Article

High-Capacity Image Steganography Based on Improved Xception

Xintao Duan 1,2,*, Mengxiao Gou 1, Nao Liu 1, Wenxin Wang 1 and Chuan Qin 3

1 College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China; gou5695761@163.com (M.G.); liuao18437982698@126.com (N.L.); wangwenxin3023@126.com (W.W.)
2 Laboratory of Artificial Intelligence and Personalized Learning in Education of Henan Province, Xinxiang 453007, China
3 School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China; qin@usst.edu.cn
* Correspondence: duanxintao@htu.edu.cn

Received: 3 November 2020; Accepted: 14 December 2020; Published: 17 December 2020

Abstract: The traditional cover modification steganography method only has low steganography ability. We propose a steganography method based on the convolutional neural network architecture (Xception) of deep separable convolutional layers in order to solve this problem. The Xception architecture is used for image steganography for the first time, which not only increases the width of the network, but also improves the adaptability of network expansion, and adds different receiving fields to carry out multi-scale information in it. By introducing jump connections, we solved the problems of gradient dissipation and gradient descent in the Xception architecture. After cascading the secret image and the mask image, high-quality images can be reconstructed through the network, which greatly improves the speed of steganography. When hiding, only the secret image and the cover image are cascaded, and then the secret image can be embedded in the cover image through the hidden network in order to obtain the secret image. After extraction, the secret image can be reconstructed by bypassing the secret image through the extraction network. The results show that the results that are obtained by our model have high peak signal-to-noise ratio (PSNR) and structural similarity (SSIM), and the average high load capacity is 23.96 bpp (bit per pixel), thus realizing large-capacity image steganography surgery.

Keywords: image steganography; deep separable convolutional neural network; deep learning; Xception; ResNet

1. Introduction

Information globalization has become an inevitable trend in the context of the rapid development of computer technology. The same security issues of sensitive information are bound to face huge challenges, and information security is very important. For example, personal private information, medical patient information, commercial business data, military secrets, etc., will have serious consequences if they are leaked. Information security measures are constantly being strengthened in the face of such challenges.

In the early days of information security, encryption technology was used to protect the security of information. Encryption technology makes secret information incomprehensible, but it is also prone to suspicion. Image steganography hides one image in another image in order to make the secret information invisible, thus protecting the communication channel and hiding the communication behavior. The most commonly used cover in information hiding is images. The reasons are as follows: (1) in the context of information technology, images will flow into the Internet on a large scale and be
stored. This can provide a large number of samples for information hiding. (2) When compared with text and audio, pictures have more data storage capacity and greater data redundancy, which makes information hiding easier to hide and extract secret information that needs to be protected.

As one of the important methods of image information security protection, image steganography aims to hide sensitive images, which is, secret images, in its irrelevant cover data, and does not affect the cover's sensory experience, so as to hide information without causing more attention and awareness. Traditional information hiding is based on the continuous improvement of the STC framework [1]. For example, HUGO steganography with a high degree of undetectability [2], Wavelet-Obtained Weights (WOW) steganography [3], JPEG general wavelet relative distortion (J-UNIWARD) steganography [4], a steganography of concatenated codes composed of Syntrome-Trellis codes (STC), and cyclic redundancy codes (CRC) codes [5], and the more novel scheme [6] uses the decentralized and fair hierarchical threshold secret sharing scheme of blockchain to divide the secret into several shares and assign it to different levels of participants; any small subset can get the secret. There are some shortcomings that require experts in this area to manually design the cost function, which requires attention to the fitting problem while also minimizing the loss; although this works, it is very troublesome. Therefore, using deep learning to design the cost function of steganography has become a novel and reliable steganography scheme. Meanwhile, He et al. [7] proposed a residual learning framework in order to simplify network training deeper than previously used networks and solve the problem of difficulty in training networks with larger depths, indicating that these residual networks are easier to optimize. Additionally, the accuracy can be obtained by increasing the depth. This scheme greatly promotes the wide range of use of neural networks in steganography, and it can maintain a certain accuracy. At present, image steganography that is based on deep learning mainly includes three aspects: image steganography based on Convolutional Neural Network (CNN) [8], image based on Generative Adversarial Networks (GAN) [9] Steganography, and other network structure. Yang et al. [10] proposed a 32-layer convolutional neural network (CNN) in order to improve the efficiency of preprocessing and reusing features by cascading all the features in the previous layers with the same feature map size, thereby improving Information flow and gradient can further reduce detection errors. Chen et al. [11] introduced advanced ideas, such as skip connection and feature fusion, and designed a novel two-channel deep hidden network (TDHN). Wu et al. [12] introduced an attention mechanism in steganography to alleviate security issues and increase the load capacity.

In this work, we use an improved [13] framework, but removed the preprocessing network, and replaced the hidden network with our improved deep separable convolutional Xception network [14] in order to achieve the same size as the cover image and the secret image with the same number of channels (three channel color image or one channel gray image) can be perfectly embedded in the cover image with only slight pixel distortion. The secret image can be steganographically written in all bits and channels of the cover image. The experiments show that our proposed scheme not only effectively improves the steganography capacity, but also the peak signal-to-noise ratio (PSNR) [15] and structural similarity (SSIM) [16] are relatively high.

The main contributions of this article are, as follows:

1. New plan, we use the improved Xception network for the first time in order to realize image steganography by constructing hidden networks and extraction networks.
2. Improve Xception to speed up convergence, we have added jump connections to the Xception network to increase the retention of high-dimensional features of the image to speed up the convergence rate when the image is hidden.
3. High capacity and high quality, we use the improved Xception network in order to extract the high-dimensional features of the secret image, so that the entire secret image is hidden in the cover image, achieving high-capacity steganography.
2. Related Knowledge

In this chapter, we mainly introduce the development and characteristics of Inception V1, Inception V2, and Inception V3, the predecessors of Xception. Because Xception is the ultimate Inception, it is another improvement to Inception V3 that was proposed by Google, so this chapter will briefly describe the Inception improvement process in order to establish the foundation for the Xception architecture later.

2.1. Inception V1

In the past few years, with the rapid development of deep learning and convolutional networks, the object classification ability of the network has been greatly improved, and the requirements for the performance of the network have also been continuously improved. There are many ways for improving network performance, such as computer hardware upgrades, better data sets, and so on. How to further improve the performance of the network within the same resources has become the biggest problem. The most important method is to increase the number of layers of the network and the number of channels per layer, which is, the depth and width of the network. However, this method will allow us to add a large number of parameters while increasing the depth and width of the network, which results in a greatly increased amount of calculation and easy overfitting. In order to solve this problem, we determine how the Inception architecture can be proposed to approximate the best local sparse structure of the convolutional vision network, and cover its ideas with easy-to-use dense components [17].

The structure of Inception V1 mainly increases the network width, as shown in Figure 1. It provides different receptive fields for each branch, and it finally allows more scale information to stay in it, and the last stitching operation completes different scales. A fusion of features. However, there is a shortcoming of such a network, which is, increasing the width of the network, while increasing the number of parameters, because the number of parameters is too large to cause over-fitting problems. In order to solve this problem, $1 \times 1$ convolution is introduced to reduce the parameters, and $1 \times 1$ convolution will introduce more nonlinearity in order to improve generalization ability.

![Figure 1. Inception V1 architecture.](image)

2.2. Inception V2

For the Inception V1 network, the setting of initialization parameters and the learning rate is the difficulty of its training. Small changes in the first few layers of the network will be continuously amplified by the subsequent networks. If there are some changes in the input data distribution of a certain layer, then you need to learn this distribution, which results in the network that has been learning new changes to seriously affect the network training speed. Sergey Ioffe et al. [18] proposed normalizing the data. Therefore, Inception V2 introduces BN (Batch Normalization) operations on the basis of Inception V1. The advantages of introducing BN are: (a) a larger learning rate can be
set to reduce the network’s dependence on initialization parameters. (b) The problem of gradient dissipation is alleviated, and the original $5 \times 5$ convolution is replaced with two $3 \times 3$ convolutions in the Inception V2 framework in order to reduce the number of parameters. The BN operation is to normalize a certain layer, by formula (1):

$$
\hat{x}^{(k)} = \frac{x^{(k)} - E[ x^{(k)}]}{\sqrt{\text{Var}(x^{(k)})}}
$$

(1)

Where $E[ x^{(k)}]$ is the average value of neurons in this batch of data, and $\text{Var}(x^{(k)})$ is the standard deviation of the input value of each neuron. However, if you simply use the normalization operation, then it will have a greater impact on the features learned by the network. Hence, transform reconstruction is proposed, introducing two learning parameters $\gamma$ and $\beta$:

$$
y^{(k)} = \gamma^{(k)} \hat{x}^{(k)} + \beta^{(k)}
$$

(2)

In this way, each neuron has two corresponding parameters: $\gamma$ and $\beta$. When $\gamma^{(k)} = \sqrt{\text{Var}(x^{(k)})}$ and $\beta^{(k)} = E[ x^{(k)}]$, the features that are learned by the network can be reconstructed.

2.3. Inception V3

Before we can see that most of the architectures of Inception V1 and Inception V2 use convolution kernels of $3 \times 3$ or $5 \times 5$ convolution operations, because, the larger the convolution kernel, the larger the number of network parameters. Accordingly, how to use the convolution operation of a larger convolution kernel with limited resources? Inception V3 proposes decomposing a convolution operation with a kernel size of $n \times n$ into a set of convolution operations with a kernel size of $1 \times n$ and $n \times 1$, as shown in Figure 2. The experiments show that this idea can achieve the purpose of deepening the network, accelerating the calculation, and increasing the nonlinearity of the network.

![Figure 2. Inception V3’s improvements to Inception V2.](image)

2.4. Image Steganography

Baluja et al. [13] proposed the framework for hiding the image in the image. It consists of three parts of the network: preprocessing network, hiding network, and extraction network. Figure 3 shows the general framework. First, the first part of the preprocessing network adjusts images of different input sizes to a uniform size of $N \times N$ images (in this paper, the size is $256 \times 256$); then, the second part of the hidden network is to cascade the adjusted images of the preprocessing network. The output after operation is used as input. The features of the secret image are hidden in the various channels of the cover image and the stego image is generated as similar as possible to the cover image; the third part of the extraction network is to use the output of the hidden network as input from the stego image to extract the hidden secret image.
3. Proposed Solution

In this chapter, we mainly describe the specific framework of our proposed solution, the specific content of each part, namely the network architecture, and the composition and function of each small part of the network.

3.1. Network Architecture

Our proposed network first performs a cascading operation on the cover image (Cover) and the secret image (Secret), where the size and number of channels of the Cover and Secret are the same (the number of color image channels is 3, and the size is $256 \times 256$), and then through the hiding network (Hiding Network) to obtain the secret image (Stego-image), which is the image that we get after hiding, and then through the extraction network (RevSec Network) in order to reconstruct our hidden secret image (RevSec image). Figure 4 shows the concrete block diagram.

3.2. Hidden Network

The input of the hidden network is the image $R^{256 \times 256 \times 6}$ after the cascade of Cover and Secret, and the output is the hidden image $R^{256 \times 256 \times 3}$, which has two $4 \times 4$ convolutional layers and five Xception blocks (the Xception block is shown in Figure 5, as shown) and five residual blocks, of which the first $4 \times 4$ convolutional layer and five Xception blocks are used in order to reduce the feature dimension and allow for the secret image and cover image to perform feature fusion, and at the
same time in the Xception Block the use of fully separable convolution in, can greatly reduce the parameters. The five residual blocks are composed of five deconvolutions and five concatenation operations. The cascade is to superimpose the two sets of feature maps that need to be cascaded. The function of the residual block is to restore the cover image and retain the characteristics of the secret image in the cover image. The function of the hidden network is to embed the secret image in each channel of the cover image and obtain a secret image that is similar to the cover image. Our goal is to minimize the loss between Cover and Stego-image during embedding:

$$\tau = ||c - c'||$$

(3)

where \(c\) and \(c'\) represent the cover image and stego-image.

Where Figure 5 shows the architecture of the Xception block.

We will visualize the hiding and extraction operations in the proposed scheme in the form of pseudo code, as follows.

**Improved Xception Hidden and Extracted Procedures.**

**Input:** \(c, s\)

1. Initialize the convolution operation weight: \(\text{Normal}(0.0, 0.02)\);
   Initialize the BatchNorm operation weight: \(\text{Normal}(1.0, 0.02)\).

2. \(x = \text{Cat}(c, s)\)

3. For \(i = 1\) to \(200\) do

4. \(c' = \text{Encoder}(x)\).

5. Update the weight of the Encoder by \(||c - c'||\).

6. \(y = \text{Decoder}(c')\).

7. \(s' = \text{RevealNetwork}(y)\); (7)

8. Update the Encoder and Decoder weights through \(\zeta\).

9. End For

**Output:** \(c', s'\)

Table 1 shows the specific process of hidden network.

| Process               | Input (Channal) | Output (Channal) | Size (H x W) |
|-----------------------|-----------------|-----------------|--------------|
| Conv (k = 4, s = 2, p = 1) | 6               | 32              | 128 x 128    |
| Xception block        | 32              | 64              | 64 x 64      |
| Xception block        | 64              | 128             | 32 x 32      |
| Xception block        | 128             | 256             | 16 x 16      |
| Xception block        | 256             | 512             | 8 x 8        |
| Xception block        | 512             | 1024            | 4 x 4        |
| DeConv (k = 4, s = 2, p = 1) | 1024            | 512             | 8 x 8        |
| DeConv (k = 4, s = 2, p = 1) | 1024 + 512      | 512             | 16 x 16      |
| DeConv (k = 4, s = 2, p = 1) | 512 + 256       | 256             | 32 x 32      |
| DeConv (k = 4, s = 2, p = 1) | 256 + 128       | 128             | 64 x 64      |
| DeConv (k = 4, s = 2, p = 1) | 128 + 64        | 64              | 128 x 128    |
| DeConv (k = 4, s = 2, p = 1) | 64 + 32         | 32              | 256 x 256    |
| Conv (k = 3, s = 1, p = 1) | 32              | 3               | 256 x 256    |
3.3. ExtractS Network

The input of the extraction network is the output of the hidden network. At the same time, the channel number and size of the hidden network will not change. As shown in Figure 4, the extraction network has eight convolutional layers, seven BN operations, seven ReLUs, and one Sigmoid function. The first six groups are composed of one convolutional layer, one BN operation, and one ReLU in order to form the last group. It consists of a convolutional layer and a Sigmoid function. Their main function is to reconstruct the secret image from the steganographic image that was obtained by the hidden network. Of course, our reconstructed RevSec image and the original secret image will have a certain loss. Our goal is to minimize this loss:

\[
\rho = \|s - s'|\|
\]  

(4)

Table 2 shows the specific process.

| Process | Output Size (H × W × D) |
|---------|-------------------------|
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 32 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 64 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 128 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 256 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 128 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 64 |
| Conv (k = 3, s = 1, p = 1) + BN + ReLU | 256 × 256 × 32 |
| Conv (k = 3, s = 1, p = 1) + Sigmoid | 256 × 256 × 3 |

3.4. Loss Function

From the first two subsections, we can see that there are two losses in our network. The first is the loss \(\tau\) of the Stego-image process that was obtained by the cascaded image through the hidden network, and the second is the Stego-image reconstruction through the extraction network. The loss in the RevSec image process \(\rho\). The total loss of the network can be expressed by the following formula:

\[
\theta = \tau + \rho
\]  

(5)

However, while minimizing the total loss \(\theta\), we should also pay attention not to cause too much distortion of the Stego-image or RevSec image, so we add a weight in order to maintain it so that can better hide and extract information, as follows:

\[
\theta = \tau + \alpha \rho
\]  

(6)

where \(\alpha\) is the weight.

4. Experimental Results and Analysis

In this chapter, we introduce the specific experimental results and some data comparisons while using our proposed scheme. At the same time, we also tested the generalization ability and anti-stegal analysis ability of the scheme, and conducted ablation experiments in order to prove the rationality of our improved method. In this experiment, 20,000 images are used for model training, and 3000 images are used for model testing. These images are all from ImageNet. The initial learning rate of the model is 0.001, the hyperparameter \(\alpha = 0.75\), the batch of images entering the model is 8, and the number of training iterations is 200. In this experiment, the GPU that we used is NVIDIA GeForce 2070 (NVIDIA, Santa Clara, CA, USA), the experimental environment is Pytorch 1.1.0, the application is Python 3.6 for simulation experiments, and the number of GPUs is 2. Finally, we performed the visual subjective
evaluation and SSIM and PSNR objective evaluation on the final results. At the same time, we will also conduct an ablation experiment in order to show the importance of our improvement to the original Xception network.

4.1. Visual Subjective Assessment

First, the cover picture and secret picture (secret picture) were visually subjectively evaluated, and the reconstructed image, and also the error map between the cover image and the secret image. From Figure 6 (from left to right are cover image, secret image, stego image, revsec image, error diagram of cover image, and secret image (×1, ×5, ×10)), you can see the cover image and secret image. There is no visual difference between the secret image and reconstructed image. At the same time, we can only see the cover image artifacts when the error map is ×5 and ×10, respectively. No artifacts of the secret image are visible.

![Figure 6](image.png)

**Figure 6.** Randomly select three sets of images for subjective visual assessment.

4.2. PSNR, SSIM Objective Evaluation

PSNR is an important objective detection index of image quality, which is used in order to detect the distortion rate of the image. In this article, it is the logarithm of the mean square error between the cover image and the secret image, between the secret image and the reconstructed image, relative to \((2^n - 1)^2\) (the square of the maximum signal). For example, given two images \(A\) and \(B\) of size \((w, h)\), the mean square error (MSE) is:

\[
MSE = \frac{1}{WH} \sum_{a=1}^{W} \sum_{b=1}^{H} (A_{a,b} - B_{a,b})^2
\]

where \(n\) is the number of bits in each sample value. SSIM is an important indicator of image similarity. The structural similarity (SSIM) between the cover picture and secret picture (SSIM) will be given, which is, given the cover image and the secret image \(A\) and \(B\), the structural similarity can be calculated according to the following formula:

\[
SSIM = \frac{(2\mu_A \mu_B + c_1)(2\sigma_{AB} + c_2)}{(\mu_A^2 + \mu_B^2 + c_1)(\sigma_A^2 + \sigma_B^2 + c_2)}
\]
where \( \mu_A \) is the average value of \( A \), \( \mu_B \) is the average value of \( B \), \( \sigma^2_A \) is the variance of \( A \), \( \sigma^2_B \) is the variance of \( B \), \( \sigma_{AB} \) is the covariance of \( A \) and \( B \), \( c_1 = (k_1 L)^2 \), \( c_1 = (k_2 L)^2 \) is a constant used to maintain stability, and \( L \) is the range of pixel values. \( k_1 = 0.01 \), \( k_2 = 0.03 \). The value range is between 0–1. When the SSIM value is equal to 1, it means that the two images are exactly the same. At the same time, we randomly selected and drew the cover image and secret image. Figure 7 shows the histogram of the secret image and the reconstructed image, where, from left to right, are the cover image, the secret image, the stego image, the revsec image, and its histogram, and we can see that there is not much difference between them.

The two groups of randomly selected experimental results in Figure 7 and the average PSNR and SSIM of the scheme are shown in Table 3.

### Table 3. Peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) of stego-image and secret image, secret image, and RevSec image.

| Figure     | Cover and Stego-Image (PSNR,SSIM) | Secret vs. RevSec (PSNR,SSIM) |
|------------|-----------------------------------|-------------------------------|
| Figure 7 row1 | 40.632, 0.997 | 37.433, 0.979 |
| Figure 7 row3 | 40.327, 0.995 | 37.021, 0.971 |
| Average     | 40.211, 0.993 | 37.704, 0.983 |

At the same time, we compared SSIM and PSNR with other solutions, and the results are shown in the Table 4.

### Table 4. Comparison with other methods.

| Schemes  | Cover and Stego-Image (PSNR,SSIM) | Secret vs. RevSec (PSNR,SSIM) |
|----------|-----------------------------------|-------------------------------|
| [19]     | 34.892, 0.968 | 33.420, 0.947 |
| [20]     | 32.907, 0.960 | 36.601, 0.960 |
| [13]     | 39.894, 0.987 | 37.455, 0.977 |
| Average  | 40.211, 0.993 | 37.704, 0.983 |
4.3. Steganographic Capacity and Load Capacity

Steganographic capacity is an important indicator of network performance, which refers to the ratio of the size of the actual embedded secret information to the size of the cover information, which is Formula (10). In the case of the same image quality, the larger the steganographic capacity, the better.

\[
\text{Relative capacity} = \frac{\text{Absolute capacity}}{\text{The size of image}}
\]

where absolute capacity refers to the size of the storage space that is occupied by the secret information or the size of the secret image. The size of image refers to the size of the cover image. Because the size of the secret image in our proposed scheme is equal to the size of the cover image, the relative capacity is 1. The steganographic capacity comparison results of the proposed scheme and other schemes are shown in Table 5.

| Schemes | Absolute Capacity (bytes or pixel) | Stego-Image Size (pixel) | Relative Capacity (bytes/pixel) |
|---------|----------------------------------|--------------------------|----------------------------------|
| [21]    | ≥37.5                            | 64 × 64                  | 9.16 × 10^{-3}                  |
| [22]    | 0.375                            | 32 × 32                  | 3.7 × 10^{-4}                   |
| [23]    | 26,214~104,857                   | 512 × 512                | 1 × 10^{-1}~4 × 10^{-1}         |
| [24]    | 1535~4300                        | 1024 × 1024              | 1.46 × 10^{-3}~4.10 × 10^{-3}   |
| [25]    | 18.3~135.4                       | 64 × 64                  | 1.49 × 10^{-3}~1.10 × 10^{-2}   |
| Ours    | 256 × 256                        | 256 × 256                | 1                               |

The load capacity is the number of bits of information hidden by each pixel in the cover image. The load capacity can be obtained by the following formula:

\[
C = \omega \times 8 \times 3 \text{(bpp)}
\]

where \(C\) is the load capacity, each pixel occupies eight bits, a total of three channels, \(\omega\) is the probability that we reconstruct the network to correctly reconstruct the secret image, which is obtained by the following formula:

\[
\omega = 1 - \frac{\sum_{a=1}^{W} \sum_{b=1}^{H} |S_{a,b} - R_{a,b}|}{W \times H}
\]

We randomly selected four sets of experimental data to calculate the load capacity, as shown in Figure 8, the calculated load capacity is shown in Table 6 and, from left to right, are cover image, secret image, stego image, and revsec image. At the same time, 200 sets of experimental data are randomly selected to calculate its load capacity, and it can be concluded that the average load capacity of the proposed scheme is 23.96 bpp.

| Figure | load Capacity(bpp) |
|--------|---------------------|
| Figure 8 row1 | 23.96 |
| Figure 8 row2 | 23.96 |
| Figure 8 row3 | 23.95 |
| Figure 8 row4 | 23.97 |
4.4. Generalization Ability Test

Generalization ability refers to the adaptability of the program in new samples. The purpose of learning is also to learn the laws behind the data. Other data sets with the same laws can also get appropriate output through the network [26]. For image steganography, using samples other than the training set can also achieve the same steganography effect without reducing PSNR and SSIM. In order to ensure the generalization ability of our proposed model, we randomly selected some images from other data sets for testing, such as CelebA (Figure 9) and COCO (Figure 10); at the same time, we have also tested application scenarios, such as aerial images (Figure 11) and CT images (Figure 12). The results prove that our proposed model works well. The result images of each of the following tests are the cover image, the secret image, the secret image, the reconstructed image, the error map of the cover image, and the secret image (×1, ×5) from left to right.

4.5. Ablation Experiment

The ablation experiment is to prove the advantages of changing the model. Here, in order to prove the importance of the jump connection, we added whether we removed the jump connection and did this experiment again, and the result is shown in Figure 13, and, from left to right, are the cover image, secret image, stego image, and revsec image. This is the training image obtained after 158 iterations. It can be seen that the image convergence effect is still not satisfactory.
Figure 9. Test results of our proposed scheme on CelebA.

Figure 10. Test results of our proposed scheme on COCO.
**Figure 11.** The test results of our proposed scheme on aerial images.

**Figure 12.** The test results of our proposed scheme on CT images.
4.6. Steganalysis Test

Steganalysis is a means to detect whether there is hidden information in an image. Here, in order to test the performance of our experimental results against steganalysis, the steganalysis tool StegExpose [27] has been used for steganalysis. In the experiment, we used a standard threshold with a threshold of 0.2 and drew the ROC curve, as shown in Figure 14. The green line represents the random prediction, and the blue line represents the prediction of the steganalysis tool. You can see that the result of using the steganalysis tool is closer to the random prediction result.
5. Conclusions

In this article, we use deep learning instead of manual design in order to design the cost function. For the first time, we used the Xception network to hide the secret image with the same size as the cover image and improve it. Jump connections have been added to the Xception network to greatly speed up the convergence speed of the network. The experiments show that this scheme has a high steganography capability, with a load capacity of 23.96 bpp, which can realize high-capacity image steganography. Stego images and RevSec images also have high PSNR and SSIM, and the average PSNR and SSIM are higher than other existing image steganography schemes. At the same time, it has good generalization ability. When testing on other data sets, you can obtain the same effect as the training set. However, this solution also has some limitations. For example, when we use a high-texture image as a secret image and a low-texture image as a cover image, the result will be worse than the normal image. In the next step, we will introduce an attention module in order to improve the model, hide the secret image more in the high-texture area of the cover image, to obtain higher SSIM and PSNR, and introduce an encryption mechanism, when using steganalysis algorithms to know that we may have hidden information, it cannot get a meaningful real image, only an encrypted image. Finally, as attacks on stego images, such as cropping, compression, etc., will cause our secret images to be unable to be reconstructed, in the next step we will add cropping, compression, and other operations to the model in order to improve the model’s anti-attack ability through adversarial training.

Author Contributions: Conceptualization, X.D., M.G., N.L., W.W. and C.Q.; methodology, M.G.; software, validation, M.G.; formal analysis, all authors; writing—original draft preparation, M.G.; writing review and editing, M.G.; visualization, M.G.; project administration, X.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 61672354, and in part by the Key Scientific Research Project of Henan Provincial Higher Education under Grant 19B510005 and Grant 29B413004, and Science and Technology Project of Henan Provincial Department of Science and Technology under Grant 202102210165.

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

References

1. Tomás, F.; Jan, J.; Jessica, F. Minimizing Additive Distortion in Steganography Using Syndrome-Trellis Codes. IEEE Trans. Inf. Forensics Secur. 2011, 6, 920–935. [CrossRef]
2. Tomás, P.; Tomás, F.; Patrick, B. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In Proceedings of the Information Hiding—12th International Conference, Calgary, AB, Canada, 28–30 June 2010; pp. 161–177.
3. Vojtech, H.; Jessica, F. Designing steganographic distortion using directional filters. In Proceedings of the 2012 IEEE International Workshop on Information Forensics and Security, Costa Adeje, Tenerife, Spain, 2–5 December 2012; pp. 234–239.
4. Holub, V.; Jessica, F. Digital image steganography using universal distortion. In Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, New York, NY, USA, 17–19 June 2013; pp. 59–68.
5. Zhang, Y.; Luo, X.; Zhu, X.; Li, Z.; Adrian, G.B. Enhancing reliability and efficiency for real-time robust adaptive steganography using cyclic redundancy check codes. Real Time Image Process. 2020, 17, 115–123. [CrossRef]
6. Zhang, E.; Li, M.; Yu, S.; Du, J.; Jin, G. Fair Hierarchical Secret Sharing Scheme Based on Smart Contract. Inf. Sci. 2020. [CrossRef]
7. 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. [CrossRef]
8. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. IEEE Proc. 1998, 86, 2278–2324. [CrossRef]
9. Goodfellow, I. Generative Adversarial Networks. arXiv 2016, arXiv:1701.00160.
10. Yang, J.; Shi, Y.; Wong, E.; Kang, X. JPEG Steganalysis Based on DenseNet. arXiv 2017, arXiv:1711.09335.
11. Chen, F.; Xing, Q.; Liu, F. Technology of Hiding and Protecting the Secret Image Based on Two-Channel Deep Hiding Network. *IEEE Access* **2020**, *8*, 21966–21979. [CrossRef]
12. Wu, P.; Chang, X.; Yang, Y.; Li, X. BASN-Learning Steganography with a Binary Attention Mechanism. *Future Internet* **2020**, *12*, 43. [CrossRef]
13. Baluja, S. Hiding Images within Images. *Trans. Pattern Anal. Mach. Intell.* **2020**, *1685–1697*. [CrossRef]
14. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. *arXiv* **2016**, arXiv:1610.02357.
15. Alain, H.; Ziou, D. Image Quality Metrics: PSNR vs. SSIM. *IEEE Comput. Soc.* **2010**, 2366–2369. [CrossRef]
16. Ye, Y.; Shan, J.; Bruzzone, L.; Shen, L. Robust Registration of Multimodal Remote Sensing Images Based on Structural Similarity. *IEEE Trans. Geosci. Remote. Sens.* **2017**, *55*, 2941–2958. [CrossRef]
17. Szegedy, C.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, V. Going Deeper with Convolutions. *arXiv* **2014**, arXiv:1409.4842.
18. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
19. Zhang, R.; Dong, S.; Liu, J. Invisible steganography via generative adversarial networks. *Multim. Tools Appl.* **2019**, *8559–8575*. [CrossRef]
20. Rehman, A.U.; Rahim, R.; Nadeem, M.S.; Hussain, S. End-to-End Trained CNN Encoder-Decoder Networks for Image Steganography. In Proceedings of the Computer Vision, Workshops, Munich, Germany, 8–14 September 2018; pp. 723–729.
21. Hu, D.; Wang, L.; Jiang, W.; Zheng, S.; Li, B. A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks. *IEEE Access* **2018**, *6*, 38303–38314. [CrossRef]
22. Liu, M.; Zhang, M.; Liu, J.; Zhang, Y.; Ke, Y. Coverless Information Hiding Based on Generative adversarial networks. *arXiv* **2017**, arXiv:1712.06951.
23. Yang, J.; Liu, K.; Kang, X.; Wong, E.; Shi, Y. Spatial Image Steganography Based on Generative Adversarial Network. *arXiv* **2018**, arXiv:1804.07939.
24. Wu, K.; Wang, C. Steganography Using Reversible Texture Synthesis. *IEEE Trans. Image Process* **2015**, *24*, 130–139. [CrossRef]
25. Zhang, Z.; Liu, J.; Ke, Y.; Lei, Y.; Li, J.; Zhang, M.; Yang, X. Generative Steganography by Sampling. *IEEE Access* **2019**, *7*, 118586–118597. [CrossRef]
26. Zhou, Z. *Machine Learning*; Tsinghua University Press: Beijing, China, 2016.
27. Boehm, B. StegExpose—A Tool for Detecting LSB Steganography. *arXiv* **2014**, arXiv:1410.6656.

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0__).