applied sciences

MDPI

*Article*

# Reversible Adversarial Examples with Minimalist Evolution for Recognition Control in Computer Vision

**Shilong Yang [1], Lu Leng [1,*], Ching-Chun Chang [2] and Chin-Chen Chang [3]**

[1] Jiangxi Provincial Key Laboratory of Image Processing and Pattern Recognition, Nanchang Hangkong University, Nanchang 330063, China
[2] Information and Communication Security Research Center, Feng Chia University, Taichung 407102, Taiwan
[3] Information Engineering and Computer Science, Feng Chia University, Taichung 407102, Taiwan
* Correspondence: leng@nchu.edu.cn

**Abstract:** As artificial intelligence increasingly automates the recognition and analysis of visual content, it poses significant risks to privacy, security, and autonomy. Computer vision systems can surveil and exploit data without consent. With these concerns in mind, we introduce a novel method to control whether images can be recognized by computer vision systems using reversible adversarial examples. These examples are generated to evade unauthorized recognition, allowing only systems with permission to restore the original image by removing the adversarial perturbation with zero-bit error. A key challenge with prior methods is their reliance on merely restoring the examples to a state in which they can be correctly recognized by the model; however, the restored images are not fully consistent with the original images, and they require excessive auxiliary information to achieve reversibility. To achieve zero-bit error restoration, we utilize the differential evolution algorithm to optimize adversarial perturbations while minimizing distortion. Additionally, we introduce a dual-color space detection mechanism to localize perturbations, eliminating the need for extra auxiliary information. Ultimately, when combined with reversible data hiding, adversarial attacks can achieve reversibility. Experimental results demonstrate that the PSNR and SSIM between the restored images by the method and the original images are ∞ and 1, respectively. The PSNR and SSIM between the reversible adversarial examples and the original images are 48.32 dB and 0.9986, respectively. Compared to state-of-the-art methods, the method maintains high visual fidelity at a comparable attack success rate.

**Keywords:** data security; reversible data hiding; reversible adversarial example; dual-color space; black-box attack

check for updates

Academic Editors: Ana Lucila Sandoval Orozco and Luis Javier García Villalba

Received: 18 December 2024
Revised: 13 January 2025
Accepted: 21 January 2025
Published: 23 January 2025

**Citation:** Yang, S.; Leng, L.; Chang, C.-C.; Chang, C.-C. Reversible Adversarial Examples with Minimalist Evolution for Recognition Control in Computer Vision. *Appl. Sci.* **2025**, *15*, 1142. https://doi.org/10.3390/app15031142

## 1. Introduction

In recent years, deep learning models have been extensively utilized across a diverse array of industries [1,2]. These models enable the rapid and efficient analysis of substantial quantities of data, thereby significantly improving convenience in both personal and professional contexts. Furthermore, advancements in computer technology have rendered social interactions more accessible, thereby encouraging a growing number of individuals to share their photographs on social networks. Nevertheless, the development of web scraping technologies has similarly surged. Traditional scraping techniques, when combined with deep learning models, can analyze and extract images or other information from social media with a high degree of accuracy and efficiency [3,4]. Regrettably, certain malevolent actors exploit these technologies to unlawfully acquire sensitive user information, including photographs, preferences, geographical locations, and social connections. This unauthorized and malevolent data acquisition constitutes a significant threat to user privacy, thereby complicating the exploration of defense mechanisms against such intrusions [5].

Szegedy et al. were the first to introduce the concept of adversarial examples (AEs), positing that these examples are generated by adding specific adversarial perturbations to the original images. These AEs can mislead deep learning models, resulting in incorrect outcomes [6]. In practical applications, AEs constitute a significant security threat to deep learning systems. On the other hand, there are advantageous aspects associated with AEs; for instance, social platforms can implement minor adversarial perturbations during user photo uploads to deter attackers attempting to exploit deep learning models for identification. This methodology can also be broadly applied in the domain of privacy protection [7,8] and in counteracting malicious algorithms [9–11].

Based on different attack conditions, adversarial attacks can typically be categorized into two types: white-box attacks and black-box attacks. White-box attacks allow complete access to the information of the deep learning model, including network architecture and weights. Such attacks are typically rapid and exhibit a high success rate. Conversely, black-box attacks typically restrict access to the outputs produced by the deep learning model or the corresponding probability scores. In black-box settings, adversarial attack algorithms are unable to access the internal details of the deep learning model, leading to comparatively slower attacks and reduced success rates. However, it is crucial to acknowledge that black-box attack conditions more accurately reflect real-world scenarios.

The Fast Gradient Sign Method (FGSM), proposed by Goodfellow et al., represents a classic white-box attack method [12]. This method utilizes the loss function and the sign function to rapidly generate adversarial perturbations. Following this, several researchers have performed further optimizations based on FGSM [13,14]. Furthermore, certain researchers have employed this classic white-box attack method to investigate the robustness of deep learning models [15,16].

In contrast to gradient-based methods employed in white-box attacks, black-box attack methods resemble a form of simulated optimization of the gradient of the loss function. The One-Pixel attack, proposed by Su et al., generates adversarial examples (AEs) by modifying a single pixel under black-box conditions, utilizing a differential evolution algorithm to optimize the solution [17]. Similarly, the Scratch attack, proposed by Jere et al. [18], generates distortions resembling scratches using a differential evolution algorithm for adversarial attack purposes. Ran et al. proposed a black-box attack method based on image quality assessment [19], aimed at generating adversarial samples with high image quality.

However, although the adversarial perturbations generated by these classic adversarial attack methods are typically minor, the degradation of image data remains an unavoidable reality. Even minor perturbations can result in failures in specific computer vision tasks, particularly in critical areas such as medical imaging, military applications, and digital forensics [20,21]. Data hiding technologies embed the secrets into a cover [22], while reversible data hiding (RDH) enables the exact restoration of the original image with the aid of the embedded auxiliary information. Reversible adversarial example (RAE) generation integrates AE generation with RDH technologies [23], inheriting the functions of AE generation while enabling the exact restoration of the original images.

Liu et al. [24] proposed two methods for generating reversible adversarial examples (RAEs) utilizing RDH technology in conjunction with several classic AE generation techniques. However, the significant distortions induced by the RDH algorithm when embedding auxiliary information within AEs can result in a loss of adversarial effectiveness, ultimately culminating in attack failure. Yin et al. [25] proposed a method for generating RAE using reversible image transformations, which avoids the failure of AEs due to the embedding of auxiliary information in RDH. Nonetheless, this method is not entirely reversible, and there may still be deviations in image recovery.

Zhang et al. [26] proposed a partially reversible adversarial attack method. This method integrates adversarial attacks and restoration models into a unified task, utilizing a dimensionality reduction technique to optimize the distribution of adversarial perturbations, thereby reducing restoration error while maximizing attack capability. Cao et al. proposed a method for generating RAEs, known as W-RAE [27], which transforms the task of generating RAEs into an image steganography task. This is accomplished by embedding a specific image watermark to generate RAEs. However, these methods can only approximately restore, rather than exactly restore, the original images.

This paper proposes a novel approach for RAE generation based on evolutionary algorithms to generate minimalist adversarial perturbations. The primary contributions are summarized as follows.

(1) We proposed a RAEs generation method that achieves zero-bit error, which inherits the functions of AEs and enables the exact restoration of original images. This facilitates recognition control in computer vision, and restricts the recognition capabilities of unauthorized systems.

(2) By introducing dual-color space detection of perturbed pixels (D-CSDPP), the perturbed pixel location can be automatically detected according to the difference between each pixel and its adjacent pixels, thereby the auxiliary information of perturbed pixel location do not need to be embedded, and embedding capacity is saved. Thereby, the image quality and the attack success rate (ASR) are both improved.

(3) Experimental validation demonstrates that the RAEs generated by the proposed method exhibit higher image quality compared to the original images and achieve a greater adversarial preservation rate (APR) relative to state-of-the-art (SOTA) methods.

## 2. Preliminary

This section provides a brief overview of one-pixel attacks and differential evolution algorithm, which are employed in our method.

### *2.1. One-Pixel Attack*

Typically, the modifications to AEs involve multiple perturbations that jointly alter the overall structure of the image, causing deep learning models to make incorrect judgments. In contrast, a one-pixel attack modifies the image using only a single perturbation.

The AE generation typically involves accumulating multiple perturbations until certain conditions are satisfied. However, in a one-pixel attack, the problem can be simplified to find the optimal modification pixel within the constraints of the entire image. By focusing on a small number of pixels, the AE can achieve the desired adversarial task without constraining the modification intensity.

### *2.2. Differential Evolution*

Differential evolution solves complex multimodal optimization problems. This algorithm relies on the variation within a population, and is particularly effective in black-box conditions compared with some gradient-based methods for white-box scenarios. Specifically, in each iteration, offspring individuals are generated from their parents, and then all of the offspring individuals and their parents are evaluated together; the individuals with higher likelihoods of survival (higher fitness values) are selected and preserved. This approach allows both parents and offspring individuals to pursue the goal of improving fitness, and maintains the diversity within the population.

Due to the absence of gradient-based iterations, the differential evolution algorithm does not require the target function to be differentiable. Consequently, it has a wider scope of optimization than gradient-based approaches. It has two main advantages for generating AEs, as follows.

(1) Global optimization enhancement.

Unlike gradient descent or greedy algorithms, which are limited by the constraints of the objective function and may converge to local optima, differential evolution algorithm is less affected by such limitations, and has higher likelihood of finding the global optimum, especially in highly challenging problems.

(2) Less information requirements.

Differential evolution algorithm does not rely on prior knowledge or information for optimization. This aspect is particularly important in the context of generating AEs. Firstly, some models are not rigidly differentiable, in which gradient-based methods are inapplicable. Secondly, some certain information is inaccessible during the optimization process.

In AE generation based on a one-pixel attack, a large number of iterations search for a single perturbed pixel that impacts the image structure. The differential evolution algorithm can relieve the limitations of local optima, and efficiently identify perturbed pixels. Moreover, in many real scenarios, we cannot access to the internal details of models. Employing the differential evolution algorithm to generate AEs can maintain a high ASR, even for black-box models. Based on these advantages, the proposed method in this paper utilizes the differential evolution algorithm.

## 3. The Proposed Method

The framework, illustrated in Figure 1, consists of two phases: the generation phase and the restoration phase. The processes represented by blue arrows indicate the RAE generation, while the processes represented by red arrows indicate the restoration of original image from its RAE.

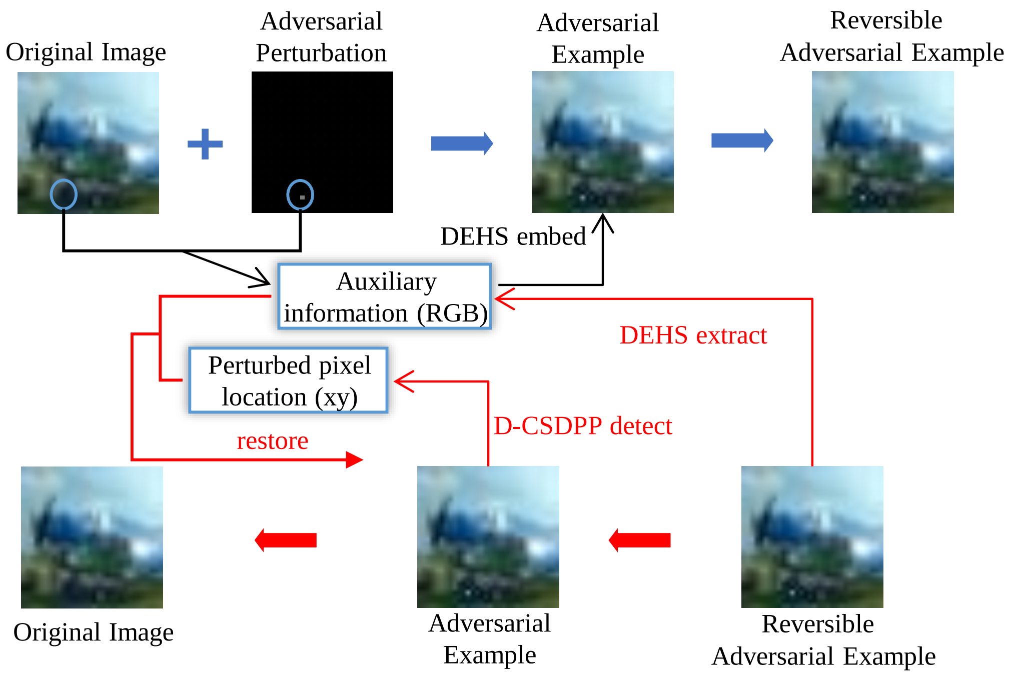

**Figure 1.** Framework for the generation and recovery of reversible adversarial examples.

In the generation phase, adversarial perturbations are generated using a differential evolution algorithm and added to the original image to obtain the AE. The original RGB values of the perturbed pixels are recorded and treated as auxiliary information. This recorded auxiliary information is then embedded into the AE using differential expanded histogram shifting (DEHS) to generated an RAE.

In the restoration phase, the embedded auxiliary information is extracted from the RAE using DEHS, thereby enabling the recovery of the AE. The D-CSDPP is employed to detect the locations $x$ and $y$ of the perturbed pixels in the AE. By utilizing the detected location information along with the auxiliary information, the AE can be restored to the original image with zero-bit error.

### *3.1. Adversarial Example Generation*

The adversarial perturbations are encoded as vectors (candidate solutions) and optimized by differential evolution algorithm. Each candidate solution contains a predefined number of perturbed pixels. Fore example, in a one-pixel attack, the number of perturbed pixels is 1. Each perturbed pixel has the modification value (one value in gray image, or three values of R, G and B in color image) and the coordinates $(x, y)$. The random numbers obeying normal distribution $N(\mu = 128, \sigma = 127)$ are the initial R, G, and B values of perturbed pixels; each solution can be represented as a vector $(x, y, R, G, B)$. The initial population has 400 candidate solutions. In each iteration, an additional 400 offspring candidate solutions are produced. The differential evolution formula is

$$\begin{aligned} x_i(t+1) = x_{r1}(t) + F\big(x_{r2}(t) - x_{r3}(t)\big), \\ \text{where } r1 \neq r2 \neq r3 \end{aligned} \tag{1}$$

where $x_i$ represents a candidate solution, i.e., the $i$-th perturbed pixel. $t$ denotes the iteration count, and $F$ is a scaling coefficient with a preset value 0.5, which restricts $x_{r1}(t), x_{r2}(t)$, and $x_{r3}(t)$. $r1, r2$, and $r3$ are three random indices of the selected parent individuals that produce the offspring individuals. Once the offspring individuals are produced, all of the offspring individuals and their parents are evaluated together, and the 400 individuals with higher likelihoods of survival (higher fitness values) are selected and preserved. Small size hinders the objective function from finding the ideal optimal solution, while large size increases the computation time. A size of 400 is sufficient to find good optimal solutions in most images, and the computational complexity is acceptable.

The image sizes of some datasets, e.g., ImageNet, are large. This can also be extended to multi-pixel attacks, where the number of perturbed pixels is larger than 1. This does not imply that altering just one pixel cannot execute an attack. If the computation is sufficient, i.e., the epoch number is large, even perturbing a single pixel can lead to a successful attack. In the case of $n$-pixel attack, RAE is fast, i.e., the epoch number is small, and each solution can be represented as a vector ($x_1$, $y_1$, $R_1$, $G_1$, $B_1$, $x_2$, $y_2$, $R_2$, $G_2$, $B_2$, …, $x_n$, $y_n$, $R_n$, $G_n$, $B_n$). Algorithm 1 presents the workflow for generating adversarial perturbation using differential evolution.

**Algorithm 1** Differential evolution for adversarial perturbation.

```
1:  Input:
      • Image $I$
      • Population size $N = 400$
      • Number of pixels to perturb $n$
      • Scaling factor $F = 0.5$
      • Maximum iterations $T$
2:  Initialize:
      • Population $P$ of size $N$
          – For 1-pixel attack: Each solution $x_i = (x, y, R, G, B)$
          – For $n$-pixel attack: Each solution $x_i = (x_1, y_1, R_1, G_1, B_1, \ldots, x_n, y_n, R_n, G_n, B_n)$
      • Initialize pixel values from $N(\mu = 128, \sigma = 127)$
3:  for $t = 1$ to $T$ do
4:      # Mutation Phase
5:      for each solution $x_i$ in $P$ do
6:          Randomly select unique $r_1, r_2, r_3$ from $P$
7:          $x_i(t+1) \leftarrow x_{r_1}(t) + F \cdot (x_{r_2}(t) - x_{r_3}(t))$
8:      end for
9:      # Crossover and Selection
10:     $Combined\_Population \leftarrow P + Offspring\_Population$
11:     Evaluate fitness of all solutions
12:     Select top 400 solutions with highest fitness
13:     Update $P$
14: end for
15: Return: best solution
```

## *3.2. Reversible Adversarial Example Generation*

One-pixel AE generation and multi-pixel AE generation share the same steps of RAE generation and original image restoration. The difference is that the lengths of the vectors, which represent the individuals, are 1 × 5 for one-pixel attacks and $n$ × 5 for multi-pixel attacks, respectively. For simplification, one-pixel AE generation is specified, which consists of two steps: auxiliary information encoding and data embedding.

### 3.2.1. Auxiliary Information Encoding

Each pixel can be represented as a vector $(x, y, R, G, B)$, $(x, y)$ represents the coordinates. If the image size is 32 × 32, $x, y \in [0, 31]$. $R, G, B$ represent the values in the red, green, and blue channels, respectively. $R, G, B \in [0, 255]$.

As shown in Figure 2, after one-pixel AE generation, the perturbed pixel is represented as a vector $(x', y', R', G', B')$. To restore the original image, we have to record and save the pixel values at the same position in the original image, i.e., $(R_0, G_0, B_0)$, where $(x_0, y_0) = (x', y')$. $(R_0, G_0, B_0)$ is converted from decimal to a fixed-length bit string. Three 8-bit codes are for $R_0, G_0, B_0$, respectively, and a 24-bit code is for them totally.

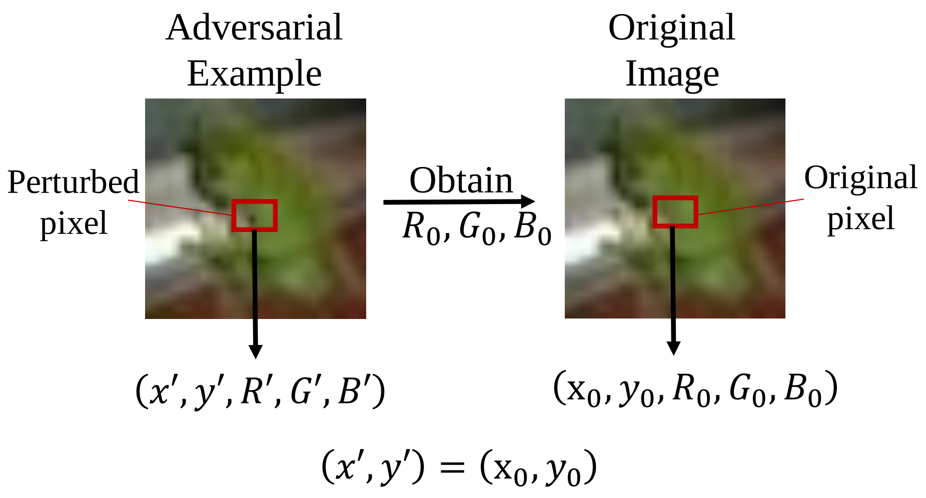

**Figure 2.** The process involves the obtaining and recording of auxiliary information (the original RGB values).

### 3.2.2. Data Embedding

In each channel, the difference in the AE is

$$\begin{gathered}\mathbf{S}_i(m,n) = \mathbf{I}'(m,n+1) - \mathbf{I}'(m,n) \\ 0 \le m \le M, 0 \le n \le N-1\end{gathered} \tag{2}$$

where $\mathbf{I}'$ represents the AE image, and $\mathbf{I}'(m,n)$ represents the pixel value at coordinates $(m,n)$ in the image. The image size is $M \times N$. The three difference matrices in the three channels are $\mathbf{S}_i$ where $i \in \{R, G, B\}$, respectively. The differential computation process is illustrated in Figure 3.

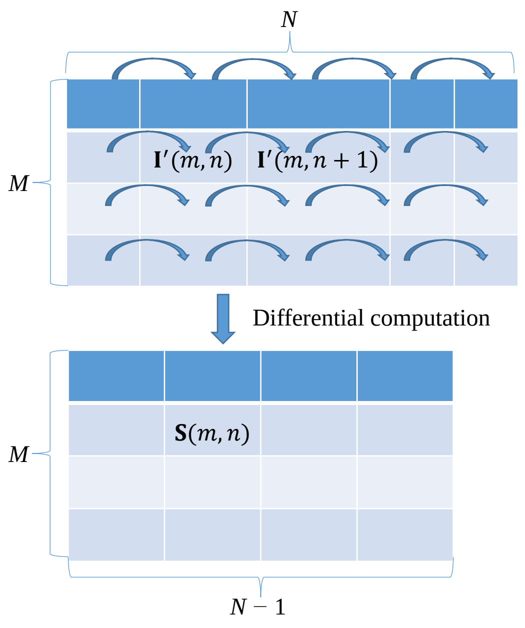

**Figure 3.** Differential computation process.

The histograms of $\mathbf{S}_i$ are generated. Figure 4 shows a histogram. The highest bar in the histogram represents the most frequent difference value (usually 0), and is denoted as $e_{\max}$, as shown in Figure 4a. The bars at the right of the highest bar are shifted to the right by one unit, as shown in Figure 4b. The binary information can be embedded into the empty space, as shown in Figure 4c.

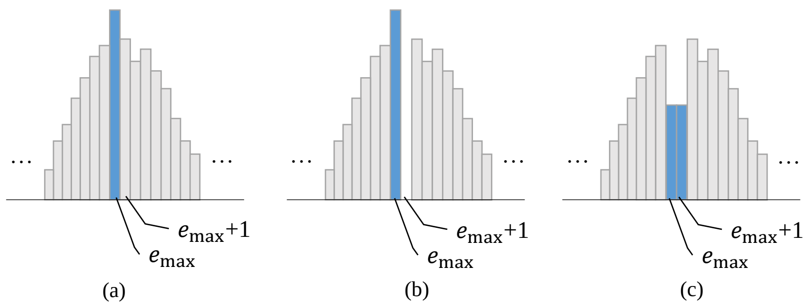

**Figure 4.** Histogram at each stage. (**a**–**c**) represent the histograms of the original image, the histogram after shifting, and the histogram after embedding, respectively.

The corresponding operation on $\mathbf{S}_i$ is to increase the values, which are greater than $e_{\max}$, and generate $\mathbf{S}_i'$ as

$$\mathbf{S}_i'(m,n) = \begin{cases} \mathbf{S}(m,n) & \text{if } \mathbf{S}(m,n) < e_{\max}+1 \\ \mathbf{S}(m,n)+1 & \text{others} \end{cases} \tag{3}$$

The first pixel with the value of $e_{\max}$ is found. If the embedded bit is 0, the value of the current $e_{\max}$-pixel is unchanged; if the embedded bit is 1, the value of the current $e_{\max}$-pixel is added by 1. The subsequent pixels outside $e_{\max}$-pixel in the same row are also added by 1 to maintain the difference between two adjacent pixels. The next $e_{\max}$-pixel is found and processed according to the next embedded bit. After data embedding, the matrices at three channels are $\mathbf{S}_i''$. Because the operations at three channels are the same, the subscript $i$ is omitted in the following equations. $\mathbf{S}''$ can be obtained according to Equation (4), while $k$ represents the value of the bit that is to be currently embedded.

$$\mathbf{S}''(m,n) = \begin{cases} \mathbf{S}'(m,n)+1 & \text{if } k=1 \\ \mathbf{S}'(m,n) & \text{if } k=0 \end{cases} \quad \text{if } \mathbf{S}'(m,n) = e_{\max} \tag{4}$$

The three matrices are added to the AE $\mathbf{I}'$, and we obtain the RAE $\mathbf{I}''$ as

$$\mathbf{I}''(m,n+1) = \mathbf{I}'(m,n) + \mathbf{S}''(m,n) \tag{5}$$

We take an instance to illustrate how the data are embedded into a difference histogram. In Figure 5, a $4 \times 4$ matrix represents block in a channel of an AE $\mathbf{I}'$. The difference matrix $\mathbf{S}$ is a $4 \times 3$ matrix, which is computed according to Equation (2). $e_{\max}$ is 0, and the $e_{\max}$-pixels are labeled by yellow color.

$\mathbf{S}'$ represents the values after histogram shifting in Figure 4. $\mathbf{S}''$ is obtained after the data "1010" are embedded. Finally, the RAE $\mathbf{I}''$ is obtained according to Equation (5). The changed pixels after data embedding are labeled by red color.

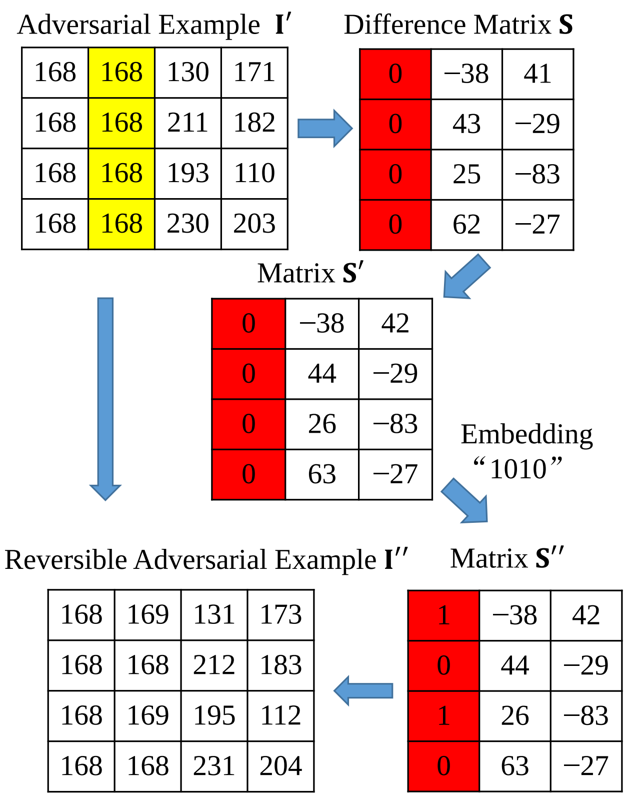

**Figure 5.** Generation of reversible adversarial example (a 4 × 4 black as an instance).

## 3.3. Original Image Restoration

Restoring the RAE back to the original image involves two steps. The first step is perturbed pixel detection, and the second step is data extraction and image restoration.

### 3.3.1. Dual-Color Space Detection of Perturbed Pixels

During generating the RAEs, only the original values of the perturbed pixels $(R_0, G_0, B_0)$ are recorded. Our method can automatically detect the perturbed pixel in an RAE, so the perturbed pixel location $(x_0, y_0)$ is discarded and not recorded, which saves the embedding capacity for the auxiliary information and mends the images short of embedding capacity.

The differences between the perturbed pixel and its adjacent pixels are remarkable, so we leverage the remarkable differences to detect perturbed pixel. In Figure 6, the RAE $\mathbf{I}''$ is split into six channels, with three channels based on the RGB channels and three channels based on the HSV channels: $\mathbf{I}''_i$ and $\mathbf{I}''_k$, where $i \in \{R, G, B\}, k \in \{H, S, V\}$. In fact, using only the RGB channels is sufficient to successfully detect perturbed pixel in the majority of RAEs. However, by incorporating the HSV channels, the detection success rate can be increased to 99%. In the following sections, we will also validate the feasibility of this approach through experiments.

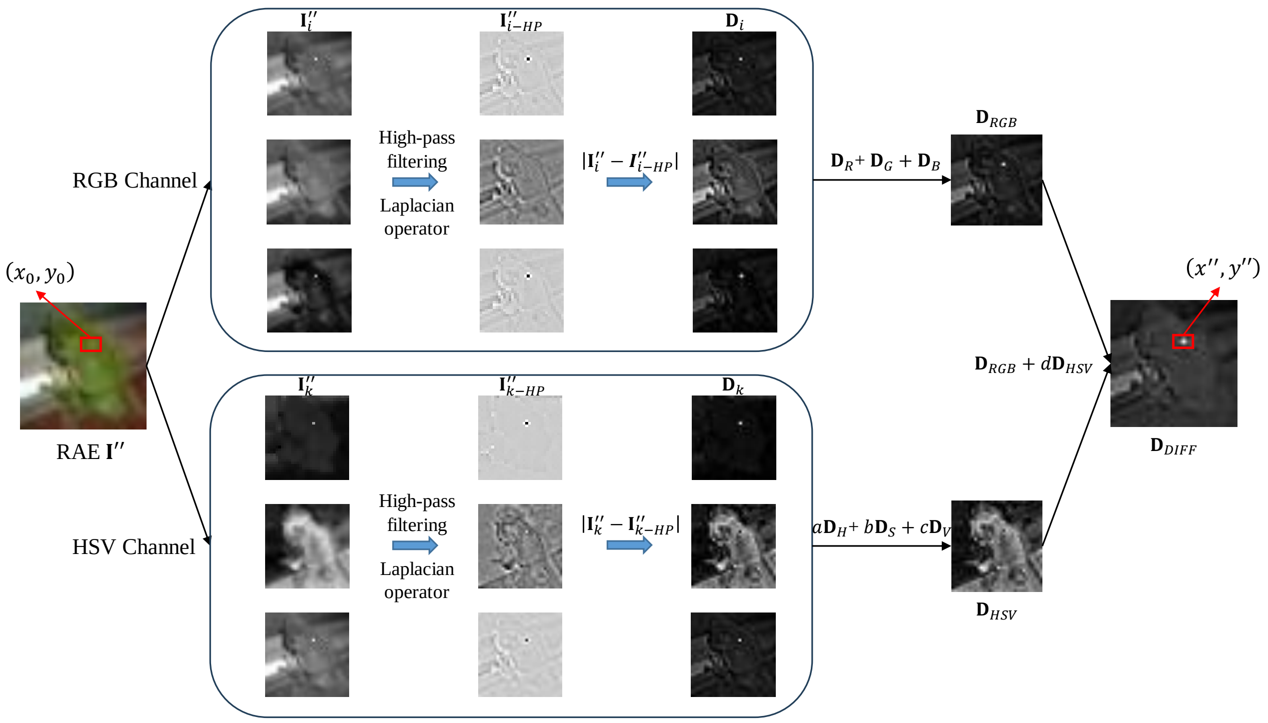

**Figure 6.** Visualization of perturbed pixel detection.

$$\mathbf{D}_{RGB} = \mathbf{D}_R + \mathbf{D}_G + \mathbf{D}_B \tag{6}$$

$$\mathbf{D}_{HSV} = a\mathbf{D}_H + b\mathbf{D}_S + c\mathbf{D}_V \tag{7}$$

$$\mathbf{D}_{DIFF} = \mathbf{D}_{RGB} + d\mathbf{D}_{HSV} \tag{8}$$

A high-pass filter, a Laplacian operator, is applied on each channel to generate six response matrices: $\mathbf{I}_{(i-HP)}''$ and $\mathbf{I}_{(k-HP)}''$. In Figure 6, for visualization, the matrices have been normalized. The actual differences between the perturbed pixel and its adjacent pixels are larger than the visualization results.

Next, the generated matrices are subtracted from their respective channels, and the absolute values of the differences are $\mathbf{D}_i = |\mathbf{I}_i'' - \mathbf{I}_{(i-HP)}''|$ and $\mathbf{D}_k = |\mathbf{I}_k'' - \mathbf{I}_{(k-HP)}''|$.

Finally, two kinds of difference matrices, $\mathbf{D}_i$ and $\mathbf{D}_k$, are summed to obtain $\mathbf{D}_{RGB}$ and $\mathbf{D}_{HSV}$ according to Equations (6) and (7), where $a = 0$, $b = 0.95$ and $c = 0.05$ in our work. The difference matrix $\mathbf{D}_{DIFF}$ is obtained according to Equation (8) where, in our work, $d = 1.5$. The location of the pixel with the highest value in $D_{\mathrm{DIFF}}$ is $(x'', y'')$, which is labeled by the red box, and considered as the location of the detected perturbed pixel in $\mathbf{I}''$. $(x_0, y_0)$ is the location of the perturbed pixel in the original image. Generally, $(x'', y'') = (x_0, y_0)$. The workflow of the D-CSDPP is shown in Algorithm 2.

**Algorithm 2** Dual-color space detection of perturbed pixels.

```
1:  **Input:** RAE Image $I''$
2:  **Steps:**
3:  # Split Image into Channels
4:  $I''_R, I''_G, I''_B \leftarrow$ RGB Channels
5:  $I''_H, I''_S, I''_V \leftarrow$ HSV Channels
6:  Apply High-Pass Laplacian Filter
7:  **for** each channel **do**
8:      # Generate high-pass response matrices
9:      Create $I''_{i-HP}$ for RGB
10:     Create $I''_{k-HP}$ for HSV
11: **end for**
12: # Calculate Difference Matrices
13: $D_i \leftarrow |I''_i - I''_{i-HP}|$ for RGB
14: $D_k \leftarrow |I''_k - I''_{k-HP}|$ for HSV
15: # Combine Difference Matrices
16: $D_{\text{RGB}} \leftarrow D_R + D_G + D_B$
17: $D_{\text{HSV}} \leftarrow a \cdot D_H + b \cdot D_S + c \cdot D_V$
18: $D_{\text{DIFF}} \leftarrow D_{\text{RGB}} + d \cdot D_{\text{HSV}}$
19: # Locate Perturbed Pixel
20: Find $(x'', y'')$ with max value in $D_{\text{DIFF}}$
21: $(x'', y'') = (x_0, y_0)$
22: **Output:** Detected Perturbed Pixel Location $(x'', y'')$
```

### 3.3.2. Data Extraction and Image Restoration

Data extraction and data embedding are inverse operations. As shown in Figure 7, first, the RAE $\mathbf{I}''$ is used to generate the difference matrix $\mathbf{S}''$ in the same way as that in embedding. Next, $\mathbf{S}''$ is scanned in the same order as that in embedding. If a pixel with the value of $e_{\max}$ is found, the extracted bit is "0", while if a pixel with the value of $e_{\max}+1$ is found, the extracted bit is "1". In this way, all the embedded data "1010" are extracted. Finally, we restore all of the differences in the matrix $\mathbf{S}''$ that are equal to $e_{\max}+1$ back to $e_{\max}$, and shift the histogram differences larger than $e_{\max}+1$ to the left by one unit to restore them. In this way, we obtain the matrix $\mathbf{S}$. The AE $\mathbf{I}'$ is obtained through the matrix $\mathbf{S}$ and RAE $\mathbf{I}''$,

$$\mathbf{S}'(m,n) = \begin{cases} \mathbf{S}''(m,n) - 1 & \text{if } \mathbf{S}''(m,n) = e_{\max}+1 \\ \mathbf{S}''(m,n) & \text{others} \end{cases}$$
$$\mathbf{S}(m,n) = \begin{cases} \mathbf{S}'(m,n) - 1 & \text{if } \mathbf{S}''(m,n) > e_{\max}+1 \\ \mathbf{S}'(m,n) & \text{others} \end{cases} \quad (9)$$

$$\begin{cases} \mathbf{I}'(m,1) = \mathbf{I}''(m,1) \\ \mathbf{I}'(m,n+1) = \mathbf{I}'(m,n) + \mathbf{S}(m,n) \end{cases} \quad (10)$$

The location and original values of the perturbed pixel are known. The location $(x_0, y_0)$ is detected, the extracted data $(R_0, G_0, B_0)$ are extracted. Finally, the RAE is restored to the original image without any loss.

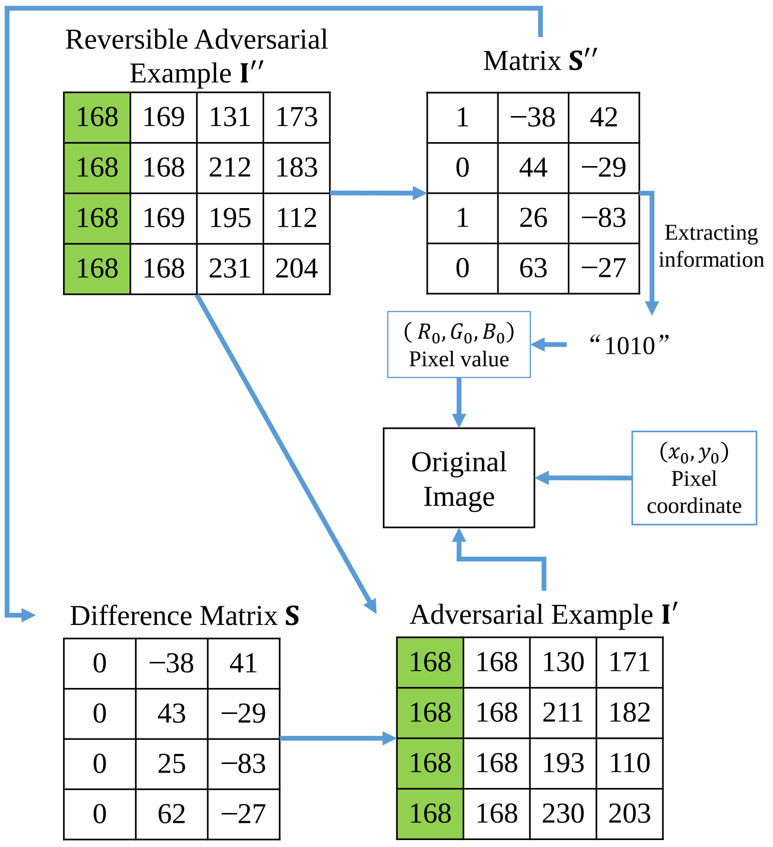

**Figure 7.** Data extraction and image restoration (the pixels filled with green color remain unchanged).

## 4. Experiments and Discussions

### *4.1. Experiment Setting*

The experiments were conducted on two datasets, CIFAR-10 [28] and ImageNet [29]. The CIFAR-10 dataset comprises 60,000 color images of 10 categories, with 6000 images per category. On CIFAR-10 dataset, one-pixel attack (with one perturbed pixel) is performed on some classical classification network models, namely LeNet [30], ResNet [31], and DenseNet [32]. A set of 1000 images are randomly selected from those correctly classified by the target models (attacked models).

On ImageNet dataset, multiple-pixel attack (with more than one perturbed pixels) is performed. The target model is MobileNet [33], Inception v3 [34], and Inception-ResNet v2 [35]. A set of 1000 images are randomly selected from those correctly classified by the target model.

All experiments were conducted under black-box conditions, meaning that the internal parameters of the target model cannot be accessed.

### *4.2. Comparison of Perturbation Pixel Detection Approaches*

As discussed in the previous section, the effectiveness of the perturbation pixel detection approach directly affects the ability of RAEs to be perfectly restored to the original image. Consequently, we conducted a comparative analysis of several approaches to validate and illustrate the effectiveness of the proposed D-CSDPP.

As shown in Table 1, by applying high-pass filtering separately to the RGB channels and calculating the difference matrix relative to the original channels (using the same method as described in Figure 6, which will not be elaborated on further), 88.42% of the RAEs to successfully detect the positions of the perturbed pixels. Utilizing only the HSV channels results in a detection success rate of 91.16%. Furthermore, if we simply combine the six channels, the success rate rises to 97.66%. By using the proposed weighted summation method, the success rate can be increased to 98.85%. Additionally, we conducted experiments utilizing a median filter similar to the Laplacian operator.However, this method proved ineffective (with a success rate of 0) because the salt-and-pepper noise present in the image is excessively amplified during this process. Figure 8 shows the visual results of the difference matrices generated by different approaches. Since the final difference matrix of each approach is obtained by summing multiple channel difference matrices, its maximum values exceed 255. Therefore, the value range is stretched to [0, 255] for visual effects.

**Table 1.** Comparison of the different perturbation pixel detection approach.

| Approaches | RGB | HSV | RGB + HSV | Proposed | RGB + HSV * |
|---|---|---|---|---|---|
| Successful Rate | 88.42% | 91.16% | 97.66% | **98.95%** | 0 |

* Use a 3 × 3 median filter to replace the Laplacian operator. The best results are shown in bold. The same applies to the tables that follow.

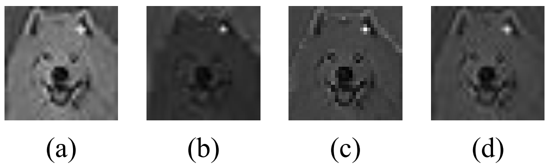

**Figure 8.** Visual results of the difference matrices generated by different approaches. (**a**–**d**) correspond to RGB, HSV, RGB + HSV, and Proposed, respectively.

### 4.3. Image Quality

Figure 9 shows the visual results of proposed method. The target models for the visual experiments are ResNet (based on CIFAR-10) and MobileNet (based on ImageNet). In the ImageNet experiment, only three pixels were perturbed, resulting in generated AE and RAE with visually indistinguishable effects. The restored image is identical to the original image, hence the intermediate process is not shown in the figure. The high image quality of RAEs can also be observed from the visual results on both the CIFAR-10 and ImageNet.

As shown in Table 2, the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index Measure (SSIM) are calculated between the original images and their corresponding RAEs, as well as between the original image and the restored image. The experimental setup involves attacking the MobileNet on the ImageNet dataset, with the number of perturbed pixels set to 3, 5, and 10, respectively.

**Table 2.** Results of the PSNR and SSIM calculations between the original image and the RAE, as well as between the original image and the restored image.

| Pixels | | Compared with RAE | Compared with Restored |
|---|---|---|---|
| 3 | PSNR(dB) | **48.32** | ∞ |
| | SSIM | **0.9986** | **1** |
| 5 | PSNR(dB) | 46.65 | ∞ |
| | SSIM | 0.9977 | **1** |
| 10 | PSNR(dB) | 43.97 | ∞ |
| | SSIM | 0.9959 | **1** |

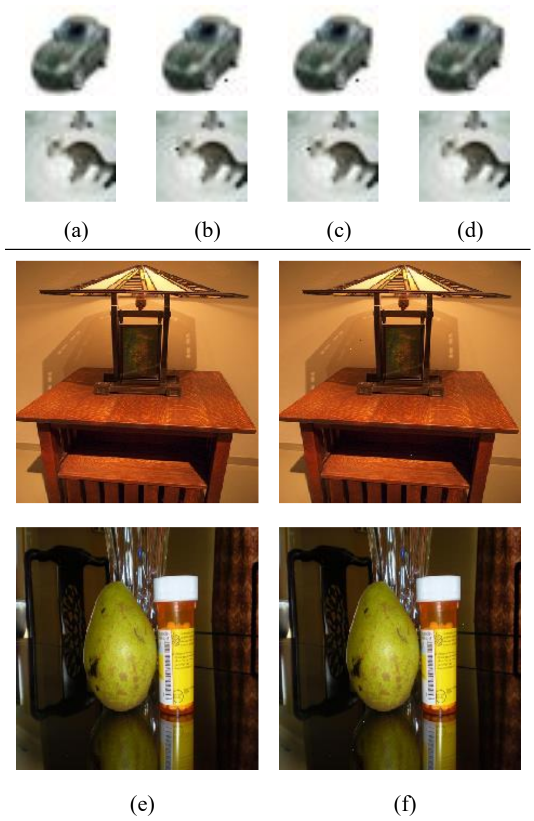

**Figure 9.** Visual results of proposed method. (**a**–**d**) based on CIFAR-10. (**a**) Original image. (**b**) AE. (**c**) RAE. (**d**) Restored image; (**e**,**f**) based on ImageNet. (**e**) Original image. (**f**) RAE.

When only three pixels are allowed to be perturbed, the RAE generated by the proposed method achieves a PSNR of 48.32 dB and an SSIM of 0.9986 when compared to the original image. This indicates that the generated RAEs possess high image quality. Moreover, even with 10 perturbed pixels, a PSNR of 43.97 and an SSIM of 0.9959 can still be obtained. The RAEs generated by the proposed method can be restored perfectly to the original image, resulting in a PSNR of ∞ and an SSIM of 1 when the restored image is compared to the original image. This indicates that the proposed method is a reversible algorithm exhibiting a zero-bit error.

## 4.4. Attack Performance

Several metrics are used to evaluate the attack performance, including

$$ASR_A = \frac{n(AE)}{N} \tag{11}$$

$$ASR_{RA} = \frac{n(RAE)}{N} \tag{12}$$

$$APR = \frac{n(RAE)}{n(AE)} \tag{13}$$

where $n(AE)$ is the number of attack-successful AEs, and $n(RAE)$ is the number of attack-successful RAEs. $N$ is the total count of samples. In our work, $N = 1000$. The data embedding should not compromise the attack performance, thus necessitating a close proximity between $ASR_{RA}$ and $ASR_A$.

### 4.4.1. CIFAR-10 Dataset

Table 3 shows the attack performances on different classification networks.

**Table 3.** Attack performances on different models; "o" and "w" represent without and with D-CSDPP on CIFAR-10.

| Model | Method | $ASR_A$ | $ASR_{RA}\uparrow$ | $APR\uparrow$ |
|---|---|---|---|---|
| LeNet | o | 66.50% | 44.04% | 63.60% |
| | w | | **57.02%** | **81.89%** |
| ResNet | o | 41.02% | 24.10% | 51.29% |
| | w | | **35.10%** | **75.32%** |
| DenseNet | o | 24.02% | 12.80% | 50.29% |
| | w | | **19.08%** | **72.93%** |

↑ indicates that higher values are better. The same applies to the tables that follow.

In Table 3, the o methods achieve the APRs of 63.60%, 51.29%, and 50.29% on the three models, respectively. The w methods improve these rates to 81.89%, 75.32%, and 72.93%, an increase of approximately 20%. The main reason for this improvement is that, due to the reduction in embedded data, the number of samples with sufficient capacity is significantly increased, resulting in an increase in the number of successful attacks.

D-CSDPP reduces the amount of the data that need to be embedded. Without D-CSDPP, $(x, y, R, G, B)$, i.e., the location information and the pixel values of the perturbed pixels, need to be embedded. With D-CSDPP, location information can be automatically detected and does not need to be embedded. Thus, D-CSDPP reduces the amount of embedding data, while allows an increase in the number of attack pixel, and leads to higher APRs.

### 4.4.2. ImageNet Dataset

More perturbed pixels are needful for attacking large-size images, such as ImageNet. As shown in Table 4, when the number of modified pixels is 3, 10, and 50, $APR$ is 100%, 99.63%, and 87.72%, respectively. Multi-pixel attacks are more robust when sufficient hiding capacity is available. More perturbed pixels can help large-size images change their overall structures, and mislead the models to make classification errors.

**Table 4.** Attack performances with different numbers of perturbed pixels on ImageNet.

| Model | Pixels | $ASR_A$ | $ASR_{RA}\uparrow$ | $APR\uparrow$ |
|---|---|---|---|---|
| MobileNet | 3 | 25.30% | 25.30% | **100.0%** |
| | 10 | 32.60% | 31.50% | 99.63% |
| | 50 | 39.10% | **34.40%** | 87.72% |

APR does not necessarily increase with the increment of the number of perturbed pixels. The differential evolution algorithm optimizes a solution vector composed of allowable perturbed pixels. Regardless of the number of allowable perturbed pixels, the ultimate goal is to achieve a certain level of overall perturbation. Meanwhile, an increase in the number of perturbed pixels also increases the amount of auxiliary information that needs to be embedded in the AE. Therefore, the number of perturbed pixels should be appropriate to satisfy the sufficient hiding capacity condition as much as possible, which can achieve a higher APR.

### *4.5. Comparison with State-of-the-Art Methods*

The proposed method perturbs 50 pixels and compares with the post and in-the-loop methods based on the BIM attack proposed in [24], on the ImageNet dataset, as shown in Table 5.

**Table 5.** Comparison with state-of-the-art methods.

| Method | Model | $ASR_A$ | $ASR_{RA} \uparrow$ | $APR \uparrow$ |
|---|---|---|---|---|
| Proposed | Inc-V3 | 27.20% | 25.50% | **93.75%** |
| | IncRes-V2 | 27.60% | 26.10% | **94.57%** |
| Post [24] | Inc-V3 | 30.82% | 19.92% | 64.63% |
| | IncRes-V2 | 37.23% | 24.60% | 66.08% |
| In-the-loop [24] | Inc-V3 | 30.82% | 25.76% | 83.58% |
| | IncRes-V2 | 37.23% | **30.91%** | 83.02% |

Both the method in [24] and the proposed method are evaluated based on the effectiveness of their $ASR_A$, which is fundamentally rooted in the adversarial attack method, i.e., BIM attack or differential evolution. In the proposed method, allocating additional computational resources can lead to better $ASR_A$ performance. However, this approach may not significantly impact the performance of reversible adversarial attack methods. Because the introduction of reversible algorithms in previous method designs may lead to a significant number of AEs failing, minimizing such occurrences is crucial for enhancing the RAE generation algorithm. What we should focus on is how many AEs can retain their adversarial attribute and become RAEs throughout the complete process, i.e., APR performance.

On the Inc-Res V2 model, our proposed method APR achieves 94.57%, which is a significant improvement compared to 66.08% of the post method and 83.02% of the in-the-loop method in [24]. This is because, in each step of the algorithm, we strive to ensure that the image quality is not excessively degraded. Additionally, the performance of the proposed method on ASR is also commendable, i.e., our method achieves an ASR of 25.50% and 26.10% on two target models, which is better than 19.92% and 24.60% of the post method in [24]. Compared to the in-the-loop method in [24], the proposed method exhibits only a slight disadvantage in ASR on the IncRes-V2 model. However, this discrepancy can be partly attributed to the higher success rate of generating AEs using the method in [24], which stands at 37.23% for the IncRes-V2 model compared to our 27.60%. Their method adopts a more aggressive strategy to achieve a higher success rate which, however, results in a greater loss of image quality during the conversion from AEs to RAEs.

## 5. Conclusions and Future Works

A novel reversible adversarial example generation method is proposed under black-box conditions. A differential evolution algorithm is utilized to generate minimal adversarial perturbations on the original image. RAEs are generated based on D-CSDPP algorithm. This method not only mislead the deep learning models in image classification tasks, but also allows the RAEs to be exactly restored to the original image. This feature is lacking in many existing methods. Furthermore, this method enables recognition control in computer vision. The proposed reversible method functions as a form of encryption for computer vision. Thus, only authorized models are allowed to recognize the images.

The D-CSDPP can automatically detect the perturbed pixels, meaning that location information is not needed for original image restoration, resulting in a decrease in embedded data. As a result, APR is improved by more than 20%. Comparative experiments with different approaches demonstrate the effectiveness and detection performance of the proposed D-CSDPP.

The PSNR and SSIM between the RAEs and the original images can reach up to 48.32 dB and 0.9986, respectively, indicating that the images generated by the proposed method are of high quality. This is also reflected in the visual results of the RAEs.

The proposed method demonstrates effectiveness across various advanced models and datasets, highlighting its generalizability. Compared to SOTA methods, the proposed method achieves superior APR due to the high image quality of the RAEs.

In the future, we will attempt to propose some methods for generating more robust and efficient adversarial perturbations. We will also try to enhance the efficiency of the optimization solving. Alternatively, we aim to further compress the required auxiliary information for image restoration.

**Author Contributions:** Conceptualization, L.L. and S.Y.; methodology, L.L., S.Y., C.-C.C. (Chin-Chen Chang) and C.-C.C. (Ching-Chun Chang); software, S.Y.; validation, S.Y.; formal analysis, L.L., S.Y., C.-C.C. (Chin-Chen Chang) and C.-C.C. (Ching-Chun Chang); investigation, S.Y.; resources, S.Y.; data curation, S.Y.; writing—original draft preparation, S.Y.; writing—review and editing, L.L., S.Y., C.-C.C. (Chin-Chen Chang) and C.-C.C. (Ching-Chun Chang); visualization, S.Y.; supervision, L.L., C.-C.C. (Chin-Chen Chang) and C.-C.C. (Ching-Chun Chang); project administration, L.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This study was funded by National Natural Science Foundation of China (62466038), Jiangxi Provincial Key Laboratory of Image Processing and Pattern Recognition (2024SSY03111), Technology Innovation Guidance Program Project (Special Project of Technology Cooperation, Science and Technology Department of Jiangxi Province) (20212BDH81003), and the Innovation Foundation for Postgraduate Students of Nanchang Hangkong University (YC2023-102).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is contained within the article.

**Acknowledgments:** This study was supported by National Natural Science Foundation of China (62466038), Jiangxi Provincial Key Laboratory of Image Processing and Pattern Recognition (2024SSY03111), Technology Innovation Guidance Program Project (Special Project of Technology Cooperation, Science and Technology Department of Jiangxi Province) (20212BDH81003), and the Innovation Foundation for Postgraduate Students of Nanchang Hangkong University (YC2023-102).

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

## References

1. Patil, D.; Rane, N.; Desai, P.; Rane, J. Machine learning and deep learning: Methods, techniques, applications, challenges, and future research opportunities. In *Trustworthy Artificial Intelligence in Industry and Society*; Deep Science Publishing: Palo Alto, CA, USA, 2024; pp. 28–81.
2. Mardieva, S.; Ahmad, S.; Umirzakova, S.; Rasool, M.A.; Whangbo, T.K. Lightweight image super-resolution for IoT devices using deep residual feature distillation network. *Knowl.-Based Syst.* **2024**, *285*, 111343. [CrossRef]
3. Shrivastava, G.K.; Pateriya, R.K.; Kaushik, P. An efficient focused crawler using LSTM-CNN based deep learning. *Int. J. Syst. Assur. Eng. Manag.* **2023**, *14*, 391–407. [CrossRef]
4. Conti, A.; Fini, E.; Mancini, M.; Rota, P.; Wang, Y.; Ricci, E. Vocabulary-free image classification. *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 30662–30680.
5. Zhang, J.; Wang, J.; Wang, H.; Luo, X.; Ma, B. Trustworthy adaptive adversarial perturbations in social networks. *J. Inf. Secur. Appl.* **2024**, *80*, 103675. [CrossRef]
6. Szegedy, C. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
7. Li, X.; Chen, L.; Wu, D. Turning attacks into protection: Social media privacy protection using adversarial attacks. In Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), Virtual Event, 29 April–1 May 2021; SIAM: Philadelphia, PA, USA, 2021; pp. 208–216.
8. Li, X.; Chen, L.; Wu, D. Adversary for social good: Leveraging adversarial attacks to protect personal attribute privacy. *ACM Trans. Knowl. Discov. Data* **2023**, *18*, 1–24. [CrossRef]

9. Wang, Z.; Wang, H.; Jin, S.; Zhang, W.; Hu, J.; Wang, Y.; Sun, P.; Yuan, W.; Liu, K.; Ren, K. Privacy-preserving adversarial facial features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 8212–8221.
10. Kumar, C.; Ryan, R.; Shao, M. Adversary for social good: Protecting familial privacy through joint adversarial attacks. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11304–11311.
11. Zhang, J.; Sang, J.; Zhao, X.; Huang, X.; Sun, Y.; Hu, Y. Adversarial privacy-preserving filter. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 1423–1431.
12. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
13. Kurakin, A.; Goodfellow, I.J.; Bengio, S. Adversarial examples in the physical world. *arXiv* **2016**, arXiv:1607.02533.
14. Wang, Y.; Liu, J.; Chang, X.; Wang, J.; Rodríguez, R.J. AB-FGSM: AdaBelief optimizer and FGSM-based approach to generate adversarial examples. *J. Inf. Secur. Appl.* **2022**, *68*, 103227. [CrossRef]
15. Lupart, S.; Clinchant, S. A study on FGSM adversarial training for neural retrieval. In Proceedings of the European Conference on Information Retrieval, Dublin, Ireland, 2–6 April 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 484–492.
16. Elsheikh, R.A.; Mohamed, M.; Abou-Taleb, A.M.; Ata, M.M. Accuracy is not enough: A heterogeneous ensemble model versus FGSM attack. *Complex Intell. Syst.* **2024**, *10*, 8355–8382. [CrossRef]
17. Su, J.; Vargas, D.V.; Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.* **2019**, *23*, 828–841. [CrossRef]
18. Jere, M.; Rossi, L.; Hitaj, B.; Ciocarlie, G.; Boracchi, G.; Koushanfar, F. Scratch that! An evolution-based adversarial attack against neural networks. *arXiv* **2019**, arXiv:1912.02316.
19. Ran, Y.; Zhang, A.X.; Li, M.; Tang, W.; Wang, Y.G. Black-box adversarial attacks against image quality assessment models. *Expert Syst. Appl.* **2025**, *260*, 125415. [CrossRef]
20. Bacci, N.; Briers, N.; Steyn, M. Prioritising quality: Investigating the influence of image quality on forensic facial comparison. *Int. J. Leg. Med.* **2024**, *138*, 1713–1726. [CrossRef]
21. Ahmed, M.T.; Islam, R.; Rahman, M.A.; Islam, M.J.; Rahman, A.; Kabir, S. An image-based digital forensic investigation framework for crime analysis. In Proceedings of the 2023 International Conference on Next-Generation Computing, IoT and Machine Learning (NCIM), Gazipur, Bangladesh, 16–17 June 2023; IEEE: New York, NY, USA, 2023; pp. 1–6.
22. Gong, L.H.; Luo, H.X. Dual color images watermarking scheme with geometric correction based on quaternion FrOOFMMs and LS-SVR. *Opt. Laser Technol.* **2023**, *167*, 109665. [CrossRef]
23. Feng, Q.; Leng, L.; Chang, C.C.; Horng, J.H.; Wu, M. Reversible data hiding in encrypted images with extended parametric binary tree labeling. *Appl. Sci.* **2023**, *13*, 2458. [CrossRef]
24. Liu, J.; Zhang, W.; Fukuchi, K.; Akimoto, Y.; Sakuma, J. Unauthorized AI cannot recognize me: Reversible adversarial example. *Pattern Recognit.* **2023**, *134*, 109048. [CrossRef]
25. Yin, Z.; Wang, H.; Chen, L.; Wang, J.; Zhang, W. Reversible adversarial attack based on reversible image transformation. *arXiv* **2019**, arXiv:1911.02360.
26. Zhang, J.; Wang, J.; Wang, H.; Luo, X. Self-recoverable adversarial examples: A new effective protection mechanism in social networks. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *33*, 562–574. [CrossRef]
27. Cao, X.; Liu, J.; Yin, J.; Cheng, X.; Li, J.; Ma, H.; Luo, G. Reversible Adversarial Examples based on Self-Embedding Watermark for Image Privacy Protection. In Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, 30 June–5 July 2024; IEEE: New York, NY, USA, 2024; pp. 1–8.
28. Krizhevsky, A.; Hinton, G. Convolutional deep belief networks on cifar-10. (Unpublished manuscript). **2010**, *40*, 1–9.
29. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: New York, NY, USA, 2009; pp. 248–255.
30. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
32. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
33. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Efficient convolutional neural networks for mobile vision applications. *Mobilenets* **2017**, *10*, 151.

34. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
35. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31.