-k-2-s,0\}}$ combinations that include the first k shares. In the case of $R_{p,2}$ and when $p - s = k$ and $p - s > k$, there are $\sum_{s=2}^{p-k} C_{p-s-k}^{\max\{n-k-2-s,0\}} \times (s+1)$ combinations that include the first k shares. So, the number of possible combinations for $R_{p,4}$ is

$$\begin{aligned} & C_{p-k}^{n-k} - \left(\sum_{s=1}^{p-k} 2C_{p-s-k}^{\max\{n-k-2-s,0\}} + \sum_{s=2}^{p-k} C_{p-s-k}^{\max\{n-k-2-s,0\}} \times (s-1)\right) \\ &= C_{p-k}^{n-k} - \left(\sum_{s=2}^{p-k} C_{p-s-k}^{\max\{n-k-2-s,0\}} \times (2+s-1) + 2C_{p-1-k}^{\max\{n-k-3,0\}}\right. \\ &= C_{p-k}^{n-k} - 2C_{p-1-k}^{\max\{n-k-3,0\}} - \sum_{s=2}^{p-k}(s+1)C_{p-s-k}^{\max\{n-k-2-s,0\}}. \end{aligned}$$

Next, based on Lemma 11, we obtain the result of stacking the first k images, and the results can be concluded. □

At last, when stacking p encrypted images using the OR decryption method, for which $k \le p < n$, these are still some combinations not included in $R_{p,1}$, $R_{p,2}$, $R_{p,3}$ or $R_{p,4}$. Let the recovered image be referred to as $R_{p,5}$ and the set of all possible $R_{p,5}$ be SR_5.

Lemma 16. *The light transmission of $R_{p,5}$ in SR_5 can be derived as follows:*

$$\Sigma_{R_{p,5}\in SR_5} T\left[R_{p,5}[S(0)]\right] = \Sigma_{R_{p,5}\in SR_5} T\left[R_{p,5}[S(1)]\right] = \beta \times \frac{1}{2^p}. \tag{31}$$

where $\beta = C_p^n - \sum_{s=1}^{\min\{n-k-1,p\}} 2C_{p-s}^{\max\{n-k-2-s,0\}} - \sum_{s=2}^{\min\{n-k-1,p\}} \sum_{i=1}^{s-1} C_{p-s}^{\max\{n-k-2-s,0\}} - C_{p-n+k}^{k} - \left(C_{p-k}^{n-k} - \sum_{s=2}^{p-k}(s+1)C_{p-s-k}^{\max\{n-k-2-s,0\}} - 2C_{p-1-k}^{\max\{n-k-3,0\}}\right)$.

Proof of Lemma 16. There will be C_p^n combinations when taking p out of n. However, we need to subtract the combinations already calculated in $R_{p,1}$, $R_{p,2}$, $R_{p,3}$ and $R_{p,4}$. So, we obtain:

$$\beta = C_p^n - \sum_{s=1}^{\min\{n-k-1,p\}} 2C_{p-s}^{\max\{n-k-2-s,0\}} - \sum_{s=2}^{\min\{n-k-1,p\}} \sum_{i=1}^{s-1} C_{p-s}^{\max\{n-k-2-s,0\}} - C_{p-n+k}^{k} - \left(C_{p-k}^{n-k} - \sum_{s=2}^{p-k}(s+1)C_{p-s-k}^{\max\{n-k-2-s,0\}} - 2C_{p-1-k}^{\max\{n-k-3,0\}}\right)$$

Then, based on Lemma 1, we obtain the result of stacking all of the p images in $R_{p,5}$, and the results can be concluded. □

Lemma 17. *In Algorithm 3, stacking all n encrypted images from $\{B_1, B_2, \ldots, B_n\}$, we have*

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n\ [S(0)]] = \frac{2(n-k)-1}{(n-k)\times 2^{p-2}}, \tag{32}$$

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n\ [S(1)]] = 0. \tag{33}$$

Proof of Lemma 17. When all the encrypted images are stacked together using the OR operator, it is known that the first k images will be equal to s, and all n images will also be equal to s. In the sixth step of Algorithm 3, the first t images will also be equal to s, but there is a $\frac{1}{n-k}$ probability that t will be equal to n. Therefore, we can derive the following:

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n\ [S(0)]] = \left(\frac{1}{n-k} \times \frac{1}{2^{p-2}} + \left(1 - \frac{1}{n-k}\right) \times \frac{1}{2^{p-3}}\right),$$

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n [S(1)]] = 0.$$

Simplifying, we obtain:

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n [S(0)]] = \frac{2(n-k)-1}{(n-k) \times 2^{p-2}}$$

$$T[B_1 \oplus B_2 \oplus \ldots \oplus B_n\ [S(1)]] = 0. \square$$

Lemma 18. *In Algorithm 3, stacking any p (k≤ p≤ n) shares {B_{i1}, B_{i2}, …, B_{ip}} from {B_1, B_2, …, B_n}, we have*

$$T[B_{i1} \oplus\ B_{i2} \oplus \ldots \oplus\ B_{ip}\ [S(0)]] = \begin{cases} \frac{\Sigma_{R_{p,1} \in SR_1} T\left[R_{p,1}[S(0)]\right] + \Sigma_{R_{p,2} \in SR_2} T\left[R_{p,2}[S(0)]\right] + \Sigma_{R_{p,3} \in SR_3} T\left[R_{p,3}[S(0)]\right] + \Sigma_{R_{p,4} \in SR_4} T\left[R_{p,4}[S(0)]\right] + \Sigma_{R_{p,5} \in SR_5} T\left[R_{p,5}[S(0)]\right]}{C_p^n}, & \text{if } k \le p < n; \\ \frac{2(n-k)-1}{(n-k) \times 2^{p-2}}, \quad if\ p = n \end{cases} \tag{34}$$

$$T[B_{i1} \oplus\ B_{i2} \oplus \ldots \oplus\ B_{ip}\ [S(1)]] = \begin{cases} \frac{\Sigma_{R_{p,1} \in SR_1} T\left[R_{p,1}[S(1)]\right] + \Sigma_{R_{p,2} \in SR_2} T\left[R_{p,2}[S(1)]\right] + \Sigma_{R_{p,3} \in SR_3} T\left[R_{p,3}[S(1)]\right] + \Sigma_{R_{p,4} \in SR_4} T\left[R_{p,4}[S(1)]\right] + \Sigma_{R_{p,5} \in SR_5} T\left[R_{p,5}[S(1)]\right]}{C_p^n}, & \text{if } k \le p < n; \\ 0, \quad if\ p = n \end{cases} \tag{35}$$

Proof of Lemma 18. Based on Lemma 11 to Lemma 18, we have considered all possible cases when using OR decryption in (k, n) 2D_VCS. Therefore, to calculate the light transmission (T), we simply need to sum them up. □

From Lemma 17 and Lemma 18, we can derive the contrast under different cases for OR decryption in (k, n) 2D_VCS, as illustrated in Table 4.

Table 4. The α of the restored image in (k, n) 2D_VCS by using the OR operation.

(*k*, *n*)	*p* = 2	*p* = 3	*p* = 4	*p* = 5	*p* = 6
(2, 4)	0.0606	0.1579	0.375	-	-
(3, 4)	0	0.0526	0.25	-	-
(4, 4)	0	0	0.125	-	-
(2, 5)	0.0382	0.0786	0.1154	0.2083	-
(3, 5)	0	0.0204	0.0674	0.1875	-
(4, 5)	0	0	0.0227	0.125	-
(5, 5)	0	0	0	0.0625	-
(2, 6)	0.0260	0.0492	0.0621	0.0704	0.1094
(3, 6)	0	0.0106	0.0277	0.0495	0.1042
(4, 6)	0	0	0.0074	0.0294	0.0938
(5, 6)	0	0	0	0.0099	0.0625
(6, 6)	0	0	0	0	0.0313

Theorem 3. *Algorithm 3 (*k, n*) 2D_VCS has visual recognizability when using OR decryption.*

Proof of Theorem 3. From Lemma 17, Lemma 18, it can be observed that when collecting k or more encrypted images and using the OR decryption method, the contrast (α) is always greater than 0. A contrast greater than 0 implies that theoretically, the information of the secret image S is visible, and the decryption is successful, meeting the conditions for visual recognizability. Therefore, Theorem 3 is proven. □

Based on the aforementioned Lemmas and Theorems, it becomes evident that Algorithm 3 for (k, n) 2D_VCS is both secure and functional. This conclusion reaffirms that the system not only preserves the confidentiality of the secret image but also offers a practical approach to decrypting it, making it a robust and reliable solution for visual secret sharing.

Now, we will demonstrate the theoretical security and usability of Algorithm 4 presented in this paper for (k, n) 2D_RIVCS.

Theorem 4. *Algorithm 4 for (k, n) 2D_RIVCS is secure and visually recognizable.*

Proof of Theorem 4. According to Algorithm 4, (k, n) 2D_RIVCS is primarily constructed by repetitively employing (k, n) 2D_VCS to encrypt different secret levels Sl_d. Therefore, to establish the security and visual recognizability of Algorithm 4, it is necessary to first demonstrate the security and visual recognizability of Algorithm 3. As shown in Theorems 1, 2 and 3, the security and visual recognizability of Algorithm 3 have already been established. Consequently, it can be inferred that Algorithm 4 is secure and visually recognizable as well. □

As the secret levels Sl of the secret image S are arbitrarily determined by the user, the proportion of these levels can vary with each division. Consequently, the fidelity of the reconstruction may differ when different numbers of encrypted images p are collected. Therefore, it is impractical to establish a fixed contrast between the reconstructed image and the secret image S for each case. According to Theorem 1, it is evident that obtaining a minimum of k or more encrypted images enables the recovery of clues related to the secret image S. With fewer than k images, only a chaotic and indecipherable image can be obtained.

Next, this paper analyzes the time complexity of Algorithm 4. In steps 1~4 of Algorithm 4, it can be observed that, for each pixel (i, j), the steps from 2~4 in Algorithm 4 are repeated. The size of the secret image is $W \times H$, indicating that these steps will be repeated WH times. Note that the time complexity of Steps 2–5 of Algorithm 3 is $O(k + t + n) = O(n)$. Therefore, the time complexity of Algorithm 4 is $O(nWH)$. It is equal to the size of the output, so Algorithm 4 is a linear-time algorithm. Due to the nature of the algorithm, hardware devices and processor efficiency do not significantly impact the execution speed. Users can employ any hardware, software and programming language to implement the algorithm presented in this paper. However, we will provide detailed information about the specific software used in the experiments (in the next section), namely MATLAB R2022b.

Algorithm 4 combines Algorithm 3 and a region incrementing visual cryptography scheme (RIVCS), thus introducing a novel application for Algorithm 3 while affording a broader spectrum of use cases. This innovative approach leverages the inherent strengths of both Algorithm 3 and RIVCS, paving the way for the fusion of these two cryptographic techniques. As a result, Algorithm 4 is capable of addressing diverse scenarios, showcasing its adaptability and extending the realm of possibilities for secure and visually recognizable image encryption and decryption.

5. Experimental Results

In this section, we will present the experimental results of Algorithm 4, the (k, n) 2D_RIVCS. The experiments were conducted using MATLAB R2022b, and all images in the dataset have dimensions of 300×300 pixels. This paper will now proceed to discuss and analyze the outcomes of our experiments with the 2D_RIVCS algorithm.

Figures 8 and 9 illustrate the experimental results of (2, 3) 2D_RIVCS. In both sets of experiments, the parameters such as k, n and S remain consistent, and the number of secret

levels is the same. The only variation lies in the user-defined partitioning of secret levels. It is evident from Figures8b,c and 9b,c that in Figure 8, the secret image S is uniformly divided, while Figure 9 depicts an arbitrary division, resulting in varying levels of information on each side. The encrypted results are presented in Figures 8d–f and 9d–f. Upon observation, it is evident that the encrypted image B does not reveal any information related to the secret image S.

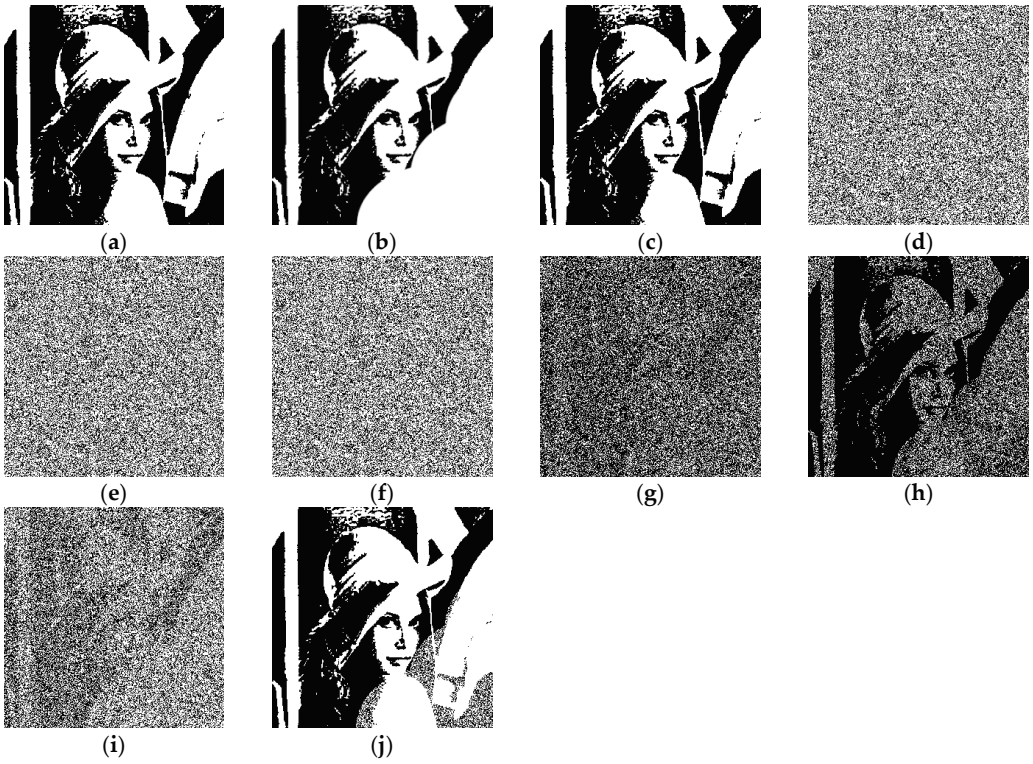


Figure 8. The experimental results of the proposed (2, 3) 2D_RIVCS with two secret levels. (**a**) S. (**b**) Sl_1. (**c**) Sl_2. (**d**) B_1. (**e**) B_2. (**f**) B_3. (**g**) $B_1 \otimes B_2$. (**h**) $B_1 \otimes B_2 \otimes B_3$. (**i**) $B_1 \oplus B_2$. (**j**) $B_1 \oplus B_2 \oplus B_3$.

Furthermore, Figures8g,h and 9g,h show the results of stacking two and three images using the OR decryption method. The contrasts between Figure 8g,h and Sl_1 are 0.06517 and 0.35696, respectively, and the contrasts with Sl_2 are 0.05313 and 0.32209, respectively. In Figure 9g,h, the contrasts with Sl_1 are 0.067229 and 0.31722, while with Sl_2, they are 0.012283 and 0.15064. In the XOR decryption part, it can be observed that there is a significant difference in the contrast values between Figures 8 and 9. This demonstrates that even under identical conditions of k, n, S and the number of secret levels, the achieved contrast upon reconstruction varies. For easier reference and comparison, this paper summarizes the contrast values mentioned above in Table 5.

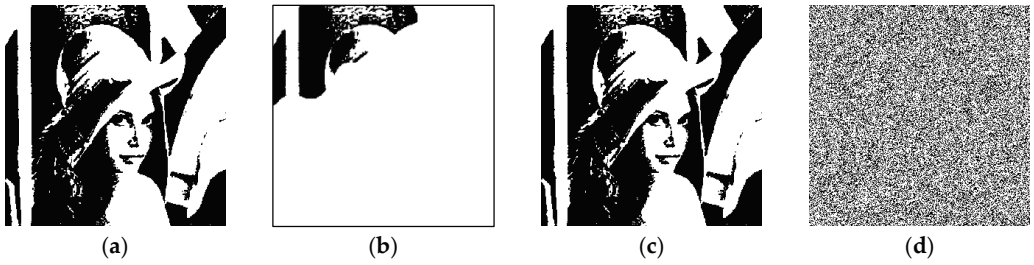


Figure 9. *Cont*.

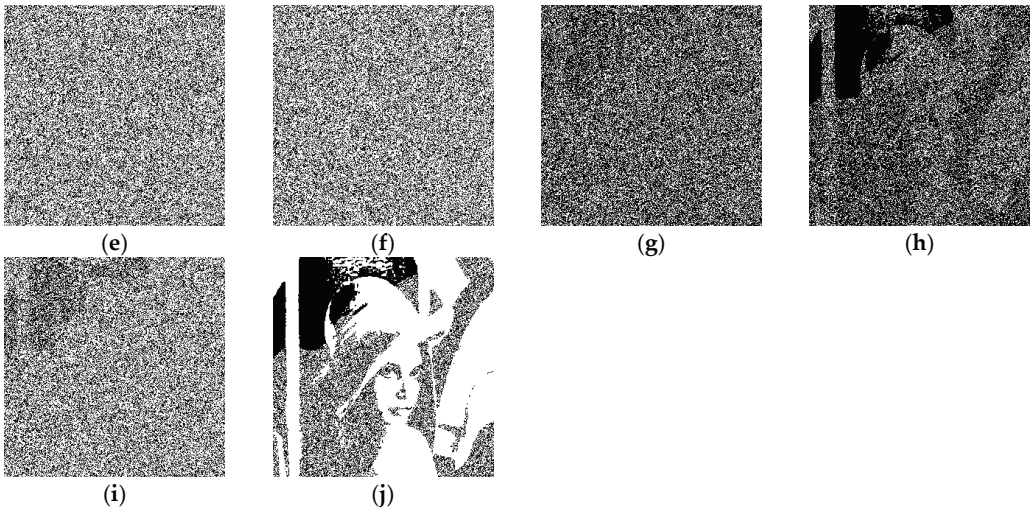


Figure 9. The experimental results of the proposed (2, 3) 2D_RIVCS with two secret levels. (**a**) S. (**b**) Sl_1. (**c**) Sl_2. (**d**) B_1. (**e**) B_2. (**f**) B_3. (**g**) $B_1 \otimes B_2$. (**h**) $B_1 \otimes B_2 \otimes B_3$. (**i**) $B_1 \oplus B_2$. (**j**) $B_1 \oplus B_2 \oplus B_3$.

Table 5. The contrast values of Figures 8 and 9.

		Figure 8		Figure 9	
	t	Sl_1	Sl_2	Sl_1	Sl_2
OR Decryption	2	0.06517	-	0.067229	-
	3	0.35696	0.32209	0.31722	0.15064
XOR Decryption	2	0.11629	-	0.11778	-
	3	0.92647	0.85572	0.76527	0.44327

Figure 10 showcases (2, 4) 2D_RIVCS with three secret levels, as depicted in Figure 10b–d. In this set of cases, the image dimensions are all set to 1000. The encrypted results in Figure 10e–h reveal that the entire image appears more grayscale. For this experiment, a textual image was used as the secret image, and the results display remarkably impressive outcomes. By partitioning secret levels, the encryptor can control how much text the decipherer can view when collecting a certain number of encrypted images. The contrast between the reconstructed image and each secret level is presented in Table 6.

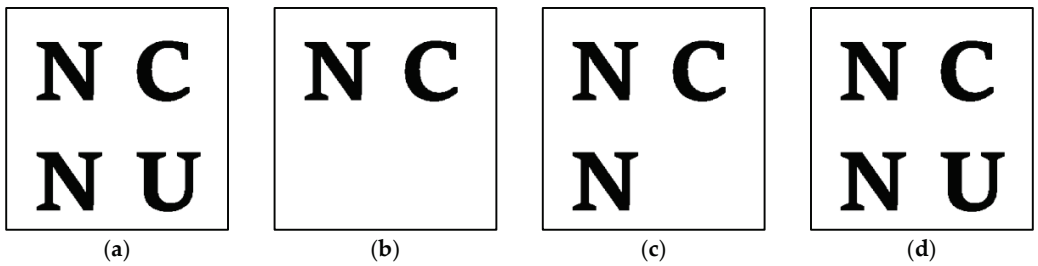


Figure 10. *Cont.*

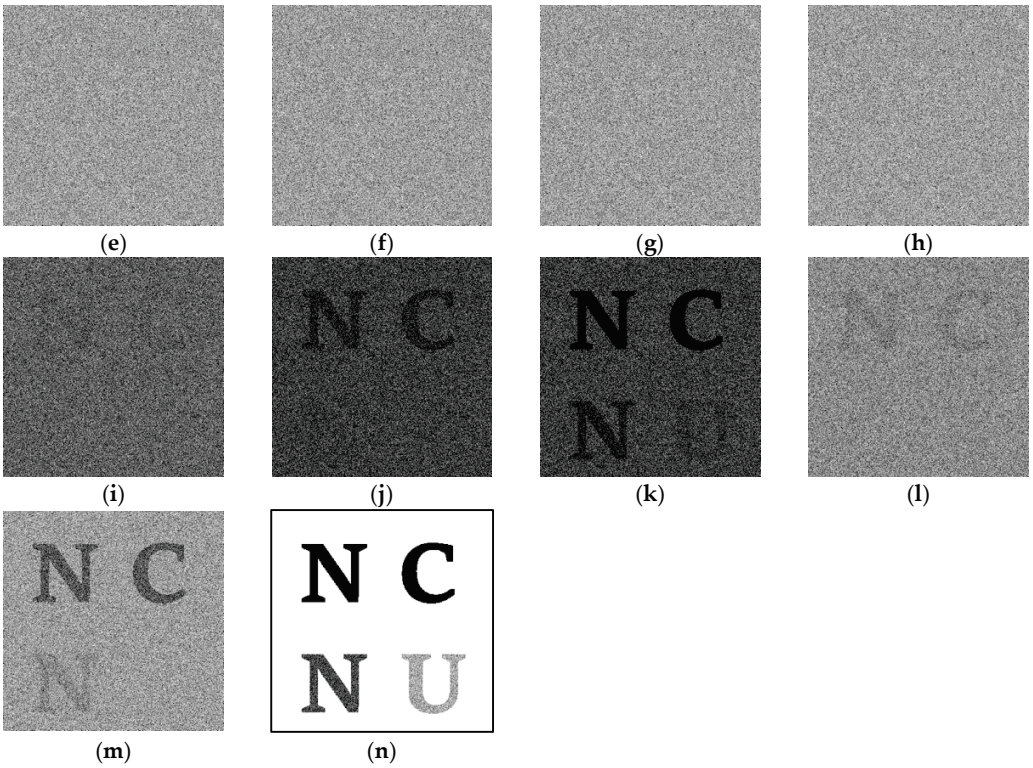


Figure 10. The experimental results of the proposed (2, 4) 2D_RIVCS with three secret levels. (**a**) S. (**b**) Sl_1. (**c**) Sl_2. (**d**) Sl_3. (**e**) B_1. (**f**) B_2. (**g**) B_3. (**h**) B_4. (**i**) $B_1 \otimes B_2$. (**j**) $B_1 \otimes B_2 \otimes B_3$. (**k**) $B_1 \otimes B_2 \otimes B_3 \otimes B_4$. (**l**) $B_1 \oplus B_2$. (**m**) $B_1 \oplus B_2 \oplus B_3$. (**n**) $B_1 \oplus B_2 \oplus B_3 \oplus B_4$.

Table 6. The contrast of Figure 10.

	OR Decryption				XOR Decryption		
t	Sl_1	Sl_2	Sl_3	t	Sl_1	Sl_2	Sl_3
2	0.019801	-	-	2	0.03535	-	-
3	0.087095	0.059633	-	3	0.17898	0.13041	-
4	0.28509	0.21552	0.16656	4	0.96108	0.77239	0.5973

In summary, the experimental results presented in Figures 8–10 highlight the versatility of (k, n) 2D_RIVCS with its ability to handle multiple secret levels. The utilization of textual images as secret data yields impressive outcomes, allowing the encryptor to control the visibility of text based on secret level partitioning. Moreover, the contrast values indicate that XOR decryption outperforms OR decryption in these experimental results.

6. Comparison

In this segment, we will conduct a comparative analysis of our proposed approach in relation to other, previously published research. We proceed with a functional comparison. We conducted a comparative analysis among our proposed scheme and several other related RIVCS schemes: Wang [9], Wang et al. [10], Shyu and Jiang [11], Yang et al. [12], Kumar and Sharma [13] and Yang et al. [14], as shown in Table 7. Our comparison focused on various aspects such as encryption method, expansion, threshold, maximum secret level and decryption method. In can be seen that our scheme employs the use of random grids for encryption, which effectively avoids pixel expansion. Furthermore, our decryption process uses two decryption methods. Judging from the experimental results, when the proposed scheme is restored by OR operation, the results will be better when t is closer to n. Especially when $t = n$, the results are almost better than those in the past literature.

When the XOR operation is used to restore the image, the contrast is much larger than the result of the OR operation, and it is even close to perfect recovery (the contrast α is equal to 1). Therefore, we can know that the proposed scheme performs very well in restoring images. Overall, the proposed novel (k, n) 2D_RIVCS is currently the only known RG-based (k, n) threshold region incremental visual cryptography method with XOR and OR decryption capabilities, with better contrast values in most cases. It demonstrates more practical performance compared to other known RIVCS methods.

Table 7. Comparison.

Property	Scheme						
	[9]	**[10]**	**[11]**	**[12]**	**[13]**	**[14]**	**Ours**
Encryption	Basis matrices	Random grids	Basis matrices	Basis matrices	Random grids	Basis matrices	Random grids
Expansion	Yes	No	Yes	Yes	No	Yes	No
Threshold	$(2, n)$	$(2, 3)$	$(2, n)$	(k, n)	(k, n)	(k, n)	(k, n)
Max secret level	$n-1$	$n-1$	$n-1$	$n-k+1$	$n-k+1$	$n-k+1$	$n-k+1$
Decryption	OR	OR	OR	OR	OR	OR	OR and XOR

7. Conclusions

This paper introduces 2D_RIVCS, a novel RG-based region incrementing visual cryptography scheme with both OR and XOR decryption capabilities. This approach offers two distinct decryption methods to cater to various user scenarios. In situations where users lack specialized equipment, OR decryption can be employed, requiring no additional machinery; it simply involves stacking the encrypted images for restoration. Alternatively, users equipped with computers can opt for XOR decryption, offering a more advanced method that achieves perfect restoration. The versatility of this proposed method extends its application to a wide range of potential use cases.

Note that if we change the range of the number t from $\{k+1, \ldots, n\}$ to $\{k+1, \ldots, n-1\}$ in Step 4 of Algorithm 3, then Algorithms 3 and 4 are still correct and the performance will be improved.

At present, this method is limited to binary images, significantly constraining its applicability. Therefore, the next phase of our research will focus on a more comprehensive investigation of grayscale and color images, with the aspiration that this approach can extend its utility to a diverse range of visual content. The expansion into color imagery represents a significant stride towards broader applicability, making this method more versatile and accommodating a wider array of practical use cases.

In addition, another avenue for future work is to enhance the camouflage capability of this method to ensure that the encrypted image does not appear as arbitrary noise but displays the camouflaged image. Aiming to embed concealed images within encrypted data elevates the security and covert potential of this approach, bolstering its utility for a multitude of applications across domains.

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

Funding: This work was partially supported by the National Science and Technology Council, R.O.C., under grants MOST 111-2115-M-260-001- and NSTC 112-2115-M-260 -001 -MY2.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the referees for their careful reading of the manuscript and fruitful comments.

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

References

1. Naor, M.; Shamir, A. Visual Cryptography. In *Workshop on the Theory and Application of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 1994; pp. 1–12.
2. Kafri, O.; Keren, E. Encryption of pictures and shapes by random grids. *Opt. Lett.* **1987**, *12*, 377–379. [CrossRef] [PubMed]
3. Shyu, S.J. Image encryption by multiple random grids. *Pattern Recognit.* **2009**, *42*, 1582–1596. [CrossRef]
4. Chen, T.H.; Tsao, K.H. Visual secret sharing by random grids revisited. *Pattern Recognit.* **2009**, *42*, 2203–2217. [CrossRef]
5. Chen, T.H.; Tsao, K.H. Threshold visual secret sharing by random grids. *J. Syst. Softw.* **2011**, *84*, 1197–1208. [CrossRef]
6. Lin, Y.R.; Juan, J.S.T. RG-based (*k*, *n*) threshold visual cryptography with abilities of OR and XOR decryption. *Eng. Proc.* **2023**, *55*, 65.
7. Yan, X.; Wang, S.; EI-Latif, A.A.A.; Niu, X. Visual secret sharing based on random grids with abilities OR and XOR lossless recovery. *Multimed. Tools Appl.* **2015**, *74*, 3231–3252. [CrossRef]
8. Wu, X.; Sun, W. Random grid-based visual secret sharing with abilities of OR and XOR decryptions. *J. Vis. Commun. Image Represent.* **2013**, *24*, 48–62. [CrossRef]
9. Wang, R.Z. Region incrementing visual cryptography. *IEEE Signal Process. Lett.* **2009**, *16*, 659–662. [CrossRef]
10. Wang, R.Z.; Lan, Y.C.; Lee, Y.K.; Huang, S.Y.; Shyu, S.J.; Chia, T.L. Incrementing visual cryptography using random grids. *Opt. Commun.* **2010**, *283*, 4242–4249. [CrossRef]
11. Shyu, S.J.; Jiang, H.W. Efficient construction for region incrementing visual cryptography. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 769–777. [CrossRef]
12. Yang, C.N.; Shih, H.W.; Wu, C.C.; Harn, L. *k* Out of *n* Region Incrementing Scheme in Visual Cryptography. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 799–810. [CrossRef]
13. Kumar, S.; Rajendra Kumar, S. Random-grid based region incrementing visual secret sharing. *Fundam. Informaticae* **2015**, *137*, 369–386. [CrossRef]
14. Yang, C.N.; Wu, C.C.; Lin, Y.C. *k* out of *n* region-based progressive visual cryptography. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 252–262. [CrossRef]
15. Huang, S.Y.; Lo, A.H.; Juan, J.S.T. XOR-Based Meaningful (*n*, *n*) Visual Multi-Secrets Sharing Schemes. *Appl. Sci.* **2022**, *12*, 10368. [CrossRef]
16. Chen, Y.H.; Juan, J.S.T. XOR-Based (*n*, *n*) Visual Cryptography Schemes for Grayscale or Color Images with Meaningful Shares. *Appl. Sci.* **2022**, *12*, 10096. [CrossRef]
17. Panchbhai, V.V.; Varade, S.W. Enhanced block based progressive visual secret sharing scheme for grayscale and color image. *Multimed. Tools Appl.* **2023**. [CrossRef]
18. Zhang, L.; Zhang, J.; Sun, J.; Chen, Q. A global progressive image secret sharing scheme under multi-group joint management. *Math. Biosci. Eng.* **2024**, *21*, 1286–1304. [CrossRef]
19. Shyu, S.J. Image encryption by random grids. *Pattern Recognit.* **2007**, *40*, 1014–1031. [CrossRef]
20. Guo, T.; Liu, F.; Wu, C. Threshold visual secret sharing by random grids with improved contrast. *J. Syst. Softw.* **2013**, *86*, 2094–2109. [CrossRef]

symmetry

MDPI

Article

A High-Payload Data Hiding Scheme Based on Absolute Moment Block Truncation Coding for Minimizing Hiding Impact

Chia-Chen Lin [1,*], Bohan Zhang [2], Wei-Liang Tai [3,*], Pei-Feng Shiu [4] and Jinn-Ke Jan [4]

[1] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
[2] Institute of Information Management, National Yang-Ming Chiao Tung University, Hsinchu 300, Taiwan; zhangbohan.mg10@nycu.edu.tw
[3] Bachelor Degree Program of Artificial Intelligence, National Taichung University of Science and Technology, Taichung 404, Taiwan
[4] Department of Computer Science and Engineering, National Chung Hsing University, Taichung 402, Taiwan; d100056005@cs.nchu.edu.tw (P.-F.S.); jkjan@cs.nchu.edu.tw (J.-K.J.)
* Correspondence: ally.cclin@ncut.edu.tw (C.-C.L.); twl@nutc.edu.tw (W.-L.T.)

Abstract: Data hiding encompasses a wide range of applications related to hiding messages in digital images. The visual redundancy of images makes it possible to embed data in the images without attracting attention. Increasing the hiding capacity and decreasing the hiding distortion are prime objectives that data hiding intends to achieve. In this paper, we propose a high-payload data hiding scheme based on absolute moment block truncation coding (AMBTC) to minimize the impact of hiding. A two-level minimum mean square error (MMSE) quantizer generated by AMBTC is used to decrease the distortion associated with hiding. Also, we present a lookup table based on the symmetric property for adaptively hiding secrets in pixels to achieve high hiding capacity. We can embed almost 1.9 bits per pixel (bpp) with a high image quality of an average of 31 dB. Only 5.3% of pixels are changed during the data-hiding process. Compared with other schemes, we can use 1 bpp more relative payload for embedding with the same stego image quality. The experimental results show that the proposed scheme has better hiding performance because it allows a huge amount of secret data to be hidden while maintaining the high visual quality of the stego image.

Keywords: data hiding; AMBTC; high-payload; digital images; minimizing hiding impact

1. Introduction

Cryptography uses mathematical theory to prevent people from gaining access to secret data illegally. However, the secret data can still be threatened by malicious attackers since the meaningless and unintelligible form generated from encryption may attract their attention. To conceal the existence of secret data from the public, data hiding provides a satisfactory solution by making the very existence of the original messages imperceptible. Data hiding [1] is the science of concealed communication, which involves hiding secret data in meaningful cover objects with a slight and imperceptible distortion. It can be used in various applications, such as copyright protection (robust watermarking), secret communication (steganography), and image authentication (fragile watermarking). Various applications have different requirements. For the safety of secret communications, it is important to conceal the existence of the secret data in order to avoid attracting an attacker's attention. People cannot easily distinguish the difference between the meaningful cover object and the corresponding hiding result, i.e., the stego object. To meet the safety requirements, the hiding distortion should be minimized, and the hiding capacity should suffice for embedding the secret data. Intuitively, the goal of data hiding is to design schemes that have high hiding capacity but low distortion introduced by the hiding. Hence, a trade-off must be made between hiding capacity and hiding distortion.

In data hiding, we will assume that the cover image is an 8-bit grayscale digital image, which is the set of all possible pixel values in the range [0, 255]. The most common way used in data hiding is the least significant bit (LSB) of pixel values. The LSB of pixel value i can be computed as i mod 2. The LSB embedding operation is flipping the LSB of the pixel value. Low hiding distortion means that there is a low probability of detecting the existence of a secret message. Mielikainen [2] proposed a modification to the LSB matching that involves designing a binary function of two cover pixels to the desired value, which allows hiding the same payload as LSB matching but with fewer changes to the cover image. Also, van Dijk et al. [3] developed another type of ± 1 steganography for improving embedding efficiency in which two-dimensional codes were proposed to embed a 5-ary message symbol in a group of two pixels by modifying, at most, one pixel in the group by one. In addition, this can be generalized to n-dimensional codes [4,5] that allow $\log_2(2n + 1)$ secret bits to be embedded in n cover pixels by modifying, at most, one pixel in the group by one. Fridrich et al. [6] introduced a wet paper coding mechanism in which the sender embeds messages into content without sharing selection rules with the recipient. In addition, wet paper codes have been combined with most steganographic schemes to improve their embedding efficiencies [7].

However, the LSB-based data-hiding schemes have a low hiding capacity. To improve the hiding capacity, Lin et al. [8] proposed a high-payload, reversible data hiding scheme that is based on the absolute moment block truncation coding (AMBTC) compression domain. They presented four disjointed sets for embeddable blocks to embed data using different combinations of the mean value and the standard deviation. Malik et al. [9] modified the AMBTC compression technique for hiding secret data by first applying the original AMBTC technique, and they identified the smooth and complex blocks using a threshold value. They converted the one-bit plane into a two-bit plane and replaced all bits of the bit plane with the secret bits to obtain better image quality and high capacity. However, this approach permanently destroys the original AMBTC code and requires overhead information. Chen et al. [10] proposed an image authentication scheme for AMBTC of a compressed image using turtle shell-based data hiding.

In 2019, Yu et al. [11] proposed a hybrid data-hiding method for AMBTC compressed images, which combines a turtle-shell reference matrix and (7, 4) Hamming code to enhance the hiding capacity for compressed codes. Lin et al. [12] provided a reversible data-hiding method that uses adaptive block truncation coding based on an edge-based quantization approach. They utilized a Canny edge detector to obtain edge-blocks and non-edge-blocks, and they applied zero-point fixed histogram shifting to embed the secret information into the compressed code. In 2020, Yu et al. [13] proposed an adaptive image steganography method combining matrix coding. They constructed a reference data set by classifying all possible 7-bit binary number combinations and adaptively embedded 3-bit data by choosing a suitable alternative from the reference data. Their approach provided better results than the other existing matrix coding-based data-hiding schemes.

To minimize the risk of hiding data, we aim to decrease the hiding distortion and increase the hiding capacity. In this paper, we propose a high-payload data-hiding procedure based on AMBTC to improve the embedding efficiency further. Our proposed scheme embeds secret data by modifying the quantization level according to the pre-defined lookup table. Moreover, previous AMBTC-based schemes may have suffered from the problem of having a high quantization level lower or equal to a low quantization level caused by hiding the data. We propose an adaptive embedding strategy to solve the above issues and achieve high hiding capacity. We demonstrated that the proposed scheme has better hiding performance in image quality and hiding capacity.

To make this paper self-contained, in Section 2, we review the absolute moment block truncation coding (AMBTC) algorithm for data hiding. High-payload data hiding based on AMBTC is described in Section 3. The experimental results and their analyses are presented in Section 4. Also, in Section 4, the performance is compared to the performances

of existing data-hiding schemes. The brief contribution is concluded in Section 5, where we also outline the future research directions.

2. AMBTC

The absolute moment block truncation coding (AMBTC) proposed by Lema and Mitchell [14] is a type of lossy image compression technique. It is a variation of block truncation coding (BTC), but it is simpler in practical implementation. AMBTC preserved the first absolute moment and proposed a two-level, non-parametric minimum mean square error quantizer where the threshold is fixed to the sample mean. A lower mean square error yield by AMBTC is used in our proposed data hiding to minimize the impact of secret data hiding.

In AMBTC, an image is divided into non-overlapping blocks of $n \times n$ pixels, and x_i is the gray level of a pixel in the block where $1 \leq i \leq n^2$. Each block is quantized, and the corresponding resulting block has the same sample mean and the same sample first absolute central moment as each original block. For each block, the sample mean, $\overline{\eta}$, is the decision threshold of the quantizer, which is calculated as

$$\overline{\eta} = \frac{1}{n^2} \sum_{i=1}^{n^2} x_i. \tag{1}$$

In AMBTC encoding, each block is encoded as (L, H, BM), where (L, H) is a two-level MMSE (minimum mean square error) quantizer, and BM is a bitmap to denote the thresholding result. The bitmap BM is presented as:

$$BM_i = \begin{cases} 1, & \text{if } x_i \geq \overline{\eta}, \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where x_i is encoded as 1 when x_i is greater than or equal to the threshold and 0 otherwise. The two-level MMSE quantizers are computed as shown below:

$$\begin{aligned} L &= \tfrac{1}{n^2-q} \sum_{x_i < \overline{\eta}} x_i, \\ H &= \tfrac{1}{q} \sum_{x_i \geq \overline{\eta}} x_i, \end{aligned} \tag{3}$$

where q is the number of pixels above the threshold. Note that L and H are estimated conditional means given that x_i is less than or greater than $\overline{\eta}$, respectively.

In AMBTC decoding, each pixel x_i' of each block is decoded according to the two-level MMSE quantizer and BM:

$$x_i' = \begin{cases} L, & \text{if } x_i = 0, \\ H, & \text{otherwise.} \end{cases} \tag{4}$$

3. Our Proposed DH Method Based on HP-AMBTC

To improve the hiding capacity and decrease the hiding distortion, the two-level MMSE quantizer used by AMBTC is used in our proposed data hiding to minimize the hiding impact. In addition, we designed a lookup table for adaptively hiding secret information in images. The details of the proposed scheme are described as follows.

3.1. Data Embedding

Assume that the cover image is an 8-bit grayscale digital image, which is the set of all possible pixel values in the range [0, 255]. The cover image is divided into non-overlapping blocks of 4×4 pixels, and let $\{x_1, x_2, \ldots, x_{16}\}$ be the pixels in a block read in a raster scan where $x_i \in [0, 255]$.

Step 1. For each block, we use AMBTC to encode it to generate the AMBTC compression code (L, H, BM).

Step 2. Reconstruct the block using Equation (4) to obtain the reconstructed pixels $\{x'_1, x'_2, \ldots, x'_{16}\}$ read in a raster scan.

Step 3. For each reconstructed block, we mark x'_1 and x'_{16} as the non-embeddable pixels and modify x'_{16} as

$$x'_{16} = \begin{cases} L \text{ if } x'_1 = H \\ H \text{ if } x'_1 = L \end{cases}. \quad (5)$$

Step 4. Embed secret data into embeddable pixels $\{x'_2, x'_3, \ldots, x'_{15}\}$ as

$$y_i = x'_i + mv,$$

where y_i is the stego pixel, and mv is defined as follows:

Case 1: $|H-L| = 0$

Secret Data	***H/L mv***
00	−1
01	0
10	+1
11	+2

Case 2: $|H-L| = 1$

Secret Data	***L mv***	***H mv***
0	0	0
1	−1	+1

Case 3: $2 \leq |H-L| < 5$

Secret Data	***L mv***	***H mv***
00	−1	−1
01	0	0
10	−2	+1
11	−3	+2

Case 4: $|H-L| = 5$

Secret Data	***L mv***	***H mv***
1111	−1	+3
0000	−2	−2
00	−1	−1
01	0	0
10	+1	+1
11	+2	+2

Case 5: $|H-L| > 5$

Secret Data	*H/L mv*
1111	+3
0000	−2
00	−1
01	0
10	+1
11	+2

Note that the lookup table, which is our embedding and extraction rule, should be previously shared between the two parties using a secure channel. Taking Figure 1 as the data hiding to conceal 34 secret bits, for example, after decoding the AMBTC encoded code (90, 150, *BM*) to obtain a reconstructed image block, we define x'_1 and x'_{16} as the non-embeddable pixels. According to $|H-L| = 60 > 5$ and secret data, we choose the Case 5 lookup table to adaptively increase the embeddable pixels by the corresponding modified value, *mv*. Note that the stego image is also an 8-bit grayscale digital image rather than a compressed image or a decompressed image. From Figure 1, we can see that the stego image block is very similar to the cover image block. After hiding 34 bits of secret data, the hiding distortion is very low since we used a two-level MMSE quantizer to minimize the impact of hiding.

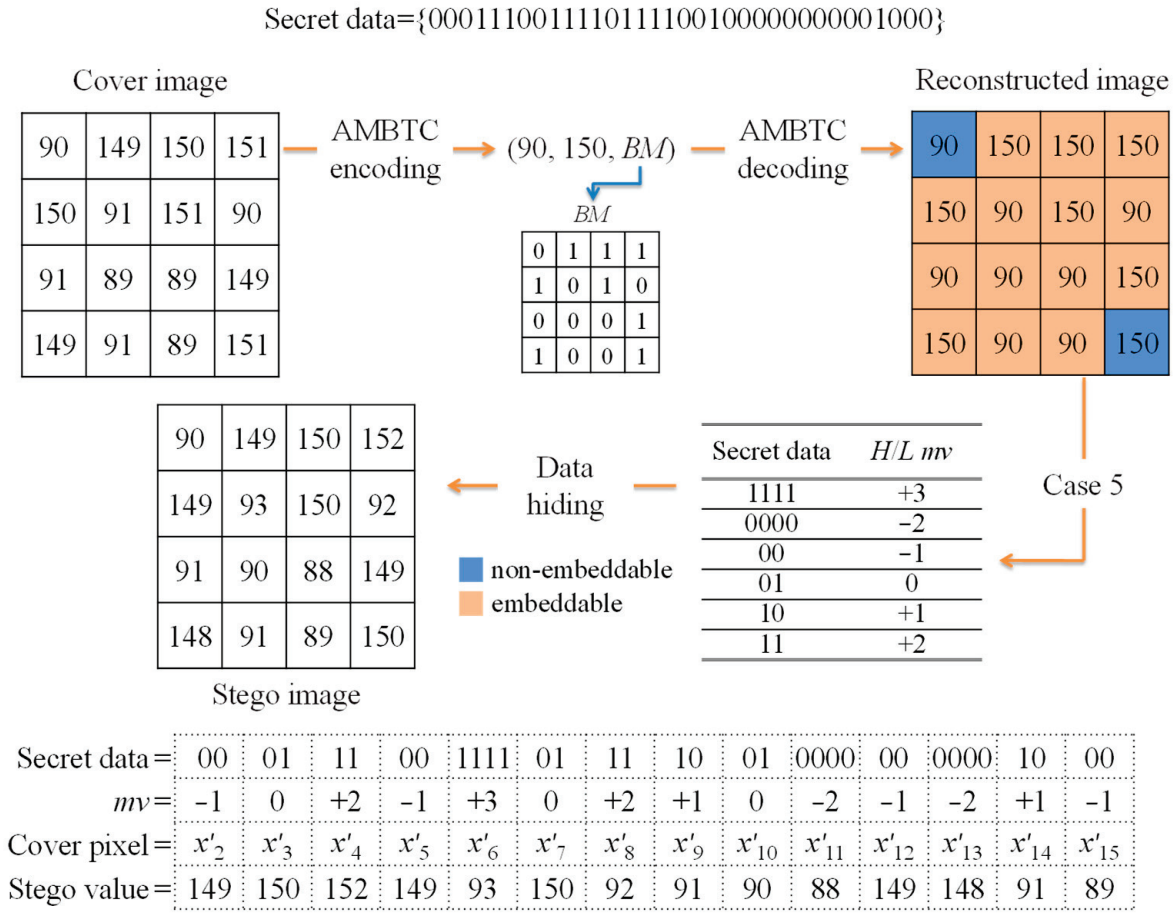


Figure 1. Example of data hiding.

3.2. Data Extraction

This process extracts the secret data from the 8-bit grayscale stego image. Assume that the grayscale stego image is divided into non-overlapping blocks of 4 × 4 pixels,

and let $\{y_1, y_2, \ldots, y_{16}\}$ be the pixels in a stego image block and read in raster scan where $y_i \in [0, 255]$.

Step 1. For each block, we mark y_1 and y_{16} as the non-embeddable pixels. If $y_1 \geq y_{16}$, it indicates y_1 as H and y_{16} as L, respectively. Otherwise, y_1 is L and y_{16} is H, respectively.

Step 2. For each pixel of embeddable pixels $\{y_2, y_3, \ldots, y_{15}\}$, we extract the secret data according to the absolute difference of H and L and the lookup table defined below:

Case 1: $|H-L| = 0$

Stego Value	**Secret Data**
$(H-1)/(L-1)$	00
H/L	01
$(H+1)/(L+1)$	10
$(H+2)/(L+2)$	11

Case 2: $|H-L| = 1$

Stego Value	**Secret Data**
H/L	0
$(H+1)/(L-1)$	1

Case 3: $2 \leq |H-L| < 5$

Stego Value	**Secret Data**
$(H-1)/(L-1)$	00
H/L	01
$(H+1)/(L-2)$	10
$(H+2)/(L-3)$	11

Case 4: $|H-L| = 5$

Stego Value	**Secret Data**
$(H+3)/(L-3)$	1111
$(H-2)/(L-2)$	0000
$(H-1)/(L-1)$	00
H/L	01
$(H+1)/(L+1)$	10
$(H+2)/(L+2)$	11

Case 5: $|H-L| > 5$

Stego Value	**Secret Data**
$(H+3)/(L+3)$	1111
$(H-2)/(L-2)$	0000
$(H-1)/(L-1)$	00
H/L	01
$(H+1)/(L+1)$	10
$(H+2)/(L+2)$	11

Figure 2 illustrates the data extraction example. We define y_1 and y_{16} as the non-embeddable pixels, and we regard y_1 as L and y_{16} as H due to $y_1 < y_{16}$. According to $|H-L| = 60 > 5$, we choose the Case 5 lookup table to extract the secret data. Finally,

34-bits of secret data can be extracted after scanning 14 pixels. Moreover, the AMBTC decoded image block can be almost restored according to the corresponding *L*s and *H*s except for the first and the last pixels.

We said the AMBTC encoded image block could be almost restored because the first and the sixteenth pixels needed to be modified and then served as the indicators for data extraction later. If these two pixels in a block are the same as the original ones after the data hiding, the original AMBTC-encoded image blocks can be completely restored. Otherwise, only fourteen pixels in a block can be restored to the original AMBTC compression codes after data extraction.

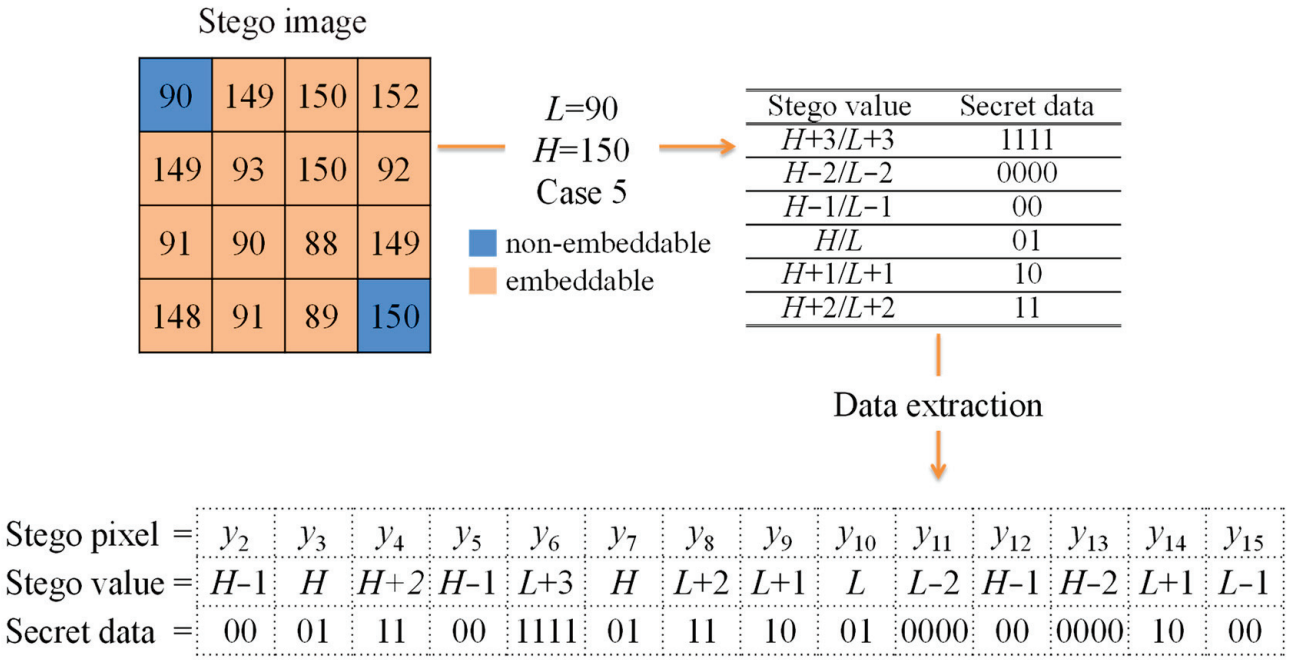


Figure 2. Example of data extraction.

4. Experimental Results

In this section, a series of analyses and simulation results are demonstrated to prove the performance of the proposed scheme. In addition, comparisons among [15–20] also are presented in this section.

4.1. Capacity Versus Distortion Performance

To test our performance on data hiding and image quality, all experiments were performed with ten commonly used grayscale images sized 512 × 512, i.e., "Lena," "Airplane," "Baboon," "Barbara," "Peppers," "Boat," "F-16," "Boat," "House," "Houses," "Zelda," and "Gold Hill." Six 512 × 512 selected grayscale images are shown in Figure 3 to demonstrate partial test images.

We use the peak signal-to-noise ratio (*PSNR*), which is the most common image quality criterion, to evaluate the visual quality of the reconstructed images. The *PSNR* (unit: dB) is defined as:

$$PSNR = 10 \times \log_{10} \frac{255^2}{MSE}, \tag{6}$$

where *MSE* is the mean square error defined by:

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(x_{i,j} - x'_{i,j} \right)^2, \tag{7}$$

where $x_{i,j}$ and $x'_{i,j}$ present the pixel values of the original image and the modified image, respectively, and $M \times N$ indicates the size of the image. Moreover, structural similarity

(*SSIM*) is also used to evaluate the similarity between the stego image and the original cover image. The *SSIM* is defined as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{8}$$

where μ_x and μ_y are the mean values of the pixel values of image x and image y, respectively. σ_x and σ_y are the standard deviations of the pixel values of image x and image y, respectively. $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$, and $k_1 = 0.01$ and $k_2 = 0.03$. In general, $SSIM(x, y)$ ranges between 1 and -1. When two images are identical, the corresponding *SSIM* value will be 1.

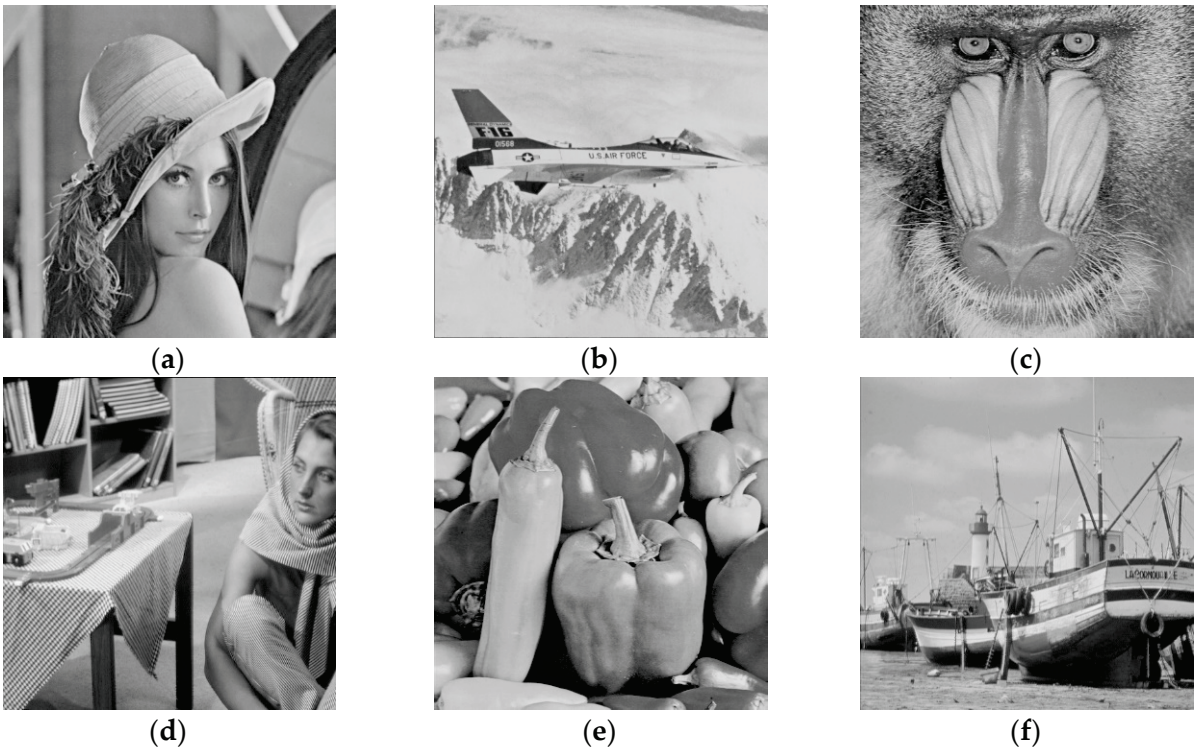


Figure 3. Greyscale test image. (**a**) Lena; (**b**) F-16; (**c**) Baboon; (**d**) Barbara; (**e**) Peppers; and (**f**) Boat.

Figure 4 presents the relationship between the capacity and the corresponding image distortion for the six test images. It is obvious that the achievable capacity of our proposed HP-AMBTC-based DH scheme depends on the features of the image. It indicates that the smooth images provide a higher capacity at the same embedding distortion; this made it evident that images with high correlation offer a larger hiding capacity than images with low correlation. To further demonstrate the performance of our scheme, Table 1 presents the performance of *PSNR*, maximum hiding capacity, and its corresponding bit per pixel (bpp) with our proposed HP-AMBTC-based DH scheme. Here, we can see that the average bpp can be rounded to 2 bpp while maintaining 31.96 dB for the average image quality of the stego image with our proposed HP-AMBTC-based DH scheme when "Baboon" is excluded. Even if "Baboon" is included, our average *PSNR* still remains at 31.340 dB, and previous works [21,22] have confirmed that the visual quality of AMBTC is acceptable, although the value of its *PSNR* value is less than 30 dB.

To further demonstrate the competitive performance of our proposed HP-AMBTC-based DH scheme in Table 2, we compare our scheme with six other existing BTC or AMBTC-based data hiding schemes [15–20]. It is noted that all data of the other six schemes are cited from the original experimental data presented by the original research teams. Based on the data presented in Table 2, it is obvious that the average maximum hiding

capacity of the test images delivered by the proposed scheme is 1.90 bpp, which is relatively higher than that offered by the other six data-hiding schemes. Our scheme embeds data into the high mean values or the low mean values derived from AMBTC. The maximum change to the mean value is within the range [−2, 3], which means that we can embed 1.9 bpp with an embedding efficiency of almost 2 bits per change. Moreover, there are differences between the original pixel value and the mean value. Embedding data to the mean value makes it possible to get close to the original pixel value, which greatly decreases the embedding distortions; this means that the stego pixel value is nearly the same as the original pixel value. As shown in Figure 5, our scheme achieves the highest average hiding capacity while maintaining better image quality at 32.26 dB. However, a lookup table is required for data hiding and extraction in our scheme; either the corresponding data hiding rules or data extraction rules can be easily programmed to replace the lookup table.

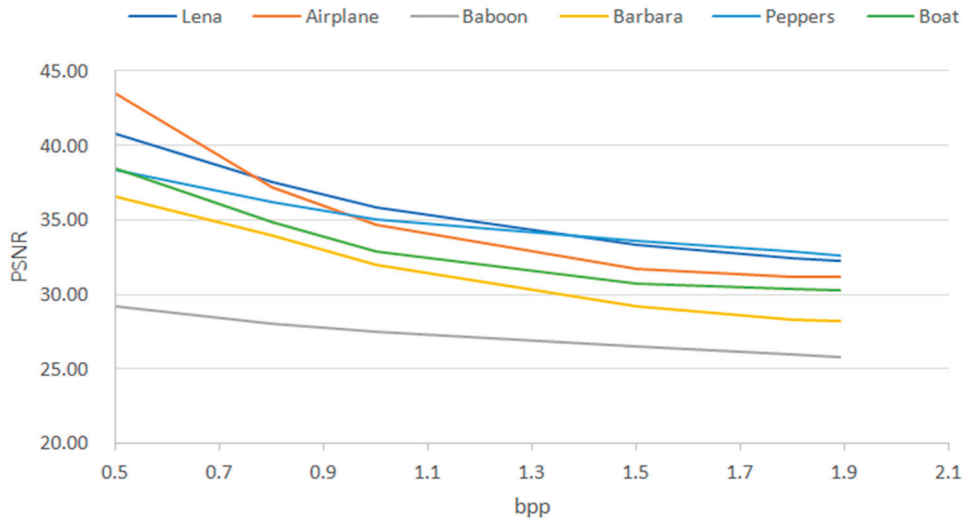


Figure 4. PSNR versus payload size for test images.

Table 1. Performance on image quality (*PSNR*), max. hiding capacity and bpp with ten 512 × 512 test images.

Criteria / Test Images	Hiding Capacity (bits)	bpp	*PSNR*
			HP-AMBTC
Lena	494,464	1.9	32.253
Baboon	509,764	1.9	25.755
Peppers	502,476	1.9	32.624
F-16	476,036	1.8	31.144
Boat	505,308	1.9	30.240
House	457,444	1.7	35.914
Houses	480,850	1.8	29.964
Zelda	499,818	1.9	35.667
Gold Hill	509,688	1.9	31.660
Barbara	500,334	1.9	28.172
Average	493,618	1.9	31.334

Table 2. Performance comparison with our scheme and six existing BTC/AMBTC-based data-hiding schemes [15–20] (Unit: dB for *PSNR*, and bit for bpp).

Methods	Test Images	Boats	Goldhill	F-16	Lena	Peppers	Zelda	Average
Ou and Sun [15]	bpp	0.79	0.81	0.9	0.92	0.93	0.85	0.87
	PSNR	29.57	29.12	30.75	30.69	31.43	29.93	30.25
Huang et al. [16]	bpp	0.99	1.02	1.09	1.12	1.12	1.05	1.07
	PSNR	29.30	29.41	30.34	30.44	31.06	29.79	30.06
Hong [17]	bpp	0.9	0.92	1.01	1.04	1.05	0.96	0.98
	PSNR	29.56	29.11	30.74	30.68	31.41	29.92	30.24
Malik et al. [18]	bpp	1.52	-	1.52	1.52	1.52	-	1.52
	PSNR	31.09	-	31.90	33.10	33.30	-	32.35
Yeh et al. [19]	bpp	1.9	-	1.3	1.67	1.83	-	1.675
	PSNR	31.04	-	32.09	33.06	33.01	-	32.30
Kim et al. [20]	bpp	1.02	1.03	1.11	1.14	1.13	1.08	1.09
	PSNR	30.39	29.65	31.86	29.89	28.57	31.02	30.24
Our proposed scheme	bpp	1.93	1.95	1.82	1.89	1.92	1.91	1.90
	PSNR	30.24	31.66	31.14	32.25	32.62	35.67	32.26

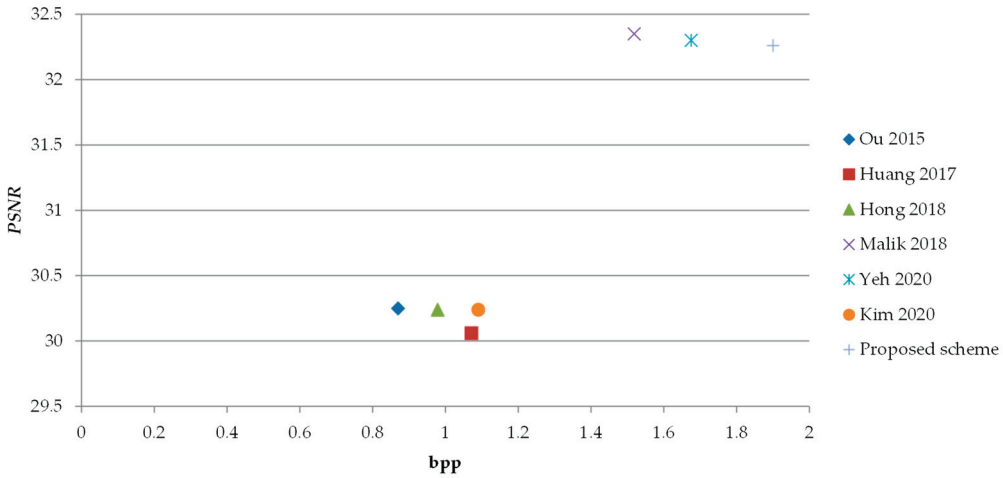


Figure 5. Average performance comparison of schemes [15–20].

4.2. Security Analysis

Since the cover image must be modified to conceal secret data with data-hiding strategies, whether the pixel distribution of the stego image remains an unidentified pattern similar to that of the cover image is crucial when evaluating the security of a data-hiding scheme. To demonstrate the security of our proposed HP-AMBTC-based DH scheme, several metrics: *SSIM* (structural similarity), number of changing pixel rate (*NPCR*), unified averaged changed intensity (*UACI*), and peak signal-to-noise ratio (*PSNR*) are used to analyze the stego images derived from our proposed scheme.

PSNR has been defined in Equation (6), and *SSIM* has been defined in Equation (8). As for *NPCR* and *UACI*, the corresponding detailed definitions are given in the following equations:

$$D(i',j') = \begin{cases} 0, & if\ O(i',j') = O^*(i',j'), \\ 1, & if\ O(i',j') \neq O^*(i',j'), \end{cases} \tag{9}$$

$$NPCR = \frac{1}{M \times N} \sum_{i',j'} D(i',j') \times 100\%, \tag{10}$$

$$UACI = \frac{1}{M \times N} \sum_{i',j'} \frac{O(i',j') - O^*(i',j')}{255} \times 100\%. \quad (11)$$

The ranges of *NPCR* and *UACI* are both [0, 1], and a higher value indicates a higher difference between the original image and the stego image. At the same time, the value of *UACI* is not as obvious as that of *NPCR*.

The security analyses for our proposed HP-AMBTC-based DH scheme are demonstrated in Table 3. Table 3 shows the similarity of structure between the original image and the stego image, which remains at 0.905, which is very close to 1. In addition, *NPCR* and *UACI* indicate that the pixel difference between the original image and the stego image is quite slight. Take *NPCR*, for example; only 5.3% of pixels between the original image and stego images are different. With the results demonstrated in Table 3, it is concluded that our proposed HP-AMBTC-based DH scheme can guarantee the similarity between the original image and the stego image. In other words, the security of the hidden secret can be guaranteed.

Table 3. Security analyses with six different test images.

Metrics \ Test Images	Baboon	F-16	Barbara	Boat	House	Peppers	Average
PSNR	25.755 dB	31.144 dB	28.172 dB	30.240 dB	35.914 dB	32.624 dB	30.642 dB
SSIM	0.8515	0.9260	0.8978	0.8958	0.9509	0.9104	0.905
NPCR	0.0592	0.0492	0.0556	0.0556	0.0437	0.0539	0.053
UACI	0.0002	0.0001	0.0001	0.0002	0.0001	0.0002	0.0001

5. Conclusions

To mitigate the impact of concealed data while enhancing the hiding capacity for AMBTC compressed images, this paper introduces a high-payload data-hiding scheme based on AMBTC. The proposed scheme leverages a two-level MMSE quantizer generated by AMBTC and our specially designed data-hiding strategies tailored for different quantizer distortions. The collaboration between the quantizer and our strategies aims to strike a balance between induced distortion and hiding capacity. Experimental results validate that our scheme surpasses six existing BTC or AMBTC-based data-hiding schemes in terms of both hiding capacity and visual quality. We can embed almost 1.9 bits per pixel with a high image quality of an average of 31 dB. The data hiding process only modifies 5.3% pixels of the original image on average. The similarity of structure between the original image and the stego image is 0.905, which means that the stego pixel image is nearly the same as the original image. Additionally, security analyses affirm that the increased capacity does not compromise the similarity in either structure or pixel distribution. Based on these findings, future efforts will explore incorporating integrity authentication codes during data-hiding operations. This enhancement will allow the receiver(s) to verify the integrity of received stego images before extracting hidden data, thereby bolstering the usability of the extracted confidential data.

Author Contributions: Conceptualization, C.-C.L. and J.-K.J.; formal analysis, C.-C.L.; methodology, C.-C.L. and J.-K.J.; validation, B.Z. and P.-F.S.; writing—original draft, W.-L.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science and Technology Council: MOST 111-2410-H-167-005-MY2.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://sipi.usc.edu/database/.

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

References

1. Wu, M.; Liu, B. Data hiding in image and video. I. Fundamental issues and solutions. *IEEE Trans. Image Process.* **2003**, *12*, 685–695. [PubMed]
2. Mielikainen, J. LSB matching revisited. *IEEE Signal Process. Lett.* **2006**, *13*, 285–287. [CrossRef]
3. van Dijk, M.; Willems, F. Embedding information in grayscale images. In Proceedings of the 22nd Symposium on Information and Communication Theory in the Benelux, Enschede, The Netherlands, 15–16 May 2001; pp. 147–154.
4. Zhang, X.P.; Wang, S.Z. Efficient steganographic embedding by exploiting modification direction. *IEEE Commun. Lett.* **2006**, *10*, 781–783. [CrossRef]
5. Fridrich, J.; Lisoněk, P. Grid colorings in steganography. *IEEE Trans. Inf. Theory* **2007**, *53*, 1547–1549. [CrossRef]
6. Fridrich, J.; Goljan, M.; Lisoněk, P.; Soukal, D. Writing on wet paper. *IEEE Trans. Signal Process.* **2005**, *53*, 3923–3935. [CrossRef]
7. Fridrich, J.; Goljan, M.; Soukal, D. Wet paper codes with improved embedding efficiency. *IEEE Trans. Inf. Forensics Secur.* **2006**, *1*, 102–110. [CrossRef]
8. Lin, C.C.; Liu, X.L.; Tai, W.L.; Yuan, S.M. A novel reversible data hiding scheme based on AMBTC compression technique. *Multimed. Tools Appl.* **2015**, *74*, 3823–3842. [CrossRef]
9. Malik, A.; Sikka, G.; Verma, H.K. A high payload data hiding scheme based on modified AMBTC technique. *Multimed. Tools Appl.* **2017**, *76*, 14151–14167. [CrossRef]
10. Chen, C.C.; Chang, C.C.; Lin, C.C.; Su, G.D. TSIA: A novel image authentication scheme for AMBTC-based compressed images using turtle shell based reference matrix. *IEEE Access* **2019**, *7*, 149515–149526. [CrossRef]
11. Yu, Z.; Lin, C.C.; Chang, C.C.; Su, G.D. HBF-DH: An enhanced payload hybrid data hiding method based on a hybrid strategy and block features. *IEEE Access* **2019**, *7*, 148439–148452. [CrossRef]
12. Lin, C.C.; Chang, C.C.; Wang, Z.M. Reversible data hiding scheme using adaptive block truncation coding based on an edge-based quantization approach. *Symmetry* **2019**, *11*, 765. [CrossRef]
13. Yu, Z.; Lin, C.C.; Chang, C.C. ABMC-DH: An Adaptive Bit-Plane Data Hiding Method Based on Matrix Coding. *IEEE Access* **2020**, *8*, 27634–27648. [CrossRef]
14. Lema, M.; Mitchell, O. Absolute moment block truncation coding and its application to color images. *IEEE Trans. Commun.* **1984**, *32*, 1148–1157. [CrossRef]
15. Ou, D.; Sun, W. High payload image steganography with minimum distortion based on absolute moment block truncation coding. *Multimed. Tools Appl.* **2015**, *74*, 9117–9139. [CrossRef]
16. Huang, Y.H.; Chang, C.C.; Chen, Y.H. Hybrid secret hiding schemes based on absolute moment block truncation coding. *Multimed. Tools Appl.* **2017**, *76*, 6159–6174. [CrossRef]
17. Hong, W. Efficient data hiding based on block truncation coding using pixel pair matching technique. *Symmetry* **2018**, *10*, 36. [CrossRef]
18. Malik, A.; Sikka, G.; Verma, H.K. An AMBTC compression based data hiding scheme using pixel value adjusting strategy. *Multidimens. Syst. Signal Process.* **2018**, *29*, 1801–1818. [CrossRef]
19. Yeh, J.Y.; Chen, C.C.; Liu, P.L.; Huang, Y.H. High-payload data-hiding method for AMBTC decompressed images. *Entropy* **2020**, *22*, 145. [CrossRef]
20. Kim, C.; Yang, C.N.; Leng, L. High-capacity data hiding for ABTC-EQ based compressed image. *Electronics* **2020**, *9*, 644. [CrossRef]
21. Bai, J.; Chang, C.C. A high payload steganographic scheme for compressed images with hamming code. *Int. J. Netw. Secur.* **2016**, *18*, 1122–1129.
22. Hong, W.; Chen, T.S.; Yin, Z.; Luo, B.; Ma, Y. Data hiding in AMBTC images using quantization level modification and perturbation technique. *Multimed. Tools Appl.* **2017**, *76*, 3761–3782. [CrossRef]

symmetry

MDPI

Article

On the Design of Multi-Party Reversible Data Hiding over Ciphered Overexposed Images

Bing Chen, Ranran Yang, Wanhan Fang, Xiuye Zhan and Jun Cai *

School of Cyber Security, Guangdong Polytechnic Normal University, Guangzhou 510665, China; chenbing@gpnu.edu.cn (B.C.); yangranran@gpnu.edu.cn (R.Y.); whfang@gpnu.edu.cn (W.F.); xyzhan@gpnu.edu.cn (X.Z.)
* Correspondence: caijun@gpnu.edu.cn

Abstract: Multi-party reversible data hiding over ciphered images (MRDH-CI) has high restorability since the image is split into multiple ciphered images by secret sharing. However, the MRDH-CI methods either fail to produce satisfied results, or only work well for conventional images. This paper introduces a multi-party reversible data-hiding approach over ciphered overexposed images. First, the pixels of the overexposed images are decomposed into two parts, each of which can be used for secret sharing. Then, the decomposed overexposed images are converted into multiple ciphered overexposed images by using a modified secret sharing method, in which the differences of the ciphered overexposed images are retained. The symmetry of the difference retaining makes the secret data conceal within the ciphered overexposed images such that the marked ciphered overexposed images can be created. Finally, by collecting sufficient marked ciphered overexposed images, it is possible to symmetrically reconstruct the concealed data and primitive overexposed image. Experimental results illustrate that the presented method can efficiently deal with overexposed images while maintaining a low computational overhead.

Keywords: reversible data hiding; multi-party data hiding; ciphered domain; overexposed images

1. Introduction

In the era of digital information transmission, privacy and security issues are becoming increasingly prominent in the fields of information processing and transmission [1,2]. Confronted with the challenge of how to effectively hide sensitive information, reversible data hiding (RDH) has garnered significant interest. The feature of this technique is that the secret data are concealed in a cover, and the cover can be recovered nondestructively after the secret data are extracted. RDH has shown broad application prospects in many fields, including but not limited to medical image transmission [3], legal evidence [4], and military communication [5]. Existing RDH schemes are classified as difference expansion [6–8], histogram shifting [9,10], pixel value ordering [11,12], prediction error expansion [13–15], etc. However, traditional RDH schemes tend to embed data in plaintext images, which can be easily detected by unauthorized users, thus reducing covertness and security. To overcome these limitations, reversible data hiding over ciphered images (RDH-CI) has gained prominence during the last few yeas.

RDH-CI enhances the security of secret data by employing encryption algorithms [16], enabling more effective protection of privacy content during the transmission of sensitive data. In [17], Wang et al. introduced an adaptive most significant bit (MSB) prediction-based method for RDH-CI, in which the pixel correlation is remained by block permutation encryption. With the help of the symmetry of the block permutation, the embedding room can be created by MSB prediction. In the method of Gao [18], the cover image underwent encryption by block substitution and bitstream cipher. The ciphered image is then compressed on the basis of adaptive block coding; therefore, the room used for embedding data is vacated. In [19], Sui et al. used a pseudo-random matrix-based symmetric encryption

to encrypt a primitive image. The receiver uses the same key matrix to restore the primitive image due to the symmetry of the encryption algorithm. In addition, MSB and the remaining bits of the primitive image are combined to generate hiding room by employing the property of image redundancy. In [20], bitstreams are initially encrypted using the JPEG encryption algorithm. Subsequently, a combination of Huffman code mapping and histogram shifting is employed to conceal secret data. Finally, the receiver extracts the concealed secret data from marked JPEG bitstreams, achieving perfect reconstruction of the primitive bitstreams. In Ge's approach [21], chaotic sequences are generated using chaotic mapping, which are utilized to perform inter-block scrambled operations, and finally secondary encryption is performed by stream cipher to improve security. In the data embedding stage, the detection of all bit planes is achieved by recording smooth and rough bit planes, which yields a high embedding capability.

The homomorphism-based RDH-CI is also presented, which is particularly prominent in scenarios where computations need to be conducted while maintaining data encryption. In Wu's scheme [22], the Pallier algorithm is used to encrypt the preprocessed images, which can realize the extraction of data in both plaintext and ciphertext domains due to the homomorphism of the Paillier algorithm. In addition, Zheng et al. [23] proposed a method to generate a ciphered cover using homomorphic encryption, which enables the data hider to conceal secret data within a ciphered cover by establishing a secret bit mapping and lossless modification. The receiver can successfully retrieve the hidden data from the ciphered domain. Ke et al. [24] proposed a fully homomorphic cryptographic encapsulation differential extension scheme to achieve ciphertext control for homomorphic encryption. By introducing a key-switching technique, the data extraction process can be efficiently achieved by directly retrieving data from the ciphered domain. However, an obvious disadvantage of homomorphic encryption is its high computational overhead, which may lead to performance decline during large-scale data processing.

The mentioned RDH-CI methods are built on a data hider, and when the data hider is compromised, the recovery of the primitive image cannot be realized. To improve the restorability, there has been a rise in the introduction of multi-party reversible data hiding over ciphered images (MRDH-CI) [25,26]. Chen et al. [25] proposed a MRDH-CI method derived from secret sharing, in which multiple ciphered images are generated and assigned to multiple data hiders. Every data hider conceals the secret data into a designated area of the ciphered image through the substitution of the least significant bits. In [26], Hua et al. firstly proposed a cryptographic feedback secret sharing (CFSS) to generate multiple ciphered images. Then, a multi-MSB prediction method is employed to reserve embedding room for data hiding. MRDH-CI methods ensure the lossless restoration of the primitive image through gathering some marked ciphered images from intact data hides.

However, the MRDH-CI methods cannot efficiently deal with overexposed images since there are many pixels that are not directly encrypted into multiple ciphered pixels (called overflow pixels). They either fail to produce satisfactory results, or only work well for conventional images. To address this issue, multi-party reversible data hiding over ciphered overexposed images (MRDH-COI) is proposed, in which the overflow pixels are efficiently processed for data hiding. In the following, the contributions of this paper are encapsulated:

- By decomposing the pixel of the overexposed image into two parts, each of which is suitable for secret sharing, it is thus efficient to handle overflow pixels.
- An encryption algorithm combining group scrambling and modified secret sharing is given for the overexposed image, by which the differences of the groups of ciphered overexposed images are retained. It means that the ciphered overexposed images can be facilely used for data hiding.
- According to the given overexposed image encryption, the overflow pixels can be encrypted without labeling, so the executing time of the overexposed image encryption is reduced.

The subsequent sections of this paper are organized below. A brief analysis of related MRDH-CI works is given in Section 2. Section 3 provides the presented MRDH-COI approach. The experiment and its analysis are demonstrated in Section 4. At last, Section 5 draws our conclusion.

2. Related Works

The existing MRDH-CI methods are designed for conventional images. In [25], a MRDH-CI method is given, in which the traditional secret sharing strategy is employed to encrypt the primitive image into multiple ciphered images, and the ciphered images are distributed to multiple data hiders so that they can be used to embed data independently. In the case of a damaged hider, by collecting marked ciphered images from other undamaged hiders, the receiver is able to restore the primitive image in a lossless manner, which significantly improves the security of the image. In [26], a CFSS-based MRDH-CI method is proposed, in which the embedding room is reserved by the content owner. First, the median edge detector is used to predict the image to produce predictable pixels, and the MSB planes are determined based on the proportion of predictable pixels. Then, the predictor error is obtained by predicting the MSB planes, and the embedding room is reserved by encoding the predictor error with Huffman coding. This method provides a feasible and secure encryption protection mechanism for multiple data hiders.

The MRDH-CI methods proposed in [25,26] are constructed over finite field F_p by Shamir secret sharing [27], where p is a prime. For grayscale images with 8 bits, the pixels exceeding 250 are not suitable for secret sharing since p is set to 251. Therefore, the pixels exceeding 250 need to be processed so that they can be used for secret sharing. In [25], a location map (LM) is used to label the pixels exceeding 250 and concealed within the primitive image in a reversible manner. In [26], to label the pixels exceeding 250, the difference between $p - 1$ and the pixel value is encoded by introducing reference information. The reference information is embedded into the multiple MSBs of the pixel vacated via prediction error expansion.

However, the methods mentioned above have deficiencies in dealing with overexposed images. Four overexposed images selected from the standard dataset [28] are used for the demonstration as shown in Figure 1. We found that the percentages of pixels exceeding 250 are 51.7%, 61.7%, 59.8%, and 62.4%, respectively. In other words, there are plenty of overflow pixels that are not directly encrypted into multiple ciphered pixels. To deal with these overflow pixels, a lot of auxiliary information is required. For the method in [25], the primitive image may not have enough room to accommodate this auxiliary information. In [26], a lot of auxiliary information will increase the computational overhead since it is implemented by the reserving embedding room technique.

(**a**) (**b**) (**c**) (**d**)

Figure 1. Four overexposed images used for the experiment. (**a**) 1851.pgm, (**b**) 1933.pgm, (**c**) 2025.pgm, and (**d**) 7343.pgm.

Faced with these challenges, we propose a novel MRDH-CI method that focuses on solving the problem of overflow pixels in overexposed images. The challenge posed by overflow pixels is effectively overcome by splitting the pixel of the image into two parts. In addition, by combining group scrambling and modified secret sharing techniques, the grouping differences of overexposed images can be preserved during encryption,

thus making the ciphered overexposed images suitable for data hiding. The key to the proposed scheme is the successful fusion of overexposed image processing and encryption algorithms, which achieves the efficient encryption of overflow pixels without labeling. The proposed scheme simplifies the process of overexposed image encryption, thereby significantly reducing the time cost of processing overflow pixels.

3. Presented Approach

Figure 2 shows the given MRDH-COI approach, which involves encrypting the overexposed image, hiding data into ciphered overexposed images, and extracting hidden data and restoring overexposed images. For encrypting the overexposed image, the owner first scrambles an overexposed image to obtain a scrambled overexposed image. Then, the scrambled overexposed image is split into multiple ciphered overexposed images by secret sharing, and the ciphered overexposed images are sent to the associated data hiders, respectively. After that, every data hider conceals the secret data within a ciphered overexposed image, resulting in the creation of a marked ciphered overexposed image. In the step of extracting hidden data and restoring the overexposed image, once a sufficient number of marked ciphered overexposed images are obtained, the receiver first performs data extraction to obtain the embedded secret data. Subsequently, the receiver proceeds with the overexposed image restoration to obtain the primitive overexposed image.

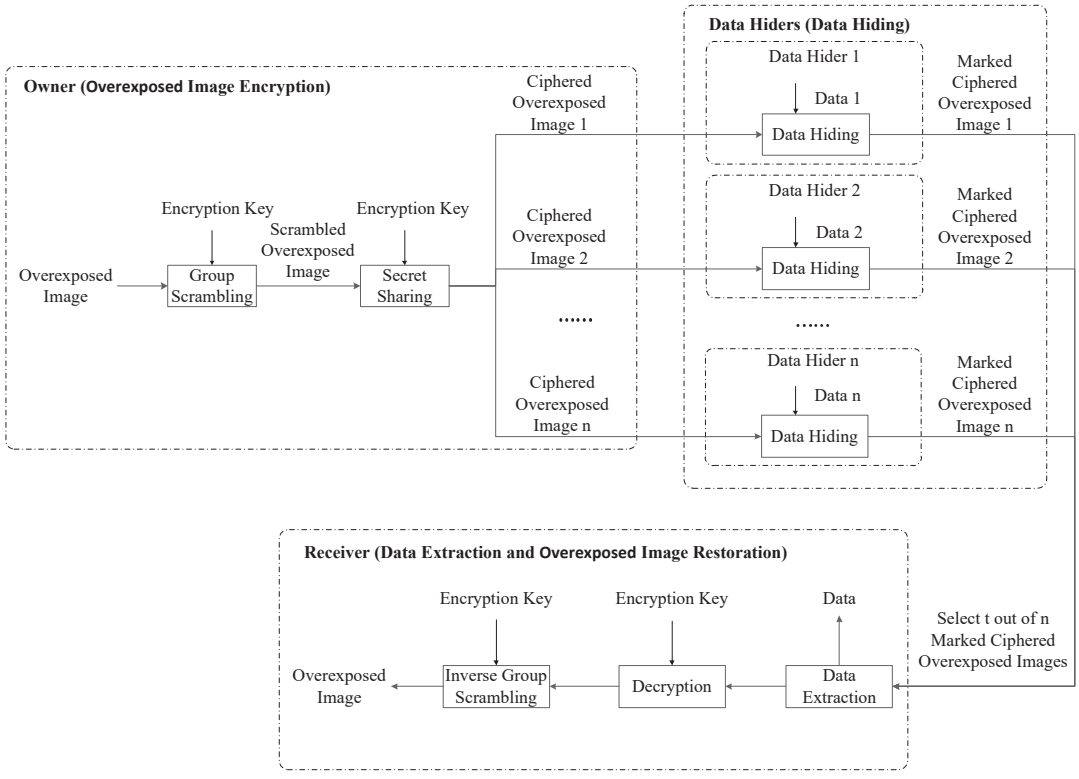


Figure 2. The framework of the presented approach, in which there are three parties actively involved.

3.1. Overexposed Image Encryption

Suppose the overexposed image O is an eight-bit grayscale image sized by $P \times Q$, and $O_{\alpha,\beta}$ is the pixel at location (α, β), where $O_{\alpha,\beta} \in [0, 255]$, $1 \leq \alpha \leq P$, $1 \leq \beta \leq Q$. For simplicity, P and Q are considered even. The overexposed image encryption is comprised of two parts: group scrambling and secret sharing. First, the owner divides the overexposed image into groups with size 1×2. It is easily obtained that there are $PQ/2$ groups. All groups of the overexposed image are scrambled by using an encryption key, then a scrambled overexposed image O' is obtained. It is worth mentioning that by using a pseudorandom number generator with a seed, the encryption key can be created. The key generation

process is shown in Figure 3. First of all, the content owner chooses a seed at random and assigns the seed to the state. Secondly, with the state, an encryption key is generated by adopting a 256-bit secure hash algorithm (SHA-256). At the same time, the state is modified to the sum of the encryption key and step m, where m is an integer. Finally, the second step is repeated to generate multiple encryption keys as needed. In general, the encryption key is shared over a covert channel or a public channel with the assistance of a key exchange protocol, such as the Diffie–Hellman key negotiation algorithm.

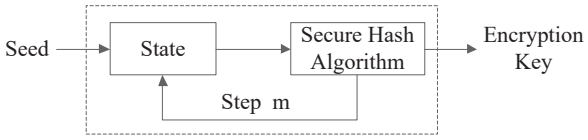


Figure 3. Sketch of the key generation process, where the security level of the secure hash algorithm is 256 bits.

After group scrambling, secret sharing is performed on the scrambled overexposed image O'. Herein, a modified secret sharing method inherited from Shamir secret sharing is used to convert the scrambled overexposed image into multiple ciphered overexposed images. For grayscale images with eight bits, pixels exceeding 250 do not lend themselves to secret sharing. To address this issue, we decompose the pixels of the overexposed image and make a simple modification to (t, n) threshold secret sharing [27], where $2 \leq t \leq n$. Let a group of the scrambled overexposed image be $(O'_{i,j}, O'_{i,j+1})$, where $1 \leq i \leq P$, $j = 1, 3, \ldots, Q-1$. To obtain multiple ciphered overexposed images, a identifier is chosen for each ciphered overexposed image, denoted as x_r, $1 \leq r \leq n$, each of which is distinct from one another. Usually, x_r is determined by the encryption key. When $O'_{i,j} \in [d, 255]$ or $O'_{i,j+1} \in [d, 255]$, a polynomial $f(x)$ with a degree of $t-1$ is formulated via

$$f(x) = \left(a_0 + dx + \sum_{q=2}^{t-1} a_q x^q \right) \bmod p, \tag{1}$$

where $a_0 = O'_{i,j} - d$ or $a_0 = O'_{i,j+1} - d$, $d \geq 5$ is a integer, and $a_2, \ldots, a_{t-1} \in F_p$ are chosen at random. For each specified x_r, a corresponding $f(x_r)$ can be calculated by Equation (1), and $f(x_r)$ is set as the pixel of the ciphered overexposed image. When $O'_{i,j} \in [0, d-1]$ or $O'_{i,j+1} \in [0, d-1]$, another polynomial $f(x)$ with a degree of $t-1$ is formulated via

$$f(x) = \left(a_0 + 0x + \sum_{q=2}^{t-1} a_q x^q \right) \bmod p, \tag{2}$$

where $a_0 = O'_{i,j}$ or $a_0 = O'_{i,j+1}$. Similarly, for each specified x_r, another corresponding $f(x_r)$ can be calculated by Equation (2), and the corresponding $f(x_r)$ is set as the pixel of the ciphered overexposed image. In this way, n ciphered overexposed images are created, and the created ciphered overexposed images are shared among n data hiders for data hiding. Algorithm 1 shows the procedure of the overexposed image encryption.

Next, we will demonstrate that the obtained ciphered overexposed images are suitable for data hiding. An example is given in Figure 4, where $(3, 3)$ modified secret sharing is adopted. Assume that a group of the scrambled overexposed image is $(252, 254)$. With the encryption key, three identifiers 2, 3, and 5 are generated. According to Equation (1), two 2-degree polynomials with $d = 5$ and random integer $a_2 = 10$ are constructed by

$$f(x) = 247 + 5x + 10x^2 \bmod 251, \tag{3}$$

and

$$f(x) = 249 + 5x + 10x^2 \bmod 251, \tag{4}$$

respectively. Three identifiers 2, 3, and 5 are substituted into Equations (3) and (4), and three corresponding groups of ciphered overexposed images $(46, 48)$, $(101, 103)$, and $(20, 22)$, are generated, respectively. As can be seen, the differences of the generated three groups are retained, which means that the ciphered overexposed images can be used for data hiding.

Algorithm 1 Overexposed image encryption.

Input: Overexposed Image O, Encryption Key.
Output: Ciphered Overexposed Image CO^r.

```
Divide O into groups with size 1×2;
Obtain scrambled overexposed image O′ by scrambling the groups;
for r ← 1 : n do
    Generate x_r with the encryption key;
    for i ← 1 : P do
        for j ← 1 : Q do
            if O′_{i,j} ≥ d && O′_{i,j} ≤ 255 then
                a_0 ← O′_{i,j} − d;
                Construct polynomial f(x) by Equation (1) with a_0, d and random integer a_q;
            else
                a_0 ← O′_{i,j};
                Construct polynomial f(x) by Equation (2) with a_0 and random integer a_q;
            end if
            CO^r_{i,j} ← Calculate f(x) with x_r;
        end for
    end for
end for
```

For the group belonging to $[0, d-1]$, the difference retaining verification is similar to the group belonging to $[d, 255]$, and thus no description is provided here.

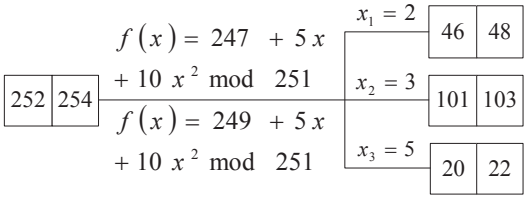


Figure 4. An example of the difference retaining in a group.

3.2. Data Hiding

To manage the obtained ciphered overexposed images, data hiders conceal secret data into them to create marked ciphered overexposed images without accessing the primitive overexposed image. Based on the analysis in Section 3.1, the differences of the groups of ciphered overexposed images are retained, so data hiding can be achieved using the classic difference expansion algorithm, such as the method in [7]. Let the ciphered overexposed image be CO^r,$1 \leq r \leq n$. The group of the ciphered overexposed image is represented by $(CO^r_{i,j}, CO^r_{i,j+1})$, $1 \leq i \leq P$, $j = 1, 3, \ldots, Q-1$. Specifically, comprehensive description of the data hiding algorithm is provided below.

1. Compute the integer mean and difference of the groups of ciphered overexposed images. For each $(CO^r_{i,j}, CO^r_{i,j+1})$, the integer mean and difference are computed by $l = \left\lfloor \frac{CO^r_{i,j} + CO^r_{i,j+1}}{2} \right\rfloor$ and $h = CO^r_{i,j} - CO^r_{i,j+1}$, respectively, where $\lfloor \cdot \rfloor$ is a floor function.
2. Determine the expandable and non-expandable groups. A group is expandable if its integer mean l and difference h satisfy $\left| 2 \times \lfloor \frac{h}{2} \rfloor + b \right| \leq \min(2(255 - l), 2l + 1)$, where

b is the bit of the secret data; otherwise, it is considered non-expandable. In addition, an LM is created to label expandable and non-expandable groups, and their associated bits are set to 0 and 1, respectively.

3. Embed secret data into the expandable groups. First, for each expandable group, a new difference is generated with the difference h by

$$h' = 2 \times h + b, \tag{5}$$

where $b \in \{0, 1\}$ is the bit of the secret data that is concealed within the ciphered overexposed images. Then, with the new difference h', the pixels of the marked ciphered overexposed image are computed by

$$\begin{cases} COD^r_{i,j} = l + \lfloor \frac{h'+1}{2} \rfloor, \\ COD^r_{i,j+1} = l - \lfloor \frac{h'}{2} \rfloor, \end{cases} \tag{6}$$

where $COD^r_{i,j}$ (resp. $COD^r_{i,j+1}$) is the pixel of the rth marked ciphered overexposed image at location (i, j) (resp. $(i, j+1)$).

By these means, the ciphered overexposed images can accommodate the embedding of secret data, and the marked ciphered overexposed images are generated, denoted as COD^r, $1 \leq r \leq n$. The data hiding procedure is given in Algorithm 2.

Algorithm 2 Data hiding.

Input: Ciphered Overexposed Image CO^r, Embedded Bit b.
Output: Marked Ciphered Overexposed Image COD^r.

for $i \leftarrow 1 : P$ **do**
  for $j \leftarrow 1 : 2 : Q$ **do**
    $h \leftarrow CO^r_{i,j} - CO^r_{i,j+1}$;
    $l \leftarrow \lfloor (CO^r_{i,j} + CO^r_{i,j+1})/2 \rfloor$;
    if $|2 \times \lfloor h/2 \rfloor + b| \leq \min(2 \times (255 - l), 2 \times l + 1)$ **then**
      $h' \leftarrow 2 \times h + b$;
      $COD^r_{i,j} \leftarrow l + \lfloor \frac{h'+1}{2} \rfloor$;
      $COD^r_{i,j+1} \leftarrow l - \lfloor \frac{h'}{2} \rfloor$;
    end if
  end for
end for

It is noted that the LM has high data redundancy since most groups are expandable. We use run-length coding to compress LM to generate compressed LM. Meanwhile, the compressed LM is required for data retrieval and overexposed image reconstruction, and hence it is embedded into the pixels of the initial few rows of the ciphered overexposed images. The initial few rows of pixels from the ciphered overexposed images will be concealed within the remaining pixels along with the secret data.

3.3. Data Extraction and Overexposed Image Restoration

When having any t marked ciphered overexposed images, the concealed data and the primitive overexposed image can be obtained by the receiver. Assume any t marked ciphered overexposed images are received, represented by $COD^{r_1}, COD^{r_2}, \ldots, COD^{r_t}$, where $\{r_1, r_2, \ldots, r_t\} \subseteq \{1, 2, \ldots, n\}$.

3.3.1. Data Extraction

Extracting concealed secret data is performed on the expandable groups of marked ciphered overexposed images. Denote $(COD^{r_s}_1, COD^{r_s}_2)$ as an expandable group, where $s = 1, 2, \ldots, t$. The data extraction is described in detail as follows:

1. Fix the expandable groups. Extract the compressed LM from the pixels of the initial few rows of marked ciphered overexposed images, and decode it to obtain the associated LM. By the decoded LM, the expandable groups are fixed, that is, if the bit of LM is 0, the corresponding group is expandable.
2. Compute the integer mean and difference of the expandable groups. For an expandable group $(COD_1^{r_s}, COD_2^{r_s})$, the integer mean l' and difference h' are computed by $l' = \left\lfloor \frac{COD_1^{r_s} + COD_2^{r_2}}{2} \right\rfloor$ and $h' = COD_1^{r_s} - COD_2^{r_s}$, respectively.
3. Retrieve the secret data by utilizing the difference h'. The embedded secret data are obtained by

$$b = h' \bmod 2, \tag{7}$$

where b is the bit of the secret data.

Algorithm 3 shows the data extraction procedure.

Algorithm 3 Data extraction.

Input: Any t Marked Ciphered Overexposed Image COD^{r_s}.
Output: Embedded bit b.

```
for i ← 1 : P do
    for j ← 1 : 2 : Q do
        if (COD^{r_s}_{i,j}, COD^{r_s}_{i,j+1}) is an expandable group then
            l' ← ⌊(COD^{r_s}_{i,j} + COD^{r_s}_{i,j+1})/2⌋;
            h' ← COD^{r_s}_{i,j} − COD^{r_s}_{i,j+1};
            b ← h' mod 2;
        end if
    end for
end for
```

3.3.2. Overexposed Image Restoration

For overexposed image restoration, we first transform the marked ciphered overexposed images to ciphered overexposed images and then restore the primitive overexposed image from the ciphered overexposed images. In fact, only expandable groups of the ciphered overexposed images are modified for data hiding. Therefore, the non-expandable groups of the marked ciphered overexposed images can be recognized as those of the ciphered overexposed images. On the other hand, with the integer mean l' and the difference h', the expandable groups of the ciphered overexposed images can be obtained by

$$\begin{cases} CO_1^{r_s} = l' + \lfloor \frac{h+1}{2} \rfloor, \\ CO_2^{r_s} = l' - \lfloor \frac{h}{2} \rfloor, \end{cases} \tag{8}$$

where $(CO_1^{r_s}, CO_2^{r_s})$, $1 \le s \le t$ is an expandable group of the ciphered image CO^{r_s} and $h = \lfloor \frac{h'}{2} \rfloor$.

After the t ciphered overexposed images are generated, the primitive overexposed image is recovered. According to the encryption key, the identified x_{r_s},$1 \le s \le t$, can be obtained. Subsequently, the polynomial $f(x)$ with a degree of $t-1$ is reconstructed via the Lagrange polynomial interpolation technique as represented by

$$f(x) = \sum_{1 \le v \le t} \left(CO_{\alpha,\beta}^{r_v} \prod_{1 \le u \le t, u \ne v} \frac{x - x_{r_u}}{x_{r_v} - x_{r_u}} \right) \bmod p, \tag{9}$$

where $CO_{\alpha,\beta}^{r_v}$ is the pixel of the ciphered overexposed image CO^{r_v} at location (α, β), $1 \le \alpha \le P$, $1 \le \beta \le Q$. Obviously, the parameter d is equivalent to the coefficient associated with a term in the function $f(x)$. If $d \ne 0$, the pixel of the scrambled over-

exposed image is calculated by $O'_{\alpha,\beta} = a_0 + d$ or $O'_{\alpha,\beta+1} = a_0 + d$; otherwise, we have $O'_{\alpha,\beta} = a_0$ or $O'_{\alpha,\beta+1} = a_0$, where a_0 is the constant term of $f(x)$. Finally, the primitive overexposed image is obtained from the scrambled overexposed image according to the inverse operation of the group scrambling with the encryption key. The procedure of overexposed image restoration is provided in Algorithm 4.

Algorithm 4 Overexposed image restoration.

Input: Any t Marked Ciphered Overexposed Image COD^{r_s}, Encryption Key.
Output: Primitive Overexposed Image O.

for $i \leftarrow 1 : P$ **do**
  for $j \leftarrow 1 : 2 : Q$ **do**
    if $(COD^{r_s}_{i,j}, COD^{r_s}_{i,j+1})$ is an expandable group **then**
      $l' \leftarrow \lfloor (COD^{r_s}_{i,j} + COD^{r_s}_{i,j+1})/2 \rfloor$;
      $h' \leftarrow COD^{r_s}_{i,j} - COD^{r_s}_{i,j+1}$;
      $h \leftarrow \lfloor h'/2 \rfloor$;
      $CO^{r_s}_{i,j} \leftarrow l' + \lfloor \frac{h+1}{2} \rfloor$;
      $CO^{r_s}_{i,j+1} \leftarrow l' - \lfloor \frac{h}{2} \rfloor$;
    end if
  end for
end for
Generate x_{r_s} with the encryption key;
for $i \leftarrow 1 : P$ **do**
  for $j \leftarrow 1 : Q$ **do**
    Reconstruct $f(x)$ by Equation (9) with $CO^{r_s}_{i,j}$ and x_{r_s};
    $a_0 \leftarrow$ Constant term of $f(x)$;
    $d \leftarrow$ One term of $f(x)$;
    if $d \neq 0$ **then**
      $O'_{i,j} \leftarrow a_0 + d$;
    else
      $O'_{i,j} \leftarrow a_0$;
    end if
  end for
end for
Obtain O by the inverse operation of the group scrambling with O';

4. Experimental Results and Analysis

Herein, the presented method will be evaluated by some experiments, such as effectiveness, security, efficiency, and embedding capacity. Four overexposed images from the common dataset [28] are employed for our experiments, including "1851.pgm", "1933.pgm", "2025.pgm", and "7343.pgm". All the overexposed images possess a 512×512 resolution and are in grayscale as shown in Figure 1.

4.1. Experimental Setup

During the experiment, a computer running 64-bit Windows 10 Professional is chosen as the hardware environment. This computer is configured with an Intel Core i5-11600 processor at 2.8 GHz and an 8 GB of Kingston DDR4 memory. The strong hardware platform provides reliable performance and computational resources to ensure efficient execution of our experiments. Matlab 2012a serves as the experimental software tool, which provides comprehensive support for the research conducted in this paper. This software is a professional tool widely used in the fields of scientific computing and engineering, which contributes to the thorough execution of the study. In the numerical processing procedure, all programs are developed by Matlab language and carried out in Matlab

2012a. The built-in plotting tools of Matlab software are utilized to ensure user-friendly visualization of the experimental results.

4.2. Effectiveness of the Presented Scheme

In this experiment, we conduct tests on an overexposed image to demonstrate the effectiveness of the presented scheme. The result of the experiment is given in Figure 5, where (3,4) modified secret sharing is used. According to the proposed overexposed image encryption algorithm, the overexposed image is encrypted into four different ciphered overexposed images as shown in Figure 5b–e. For each ciphered overexposed image, the secret data are concealed into it, and a marked ciphered overexposed image is generated. The generated marked ciphered overexposed images are shown in Figure 5f–i. With any three marked ciphered overexposed images, the concealed secret data are retrieved, and the primitive overexposed image is restored. The restored overexposed images are shown in Figure 5j–m, each of which is identical to the primitive overexposed image. Therefore, the presented approach can efficiently deal with the overexposed image. In [25,26], the overflow pixels are processed by using a labeling technique. In fact, labeling the overflow pixels generates a lot of auxiliary information, especially for overexposed images. As a result, the primitive overexposed image does not have enough room to accommodate it.

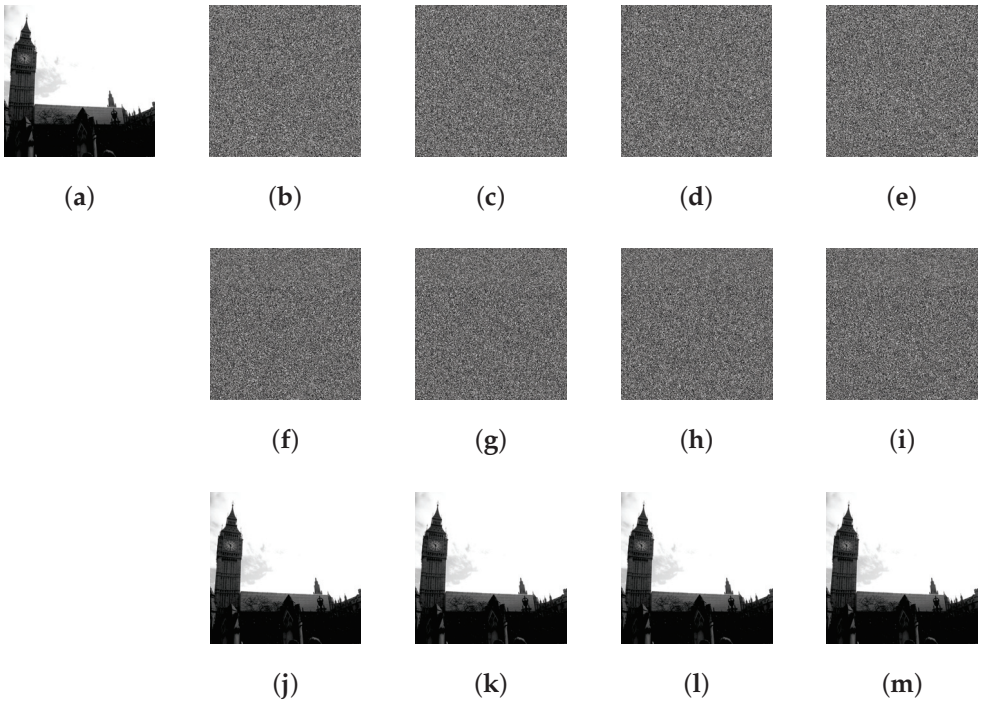


Figure 5. Validate the proposed approach employing (3,4) modified secret sharing. (**a**) The primitive overexposed image, (**b**) the first ciphered overexposed image, (**c**) the second ciphered overexposed image, (**d**) the third ciphered overexposed image, (**e**) the fourth ciphered overexposed image, (**f**) the first marked ciphered overexposed image, (**g**) the second marked ciphered overexposed image, (**h**) the third marked ciphered overexposed image, (**i**) the fourth marked ciphered overexposed image, (**j**) the reestablished overexposed image with (**f**–**h**), (**k**) the reestablished overexposed image with (**f**,**g**,**i**), (**l**) the reestablished overexposed image with (**f**,**h**,**i**), and (**m**) the reestablished overexposed image with (**g**–**i**).

4.3. Security Analysis

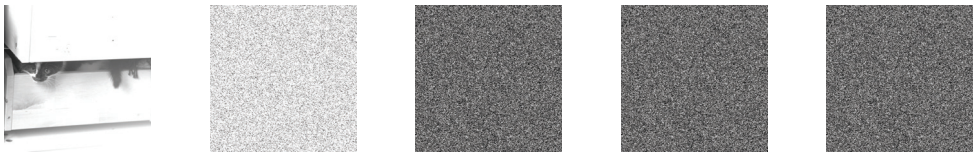
The security of the presented approach is evaluated in the following aspects.

4.3.1. Key Strength

According to the key generation process shown in Figure 3, a 256-bit secure hash algorithm (SHA-256) is adopted to generate the encryption key. The security of the encryption key is based on the SHA-256. Thus, the length of the encryption key is 256 bits, and the space of the encryption key is 2^{256}. In order to recover a certain pixel, the adversary needs to obtain parameter x_{r_s}, $1 \leq s \leq t$, $2 \leq t \leq n$. This means that the adversary can break the certain pixel by exploiting A^t_{p-1} attacks, where $p = 251$ for 8-bit overexposed image. On the other hand, the encryption key can be used to encrypt 32 pixels at a time since it is of a 256-bit length. For 32 pixels, the adversary needs $(A^t_{p-1})^{32}$ attacks. Therefore, the strength of the encryption key is $32log_2^{A^t_{p-1}}$.

4.3.2. Imperceptibility

The analysis of the imperceptibility security is illustrated in Figure 6. Figure 6a shows the primitive overexposed image, while Figure 6b presents the image after group scrambling. From Figure 6b, it can be seen that after the first stage of encryption, the image takes on a noise-like appearance and does not show any information about the primitive image. To further improve security, we perform a second round of encryption via modified secret sharing. The scrambled image is segmented into multiple ciphered images (only the effect of one ciphered image is shown here) as shown in Figure 6c. Upon closer examination of these images, we observe that after the second round of encryption, the images exhibit more randomized noise characteristics. More importantly, a sufficient number of ciphered images must be collected if the primitive overexposed image is to be successfully recovered. Figure 6d,e show the marked ciphered images at different embedding rates, respectively. From these two images, we do not obtain any information about the primitive overexposed image and the embedded data, which means that different embedding capacities do not affect the imperceptibility security. Therefore, the presented approach effectively obfuscates the visual characteristics of the image and provides reliable imperceptibility security for the primitive overexposed image.



(**a**) (**b**) (**c**) (**d**) (**e**)

Figure 6. Imperceptibility of the presented approach. (**a**) Primitive overexposed image, (**b**) the scrambled overexposed image, (**c**) the ciphered overexposed image, (**d**) the marked ciphered overexposed image with an embedding rate of 0.2 bpp, and (**e**) the marked ciphered overexposed image with an embedding rate of 0.4 bpp.

4.3.3. Statistical Security

In the proposed scheme of this paper, we adopt a modified secret sharing strategy to share the scrambled overexposed image. This scheme fully utilizes the characteristics of the secret sharing, providing excellent fault tolerance for the overexposed image. To assess the statistical properties of image encryption and data hiding, we plot the histograms of the ciphered and marked ciphered images of two overexposed images, respectively, as shown in Figure 7. The histogram distributions of these two images are shown in Figure 7a,h. Observing the histogram reveals that the pixel values of the primitive image are unevenly distributed. Figure 7b–d,i–k show the histogram distribution of the ciphered image through secret sharing. As can be seen from these histogram distributions, the pixel values of the ciphered images are evenly distributed, thereby effectively improving the statistical security. In addition, Figure 7e–g,l–n show the histogram distributions of the marked ciphered

images with embedding rates of 0.1 bpp, 0.2 bpp, and 0.3 bpp, respectively. Observing these six histograms, it is evident that despite different embedding rates, the pixel distribution of the marked ciphered images remains uniform. This indicates that the proposed scheme is effective in protecting image content, whether for different embedding rates of the same image or similar embedding rates for different images.

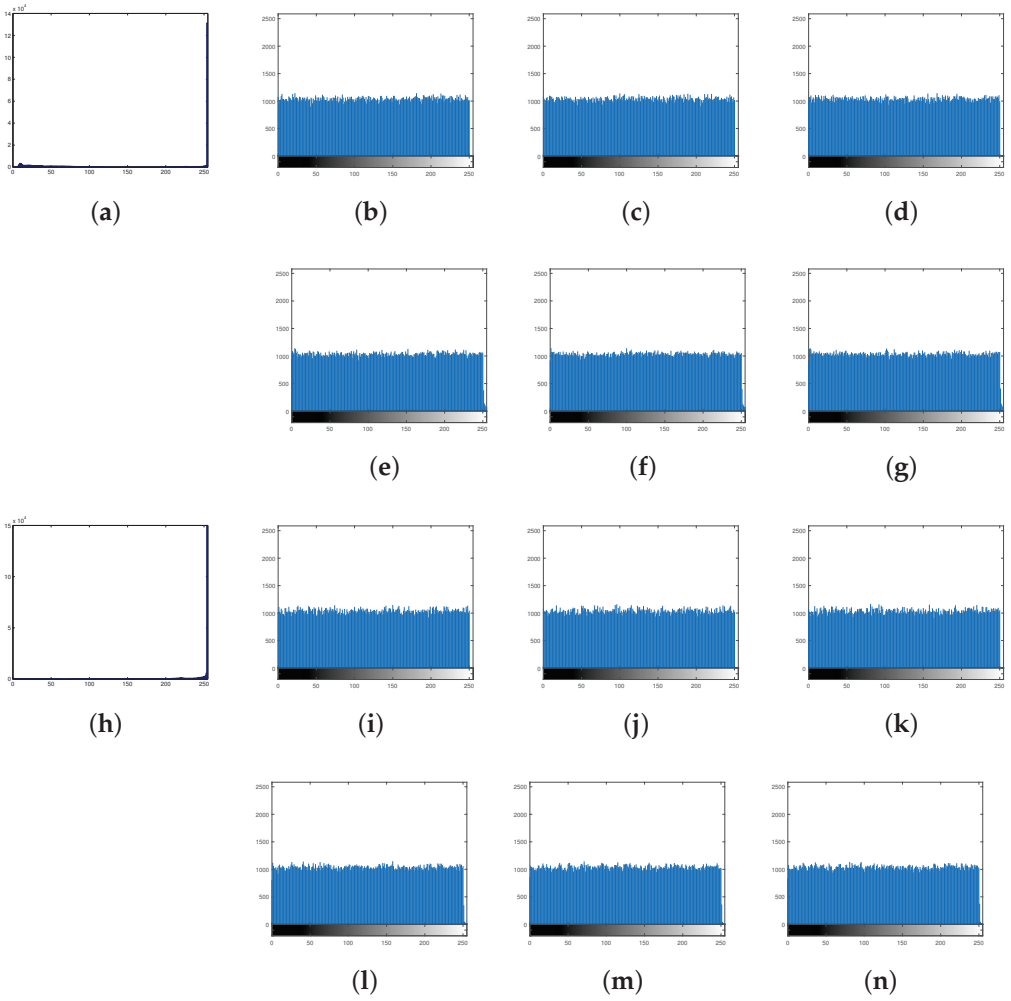


Figure 7. Histograms of the overexposed images for statistical security analysis. (**a**) Primitive overexposed image "1851.pgm", (**b**–**d**) the three ciphered overexposed images of (**a**), (**e**–**g**) the three marked ciphered overexposed images with respect to (**b**–**d**), (**h**) primitive overexposed image "7343.pgm", (**i**–**k**) the three ciphered overexposed images of (**h**), and (**l**–**m**) the three marked ciphered overexposed images with respect to (**i**–**k**).

4.3.4. Occlusion Attack

In the presented scheme, by sharing the primitive overexposed image into n ciphered overexposed images with a modified secret sharing method, we enhance the resistance of the image to occlusion attack. An adversary may attempt to hinder the understanding of the image by obscuring or disrupting it. However, due to the adoption of secret sharing, the information of the primitive overexposed image is distributed among multiple marked ciphered overexposed images. Thus, even if a portion of the marked ciphered overexposed images are subjected to an occlusion attack, it is still possible to recover the primitive overexposed image by collecting a sufficient number of marked ciphered overexposed images. This fault tolerance enables the effective restoration of the primitive overexposed image, thereby enhancing the robustness of the proposed scheme against occlusion attacks.

4.4. Discussion on Parameter d

Parameter d is one of the important factors for effectively handling overflow pixels in the presented method. The trend of embedding capacity with the variation of parameter d is illustrated in Figure 8. It can be observed from waveform diagrams that the variation of parameter d has a very small effect on the embedding capacity. It is interesting to note that the optimal choice of parameter d may vary due to the introduction of random numbers in the polynomials. This implies that the embedding capacity exhibits randomness and uncertainty for different values of parameter d. This randomness plays a positive role in improving the security and attack resistance of the scheme. Specifically, in these four images, the fluctuation range of the embedding capacity remains between 0.0018 and 0.0039 as d changes. This variation may be related to the smoothness or randomness of the image. For the image "7343.pgm", the embedding capacity fluctuates the least, showing a relatively stable behavior. Also, the experimental results show that the embedding capacity of "7343.pgm" is higher compared to the other three images. Overall, the effect of the parameter d on the embedding capacity is very small, while the introduced randomness effectively improves the security of the proposed scheme.

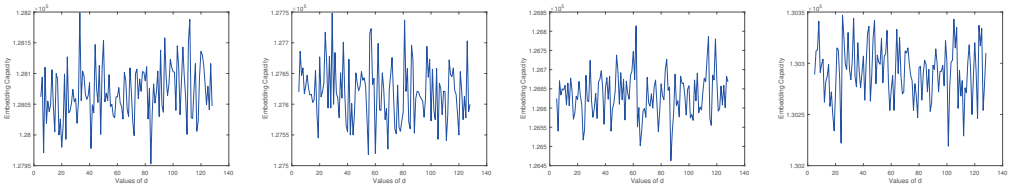


Figure 8. Effect of the parameter d changes on embedding capacity for different overexposed images. (**a**) 1851.pgm, (**b**) 1933.pgm, (**c**) 2025.pgm, and (**d**) 7343.pgm.

4.5. Efficiency Comparison

Efficiency is a key metric of the MRDH-CI method. In this context, we consider the executing time to demonstrate efficiency. Table 1 shows the comparison of the average executing time of the overexposed image encryption phase between the proposed approach and the approach in [26] under various encryption strategies for different overexposed images. For the image encryption of the presented approach, the overexposed image is directly ciphered by the group scrambling and modified secret sharing. Instead, for the image encryption of the method in [26], reserving the embedding room is performed first, and then the image is encrypted by secret sharing. It is known that the presented approach gains a faster processing speed in the image encryption phase by increasing parameters t and n. It means that reserving embedding room increases the extra computational overhead.

Table 1. Comparison of executing time (in second) between the presented approach and the approach in [26] under different encryption strategies for overexposed images.

Schemes	**Overexposed Images**	**Encryption Strategies**				
		(3,4)	**(3,5)**	**(4,5)**	**(3,6)**	**(4,6)**
Hua et al. [26]	1851.pgm	13.53	15.28	11.82	17.06	12.39
	1933.pgm	12.61	14.23	10.38	16.01	12.16
	2025.pgm	15.09	16.07	12.17	18.58	14.05
	7343.pgm	9.95	11.48	8.75	12.77	9.58
Proposed	1851.pgm	6.28	7.76	7.65	8.89	9.24
	1933.pgm	7.15	8.21	7.70	9.17	9.89
	2025.pgm	6.33	9.31	8.07	9.22	9.83
	7343.pgm	6.51	7.47	8.05	9.44	9.53

4.6. Embedding Capacity Analysis

The embedding rate is used to assess embedding performance. The embedding rate (bpp) is the proportion of the total embedded bits to the total pixels of each ciphered overexposed image. For the method in [25], the maximum embedding rate is approximately 3.5 bpp since 7 bits can be embedded into one pixel when sharing two ciphered images. At the same time, the embedding rate does not vary with the distribution of images. However, the embedding rate decreases rapidly as more ciphered images are shared. For the method in [26], it is also possible to achieve a high embedding rate since the hiding room is created from the primitive image during the image encryption stage. But vacating the embedding room from the primitive image requires more computational overhead. As discussed in Section 4.5, the method in [26] has a longer executing time in the image encryption phase. In the proposed scheme, a bit is embedded into a group, and a embedding rate with 0.5 bpp is obtained. Obviously, the presented approach cannot produce a satisfactory embedding capacity due to conservative hiding. To improve the embedding capacity, multi-level hiding is introduced, that is, hiding the data into an expandable group multiple times until the expandable group becomes a non-expandable group.

5. Conclusions

Recently, there has been growing interest in the MRDH-CI approach. This approach ensures the ability to restore the primitive image, even when some data hiders are damaged. However, the MRDH-CI cannot efficiently deal with overexposed images since there are plenty of overflow pixels that are not directly encrypted into multiple ciphered pixels. This paper proposes a MRDH-COI, in which the overflow pixels can be efficiently processed. First of all, the pixels of the group of the overexposed image are decomposed into two parts so that they can be encrypted by secret sharing. In order to encrypt the overexposed image into multiple ciphered overexposed images, the overexposed image is first scrambled by group scrambling and then converted by modified secret sharing. The obtained ciphered overexposed images have the ability to facilitate data hiding since their differences are retained. With the symmetry of the difference retaining, the difference expansion algorithm can be exploited to conceal the secret data within ciphered overexposed images to generate marked ciphered overexposed images. When any sufficient marked ciphered overexposed images are authorized, the primitive overexposed image and the concealed data are retrieved symmetrically. Finally, an experiment and its analysis are provided to illustrate the effectiveness and efficiency of the presented approach.

Author Contributions: Conceptualization, B.C. and J.C.; methodology, B.C. and R.Y.; software, R.Y., W.F. and X.Z.; validation, B.C. and J.C.; formal analysis, R.Y. and J.C.; investigation, W.F. and X.Z.; writing—original draft preparation, B.C. and R.Y.; writing—review and editing, B.C. and R.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 62102101) and the Doctoral Scientific Research Foundation of Guangdong Polytechnic Normal University (No. 2021SDKYA101).

Data Availability Statement: Data will be made available on request.

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

References

1. Mathivanan, P.; Balaji Ganesh, A. QR code based color image stego-crypto technique using dynamic bit replacement and logistic map. *Optik* **2021**, *225*, 165838.
2. Mathivanan, P.; Balaji Ganesh, A. ECG steganography using Base64 encoding and pixel swapping technique. *Multimed. Tools Appl.* **2023**, *82*, 14945–14962. [CrossRef]
3. Yang, Y.; Xiao, X.; Cai, X.; Zhang, W. A Secure and Privacy-Preserving Technique Based on Contrast-Enhancement Reversible Data Hiding and Plaintext Encryption for Medical Images. *IEEE Signal Process. Lett.* **2020**, *27*, 256–260. [CrossRef]
4. Peng, Y.; Zhao, Y.; Mao, N.; Wang, J.; Fang, Y.; Zhang, T.; Shi, F. Color image reversible data hiding based on multi-channel synchronized histogram. *J. King Saud Univ. Comput. Inf. Sci.* **2023**, *35*, 101804. [CrossRef]

5. Weng, S.; Zhou, Y.; Zhang, T.; Xiao, M.; Zhao, Y. Reversible data hiding for JPEG images with adaptive multiple two-dimensional histogram and mapping generation. *IEEE Trans. Multimed.* **2023**, *25*, 8738–8752. [CrossRef]
6. Mandal, P.C.; Mukherjee, I.; Chatterji, B.N. High capacity reversible and secured data hiding in images using interpolation and difference expansion technique. *Multimed. Tools Appl.* **2021**, *80*, 3623–3644. [CrossRef]
7. Tian, J. Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [CrossRef]
8. Sahu, A.K.; Swain, G. High fidelity based reversible data hiding using modified LSB matching and pixel difference. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 1395–1409. [CrossRef]
9. Qi, W.; Li, X.; Zhang, T.; Guo, Z. Optimal Reversible Data Hiding Scheme Based on Multiple Histograms Modification. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 2300–2312. [CrossRef]
10. Weng, S.; Tan, W.; Ou, B.; Pan, J.S. Reversible data hiding method for multi-histogram point selection based on improved crisscross optimization algorithm. *Inf. Sci.* **2021**, *549*, 13–33. [CrossRef]
11. Pan, Z.; Gao, X.; Gao, E.; Fan, G. Adaptive Complexity for Pixel-Value-Ordering Based Reversible Data Hiding. *IEEE Signal Process. Lett.* **2020**, *27*, 915–919. [CrossRef]
12. Zhang, T.; Li, X.; Qi, W.; Guo, Z. Location-Based PVO and Adaptive Pairwise Modification for Efficient Reversible Data Hiding. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 2306–2319. [CrossRef]
13. He, W.; Cai, Z. Reversible Data Hiding Based on Dual Pairwise Prediction-Error Expansion. *IEEE Trans. Image Process.* **2021**, *30*, 5045–5055. [CrossRef]
14. Yu, C.; Zhang, X.; Wang, D.; Tang, Z. Reversible data hiding with pairwise PEE and 2D-PEH decomposition. *Signal Process.* **2022**, *196*, 108527. [CrossRef]
15. Kouhi, A.; Sedaaghi, M.H. Reversible data hiding based on high fidelity prediction scheme for reducing the number of invalid modifications. *Inf. Sci.* **2022**, *589*, 46–61. [CrossRef]
16. Dragoi, I.C.; Coltuc, D. On the Security of Reversible Data Hiding in Encrypted Images by MSB Prediction. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 187–189. [CrossRef]
17. Wang, Y.; He, W. High Capacity Reversible Data Hiding in Encrypted Image Based on Adaptive MSB Prediction. *IEEE Trans. Multimed.* **2022**, *24*, 1288–1298. [CrossRef]
18. Gao, K.; Horng, J.H.; Chang, C.C. High-capacity reversible data hiding in encrypted images based on adaptive block encoding. *J. Vis. Commun. Image Represent.* **2022**, *84*, 103481. [CrossRef]
19. Sui, L.; Li, H.; Liu, J.; Xiao, Z.; Tian, A. Reversible Data Hiding in Encrypted Images Based on Hybrid Prediction and Huffman Coding. *Symmetry* **2023**, *15*, 1222. [CrossRef]
20. Qian, Z.; Xu, H.; Luo, X.; Zhang, X. New Framework of Reversible Data Hiding in Encrypted JPEG Bitstreams. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 351–362. [CrossRef]
21. Ge, B.; Ge, G.; Xia, C.; Duan, X. High-Capacity Reversible Data Hiding in Encrypted Images Based on 2D-HS Chaotic System and Full Bit-Plane Searching. *Symmetry* **2023**, *15*, 1423. [CrossRef]
22. Wu, H.T.; Cheung, Y.M.; Yang, Z.; Tang, S. A high-capacity reversible data hiding method for homomorphic encrypted images. *J. Vis. Commun. Image Represent.* **2019**, *62*, 87–96. [CrossRef]
23. Zheng, S.; Wang, Y.; Hu, D. Lossless Data Hiding Based on Homomorphic Cryptosystem. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 692–705. [CrossRef]
24. Ke, Y.; Zhang, M.Q.; Liu, J.; Su, T.T.; Yang, X.Y. Fully Homomorphic Encryption Encapsulated Difference Expansion for Reversible Data Hiding in Encrypted Domain. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 2353–2365. [CrossRef]
25. Chen, B.; Lu, W.; Huang, J.; Weng, J.; Zhou, Y. Secret Sharing Based Reversible Data Hiding in Encrypted Images With Multiple Data-Hiders. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 978–991. [CrossRef]
26. Hua, Z.; Wang, Y.; Yi, S.; Zhou, Y.; Jia, X. Reversible Data Hiding in Encrypted Images Using Cipher-Feedback Secret Sharing. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 4968–4982. [CrossRef]
27. Shamir, A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
28. The Bossbase 1.01 Image Database, 2023. Available online: http://dde.binghamton.edu/download/ (accessed on 20 November 2023).

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Symmetry Editorial Office
E-mail: symmetry@mdpi.com
www.mdpi.com/journal/symmetry

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-8490-2