electronics

MDPI

Article

An End-to-End Video Steganography Network Based on a Coding Unit Mask

Huanhuan Chai [1], Zhaohong Li [1,*], Fan Li [2] and Zhenzhen Zhang [3]

[1] School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China; 19120001@bjtu.edu.cn
[2] Department of Information and Intelligent Engineering, Tianjin University Renai College, Tianjin 301636, China; lifan000111@outlook.com
[3] School of Information Engineering, Beijing Institute of Graphic Communication, Beijing 102600, China; zhangzhenzhen@bigc.edu.cn
* Correspondence: zhhli2@bjtu.edu.cn

Abstract: Steganography hides secret messages inside the covers while ensuring imperceptibility. Different from traditional steganography, deep learning-based steganography has an adaptable and generalized framework without needing expertise regarding the embedding process. However, most steganography algorithms utilize images as covers instead of videos, which are more expressive and more widely spread. To this end, an end-to-end deep learning network for video steganography is proposed in this paper. A multiscale down-sampling feature extraction structure is designed, which consists of three parts including an encoder, a decoder, and a discriminator network. Furthermore, in order to facilitate the learning ability of network, a CU (coding unit) mask built from a VVC (versatile video coding) video is first introduced. In addition, an attention mechanism is used to further promote the visual quality. The experimental results show that the proposed steganography network can achieve a better performance in terms of the perceptual quality of stego videos, decoding the accuracy of hidden messages, and the relatively high embedding capacity compared with the state-of-the-art steganography networks.

Keywords: steganography; convolutional neural network; coding unit mask; attention mechanism; pyramid like generative adversarial network

Citation: Chai, H.; Li, Z.; Li, F.; Zhang, Z. An End-to-End Video Steganography Network Based on a Coding Unit Mask. *Electronics* **2022**, *11*, 1142. https://doi.org/10.3390/electronics11071142

Academic Editor: Shinichi Yamagiwa

Received: 3 March 2022
Accepted: 31 March 2022
Published: 5 April 2022

1. Introduction

Seeing is not always trustworthy. That is to say, a normal-looking video frame may contain secret information that is invisible to users. Information hiding enables people to conceal secret information within any kind of media. The internet has revolutionized global development in the sense that it has become the most convenient and economical telecommunication medium. Early internet communication was mainly in the form of text. However, over the last two decades, images and videos have gradually become the most popular form of multimedia communication.

Different from cryptography, steganography pursues imperceptibility, which requires hiding information without arousing people's perception [1,2]. Steganography is applied wherever secret communication is required [3]. Its application fields include military, medical, and multimedia fields, where steganography has been established as a promising and competitive method. For example, the famous "9/11" incident was caused by terrorists who escaped the monitoring of U.S. intelligence agencies by using digital image steganography technology, and who finally succeeded in triggering an attack. On an individual level, when works are illegally identified or copied, robust steganography can contribute to identifying users and protecting copyright [4,5].

Traditional steganography requires expertise in the hiding process [6], where a well-defined cost function is crucial because it is a comprise between embedding distortion and

payload capacity [7–12]. Traditional steganography has many years of research history, and many excellent methods have achieved good results in the visual quality and hidden capacity of steganographic images, in which the authors of [12] used the concept of pixel value differencing and modulus function (PVDMF) to achieve an excellent performance. However, due to its rapid development and powerful representational capabilities, deep learning is widely used in many industries, including autonomous driving, automatic translation, speech recognition, etc. Deep-learning-based steganography is adaptable and generalized, which saves a lot of artificial workloads. It does not need to manually design features and the convolutional neural network (CNN) can adaptively select features for information hiding. Correspondingly, it does not need to manually extract information and CNN can extract information conveniently and quickly. There are already many deep learning-based steganography algorithms. The steganography model proposed in [13] can resist various lossy image and video compressions, including non-differentiable JPEG compression. SSGAN [14] utilizes a new form of generative adversarial network (GAN) to resist steganalysis. SGAN [15] is a secure model for generating image-like containers based on deep convolutional generative adversarial networks (DCGAN).

Among deep-learning-based hiding methods [16–22], end-to-end steganography is a completely different and groundbreaking method, which is a three-player game [19–22] based on successful generative adversarial network (GAN) [23,24] that contains an encoder, a decoder, and an adversary. It is a new type of data hiding, which trains the embedding and extracting process simultaneously, and is designed to specifically work as a pair. In the family of three-player games, HiDDeN [21] and SteganoGAN [22] adopt the encoder structure and decoder structure to perform information embedding and information recovery processes, respectively. Furthermore, a third adversary network that plays a role in steganalysis is utilized. However, whether traditional steganography methods [25,26] or the deep-learning-based steganography, only a few of them take digital videos as carriers. Only two references refer to end-to-end video steganography. RivaGAN processes 3D video sequences using a 3D convolution kernel [27]. It features a custom attention-based structure for generating content-adaptive arbitrary data and has two adversaries to critique the video quality for the robustness of the model. The other is DVMark [28], which includes a multiscale structure that can distribute watermarks across multiple spatial-temporal scales, and achieves robustness through a differential distortion layer. However, none of them consider the characteristics of video compression such as coding unit division, transformation, intra prediction, inter prediction, and quantization.

In this paper, we introduce an end-to-end video steganography method based on the pyramid-like generation adversary neural network (PyraGAN). We hide messages of randomly generated 0 and 1 bits, and make sure that the container video frames are as close as possible to the cover video frames. To facilitate the visual quality of stego video, a novel solution is proposed. Specifically, a CU (coding units) mask, which is extracted from the CU partition mode of versatile video coding (VVC), is combined with a convolutional block attention mechanism. The experimental results show that the visual quality of the proposed method surpasses the state-of-the-art deep-learning based methods on the premise of ensuring an equivalent payload capacity. Proposed video steganography is different from image steganography, in that the cover looks like an image, but actually, it is an I-frame of a continuous video. Note that when a video of many consecutive frames is compressed by VVC, it is divided into groups of pictures (GOPs). Regardless of whether the GOP is based on IIII (intra-prediction encoding for each frame) or IPPP (intra-prediction encoding for the first frame, inter-prediction encoding for other frames), we hide messages in all decoded I-frames and then transmit the entire video.

The main contributions of this paper are as follows:

- We introduce a simple and light-weight end-to-end model architecture combined with GAN to achieve steganography for VVC videos.
- While ensuring a relatively large payload capacity, the visual quality is improved based on the CU mask and convolutional block attention mechanism.

2. Proposed End-to-End Video Steganography System

In this section, the details of each component of the proposed end-to-end video steganography architecture is proposed. In addition, the training process, including the loss function, dataset, and optimization, is also illustrated.

2.1. Architectrue

In this section, we present PyraGAN, a generative adversarial network combined with a CU mask to hide random n-bit binary messages in a host video frame. The proposed architecture is shown in Figure 1 and consists of three components: (1) An encoder, which takes cover video frames and secret messages as the input, and outputs steganographic (abbreviated as stego) video frames (Section 2.1.2), which combine the CU mask of the cover video frames and the attention mechanism to facilitate the visual quality of the stego video frames. (2) A decoder, which takes the stego video frame as the input and tries to recover the secret messages (Section 2.1.3). (3) A discriminator, which scores the quality of the cover video frames and stego video frames (Section 2.1.4).

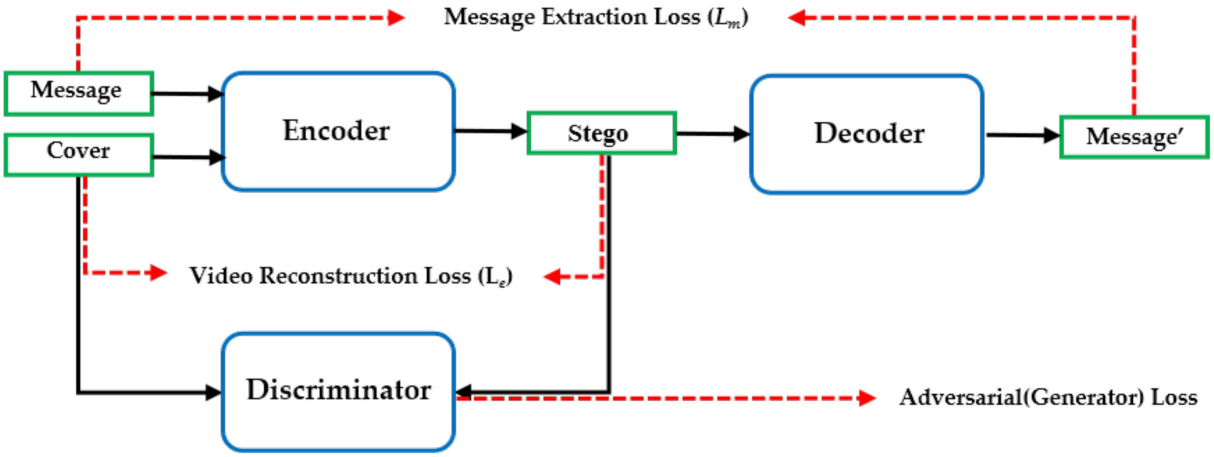


Figure 1. Architecture of the proposed PyraGAN.

2.1.1. Coding Unit Mask and Convolutional Block Attention Mechanism

Most video coding standards are based on CU partition mode to carry out transforming, quantization, entropy coding, etc. Figure 2 shows the CU division of one video frame in which the content in the red box is magnified in the lower left corner. We can see that the CU division fits the content of the video frame very well, and the CU blocks in flat areas are large, while those in areas with complex details and textures are small. The CU division of the VVC compressed video frame represents the diversity and complexity of its content. Therefore, we extracted the CU partition during VVC encoding and made it into a binary mask, named the CU mask, which assisted CNN to better extract features to hide messages in areas of the video that are not easy to detect, such as areas with intricate textures. Finally, it improved the imperceptibility of stego video frames.

Meanwhile, as shown in Figure 3, the convolutional block attention module (CBAM) [29] was used as our attention mechanism. It successively includes two parts—channel attention (CA) module and spatial attention (SA) module—which assist in focusing on "what" and "where" is meaningful regarding the input features, respectively. It is a plug-and-play module that requires few parameters to boost the representation of CNNs.

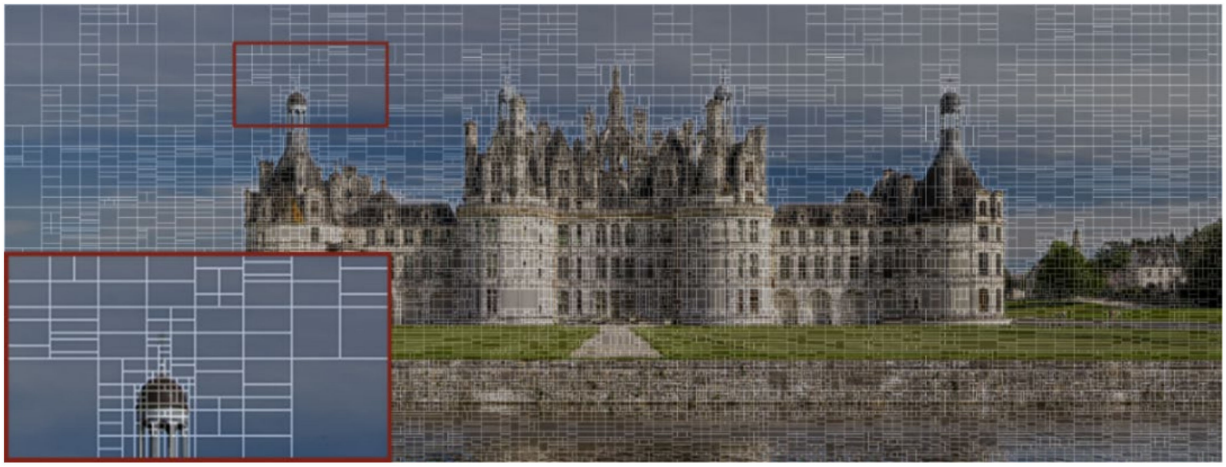


Figure 2. The CU mask of one video frame.

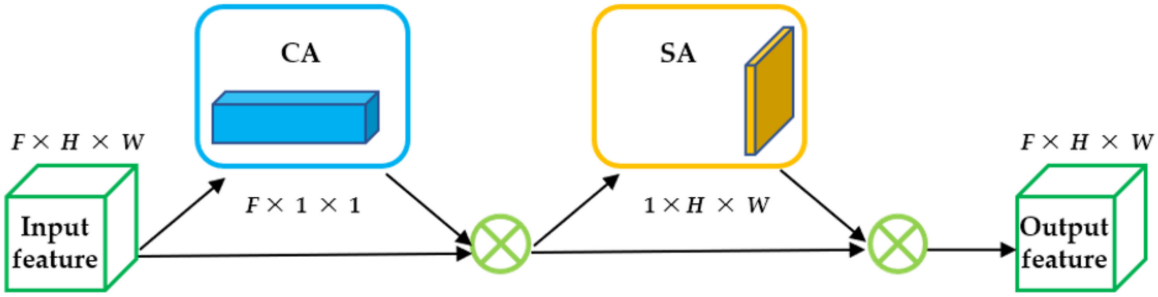


Figure 3. Convolutional block attention mechanism.

Each channel of the features can be regarded as a feature detector. The CA assigns a weight to the input F-dimensional features by learning, and obtains a tensor of size $F \times 1 \times 1$, in which the larger the weight coefficient is, the more important the channel it represents for feature learning of the model. Then, the result is copied and expanded into the tensor of $F \times H \times W$ (H and W represent the height and width of the features, respectively) along the spatial dimension. Afterwards, this along with the input features are multiplied pixel by pixel. So far, CA makes the network learn to focus on the channel features that contribute greatly to the result. Sequentially, SA assigns weights to the intermediate F-dimensional features, and outputs a tensor of size $1 \times H \times W$ in which the larger the weight coefficient is, the more important the feature is at this location for model learning. Then, the result is copied and expanded into the tensor of $F \times H \times W$ along the feature channel dimension and is multiplied by the features obtained from the previous layer. So far, SA makes the network learn to focus on the positions of features that contribute greatly to the results. Finally, the output feature size of the attention mechanism is the same as the input feature size, which is $F \times H \times W$.

2.1.2. Encoder Network Design

The encoder, as shown in Figure 4, accepts cover video frames and random n-bit binary messages as the inputs, and the stego frames as the outputs.

(1) Conv-module is used sequentially, including the convolutional layer, Leaky ReLU [30] activation function, and Batch Normalization (BN) [31,32] to extract features from the cover frames, which are sized $C \times H \times W$ (C, H, and W represent the channel, height, and width of video frame, respectively), and to get F-channel features that can be utilized by later conv-modules. The kernel size of convolution layer is 3×3, the

step is 1, and the padding is 1, in order to keep the size of the output the same as that of the input.

(2) The message is reshaped to $D \times H \times W$, concatenated in the depth dimension with F-channel features of cover frames and the CU mask, which is achieved by compressing the cover frames using VVC to form $(F + D + 1)$ dimensional feature maps, where D, H, and W represent the depth, height, and width of the message, respectively.

(3) Inspired by the U-net [33] network structure and the different-size CU partition modes of video frames in video compression, the features are fed into three feature extraction network channels. Besides the first channel that implements feature extraction without changing the height and width of features, the other two channels conduct double and quadruple down sampling on the input feature maps, respectively, that is, the height and width are changed to one-half and one-quarter of its original size through down sampling. At the same time, the channel of extracted feature maps by the convolution module becomes $2F$ and $4F$, respectively. Down sampling integrates the spatial information of adjacent areas, captures the context details for feature extraction, and assists the hidden message to find appropriate features in the cover video frames. The larger the down sampling scale, the larger the spatial domain information it can integrate. At the same time, the lost fine details can be compensated by obtaining more channels of features through the subsequent convolution module.

(4) CBAM is introduced into three network channels. The CA module enables the network to learn to focus on "which" feature maps, according to the block division of video frame content in the CU mask. Successively, the SA module enables the network to learn to focus on "where" features, according to the block division at different positions of the CU mask.

(5) The bicubic interpolation is used to reshape the feature size of three networks into $H \times W$, and meanwhile, the accurate location of the secret information in the cover video frames is realized.

When the feature maps obtained from the down sampling of three channels are stacked together, from right to left in Figure 4, the size of the feature maps changes from small to large, similar to a pyramid. Therefore, the model is called PyraGAN (pyramid-like generative adversarial network). Based on the consideration of model parameters, the above pyramid structure, which includes the model of three feature extraction channels, is called PyraGAN_F16_3, where the feature maps variable F is set as 16. In order to reduce the complexity of the model, one feature extraction channel is removed—the blue box is shown in Figure 4. This model is named PyraGAN_F32_2, where the feature dimension F is selected as 32.

(6) The three-way feature maps and the inputs of the pyramid structure (including the feature maps of cover video frame, hidden messages, and CU mask) are concatenated to get $(8F + D + 1)$ channel features.

(7) A convolution module containing only convolution layer maps them into features of $C \times H \times W$ in seize.

(8) In order to avoid vanishing and exploding of gradients, the idea of the ResNet [34] network is adopted to make identity mapping for the input cover video frame, that is, the cover frame is added to the features of a size of $C \times H \times W$ pixel by pixel. In fact, the encoder processes the residual between the stego video frames and the cover video frames, that is, the secret message is hidden in the high-frequency part of the cover video frames. Because human eyes are not sensitive to the high-frequency part of the video frames, hiding messages in the high-frequency part helps to facilitate the visual quality of the stego video frames.

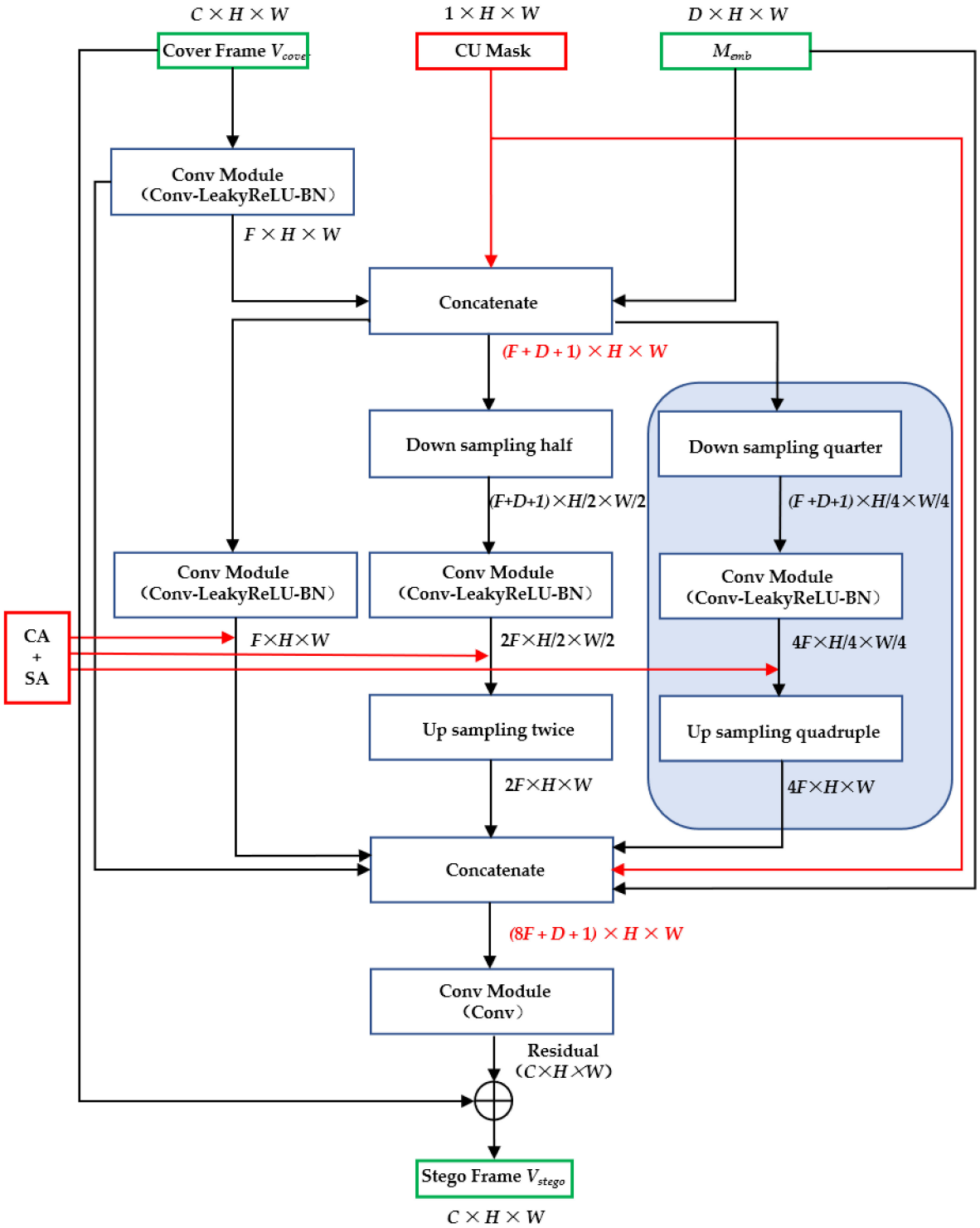


Figure 4. The encoder component of PyraGAN.

2.1.3. Decoder Network Design

The decoder (shown in Figure 5) aims at extracting messages and accepts the stego video frame, which is the output, of the encoder as the input. The function realized by the decoder and the encoder is mutually inverse, so the network structure is basically similar, made up of one conv-module, one multi-scale feature extraction module, and one final convolutional layer.

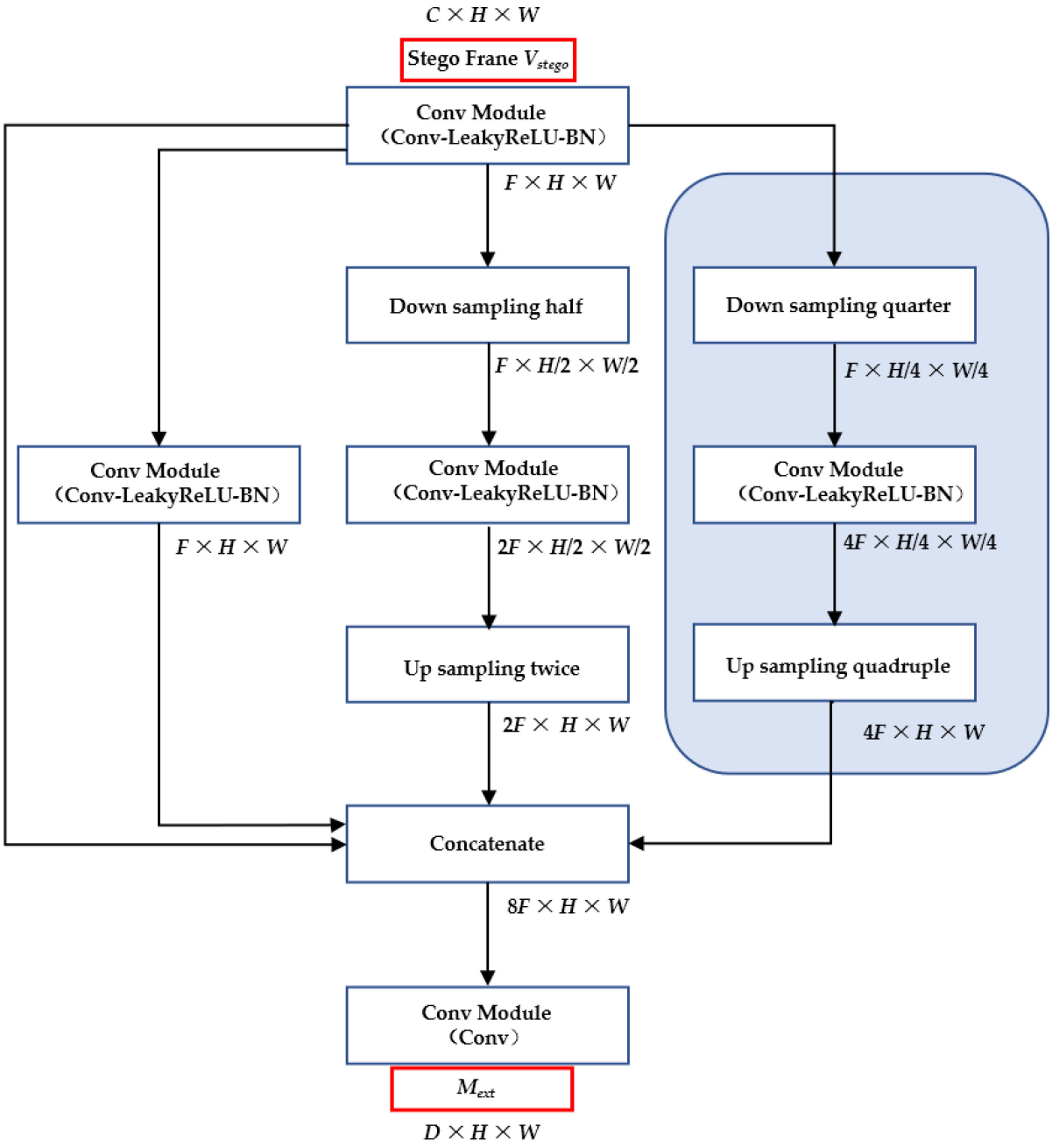


Figure 5. The decoder component of PyraGAN.

(1) F-dimensional feature maps of stego video frames can be obtained through a conv-module.
(2) Feature extraction is carried out in three ways, where the feature size is down sampled to the original, the half, and the quarter, respectively. Through conv-module, F-dimension, $2F$-dimension, and $4F$-dimension features can be achieved, respectively, which help to integrate cover video frames from different neighborhood spaces and realize comprehensive feature learning.
(3) The features of the three methods are all reshaped to $H \times W$ by up sampling.
(4) They, together with the first F-dimension feature maps, are concatenated to form a tensor of size $8F \times H \times W$.
(5) They are sent to the last convolution layer, which implements accurate prediction and outputs the extracted secret information of message. Note that Leaky ReLU and BN are used in all but the final conv-module.

When training the network, the encoder and decoder are jointly trained, that is, the network forward operation, gradient descent, parameter update, and other operations are implemented synchronously. This process of embedding secret messages into cover video frames to obtain stego video frames and extracting messages from stego video frames through one network operation is the core of end-to-end steganography. When the encoder is PyraGAN_F16_3, the decoder also sets the intermediate features F to 16 and adopts three channels. When the encoder is PyraGAN_F32_2, the decoder also sets the intermediate features F to 32 and adopts two channels, that is, the channel in the blue area of Figure 5 is disabled.

2.1.4. Discriminator Network Design

The discriminator, as shown in Figure 6, plays the role of steganalysis, which accepts both cover frames and steganographic frames as the inputs. It is made up of three conv-modules and one final convolutional layer to get a score for cover frames and stego frames, respectively. When the encoder is fixed and the discriminator is trained, the discriminator can distinguish between the cover frames and stego frames. On the contrary, the discriminator cannot distinguish them. After the alternating iterative training, the performance of the encoder continues to improve, and stego frames can confuse the discriminator.

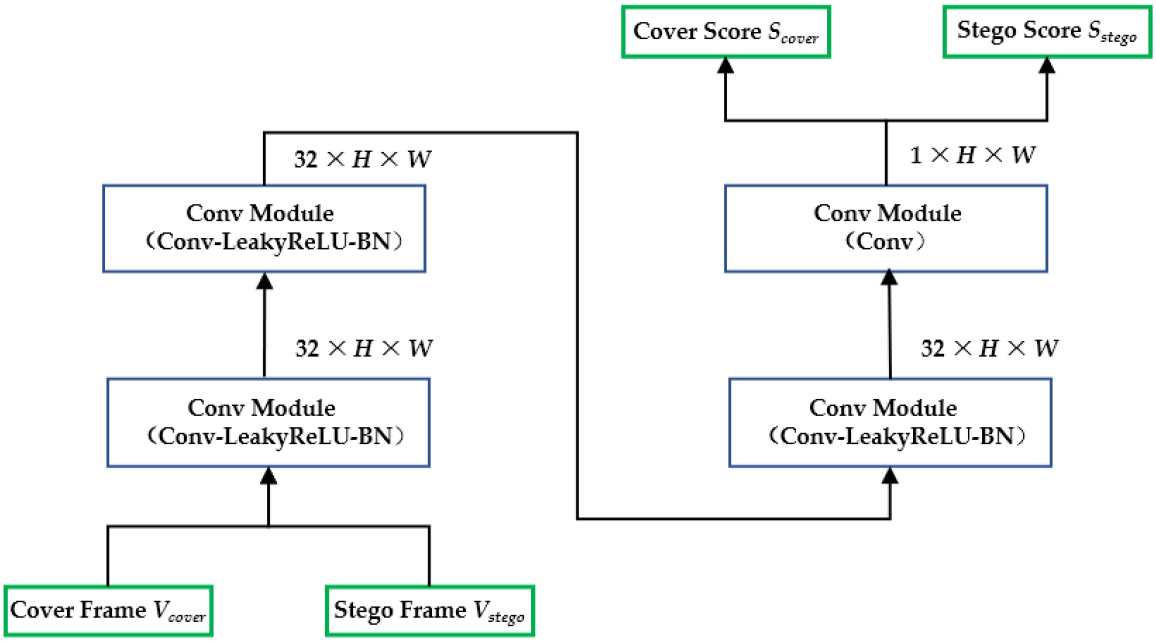


Figure 6. The discriminator component of PyraGAN.

2.2. Training

In this section, the training details of the model, including the loss function, dataset, configuration, and optimization algorithm, are introduced.

2.2.1. Loss Function

The encoder, decoder and discriminator are integrated together for end-to-end training. As shown in Figure 1, the loss functions are set reasonably and are minimized by gradient back propagation and update of the model parameters. After multiple rounds of training on the dataset, the encoder and decoder models with updated parameters are obtained, which are applied to the sender and receiver of secret messages, respectively.

The red dotted line in Figure 1 indicates three loss functions for network training. (1) Message extraction loss (L_m) in Formula (1) uses cross-entropy loss. The lower the bit

error rate between the extracted message $\boldsymbol{M_{ext}}$ and embedded message $\boldsymbol{M_{emb}}$, the smaller this loss.

$$L_m = CrossEntropy(\boldsymbol{M_{emb}}, \boldsymbol{M_{ext}}) \tag{1}$$

(2) Video reconstruction loss (L_e) in Formula (2) uses the mean square error (MSE), where C, H, and W represent the channel, height, and width of one video frame, respectively, and $\boldsymbol{V_{(k,i,j)}}$ represents the pixel value at channel k, row i, and column j of the video frame. The more similar the stego video frames and cover video frames, the smaller this loss.

$$L_e = MSE(\boldsymbol{V_{cover}}, \boldsymbol{V_{stego}}) = \frac{1}{C \times H \times W} \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(V_{cover\,(k,i,j)} - V_{stego\,(k,i,j)} \right)^2 \tag{2}$$

(3) Adversarial (generator) loss in Formula (3) uses Wasserstein [35] distance to characterize the degree of overlap between the distribution of stego frame $\boldsymbol{V_{stego}}$ and that of the cover frame $\boldsymbol{V_{cover}}$ where σ represents the discriminator and E represents the expectation of the distribution function. P_{stego} is the data distribution of the stego frame. P_{cover} is the data distribution of the cover frame.

$$\max_{\sigma} \min_{\varepsilon} L(\boldsymbol{V_{cover}}, \boldsymbol{V_{stego}}) = E_{x \sim P_{stego}} \sigma(x) - E_{x \sim P_{cover}} \sigma(x) \tag{3}$$

The generator needs to minimize the Wasserstein [35] distance so that the stego video frames can be confused with the cover frames. When training the generator, the discriminator parameters are fixed, and the latter of Formula (3) has nothing to do with the generator, so the generator loss (L_a) adopts what is shown in Formula (4), where ε represents the encoder. When training the discriminator, the generator parameters are fixed. The discriminator should be enough to distinguish the $\boldsymbol{V_{stego}}$ and the $\boldsymbol{V_{cover}}$, so it should maximize the Wasserstein [35] distance, i.e., minimize its opposite. Therefore, the adversarial loss (L_d) is shown in Formula (5).

$$L_a = E_{x \sim P_{stego}} \sigma(x) = E_{x \sim P_{stego}} \sigma(\varepsilon(\boldsymbol{V_{cover}}, \boldsymbol{M_{emb}})) \tag{4}$$

$$L_d = E_{x \sim P_{cover}} \sigma(x) - E_{x \sim P_{stego}} \sigma(x) \tag{5}$$

During each round of training, the discriminator and encoder–decoder network are trained alternately. First, the discriminator is trained to optimize the parameters by minimizing the loss L_d in Formula (5). Then, the encoder–decoder network is jointly trained to update its parameters by minimizing the total loss L_g, as shown in Formula (6). λ_1, λ_2, and λ_3 are the weight coefficients of message extraction loss L_m, video reconstruction loss L_e, and generator loss L_a, respectively.

$$L_g = \lambda_1 \times L_m + \lambda_2 \times L_e + \lambda_3 \times L_a \tag{6}$$

2.2.2. Dataset

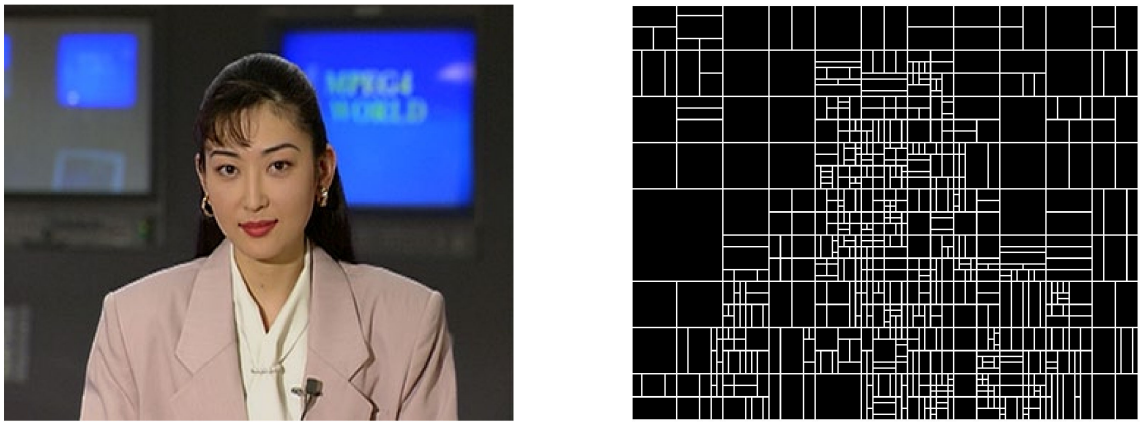
The biggest difference between the end-to-end video steganography algorithm in this paper and the image steganography algorithm is that the video steganography works on the VVC decoding video frames. The unique color space conversion between RGB and YUV, CU block division, intra prediction, inter prediction, motion estimation and motion compensation, discrete cosine transform (DCT) from spatial domain to frequency domain, quantization, and other processes of VVC bring about a series of irreversible losses to video frames. In practical applications, it is inevitable to compress the original video due to occupying a lot of memory, and the decoding video frames may provide more favorable hidden features for secret messages. Therefore, the video frames decoding by VVC compression are selected as the covers in this paper.

DIV2K [36] and MSCOCO [37] are selected as the datasets. DIV2K contains 900 natural scene images with a high resolution, including 800 pictures of the training set and 100 pictures of the verification set. The MSCOCO dataset has 118,060 pictures of the training set,

including many common pictures in life with a complex background and small size. In this experiment, 5000 images of the MSCOCO dataset are randomly selected, which are divided into five equal parts. Plus, for the training set of DIV2K, a total of six training datasets are used as the final network training dataset. The verification set of DIV2K is selected for verification.

The steps to make the dataset are as follows:

1. Because the video compressed by VVC must be YUV color space, six datasets are spliced into six videos and transformed from the RGB color space to the YUV color space.
2. VVC compresses them using the intra prediction mode, with the quantization step (*QP*) selected as 37. In this process, Make the CU mask of the corresponding cover video frame according to the division of VVC. Figure 7 shows a video frame decoding by VVC compression and the corresponding CU mask. Set the pixel value of the corresponding CU division edge area to 255 and the interior of the CU block to 0, and obtain the CU mask picture, as shown in Figure 7b, which is a single channel gray image. The CU mask in Figure 7b roughly represents the outline of the whole video frame content in Figure 7a. The content of the background area is flat and its CU block is relatively large. The human body has many textures and its CU block is relatively small and diverse.
3. Convert the decoded videos from the YUV domain to the RGB domain.
4. These, together with their CU masks, are as the whole network training dataset. This is true for the whole network verification dataset.



(**a**) VVC compressed reconstructed video frame (**b**) CU mask based on CU partition

Figure 7. VVC compressed reconstructed video frame and corresponding CU mask.

Models are trained on six training datasets. Then, they are evaluated separately on the test set and the average is obtained.

2.2.3. Configuration and Optimization

In terms of software and hardware configuration, all operations are implemented on a GeForce GTX 2080 Ti with driver version of the graphics card 450.57. The CUDA version is 11.0 and the CUDNN version is 7.6.4. The neural network framework selects Pytorch, the integration environment selects vs. Code, and the programming environment selects Python 4.5.3.

In terms of the hyperparameters of the model, the number of training rounds (epoch) of the dataset is 50. Adam [38] is selected as the optimization algorithm with a learning rate of 10^{-4}, batch size of 4, and default hyperparameters. In order to limit the memory utilization of the model, the input is cut to 256×256. The weight coefficients of the message

extraction loss λ_1 and that of the adversarial (generator) loss λ_3 are both 1, while the weight coefficient of the video reconstruction loss λ_2 is 100.

3. Experimental Results

Experiments are carried out on the proposed model PyraGAN_F32_2 and PyraGAN_F16_3. Then, we compare the performance of them with HiDDeN [21] and SteganoGAN [22], which achieved groundbreaking end-to-end steganography. HiDDeN [23] was proposed by the team of Stanford, and it realizes secure and robust data hiding despite the presence of noise such as JPEG compression, Gaussian blurring, cropping, and dropout. However, its weakness is that capacity of hiding data is extremely small. SteganoGAN [22] from the team of MIT gained a relatively large capacity for hiding data. The above two networks both take the image as a carrier. Few references take the video as the carrier [27,28], especially taking into consideration the features of video compression such as the CU partition mode. Inspired by the idea of the end-to-end steganography network, we aimed to construct a lightweight, secure, and large capacity steganography network for the latest VVC videos. We reproduce HiDDeN [21] and SteganoGAN [22] using open-source code for comparison experiments. For the sake of fairness, their performances are evaluated on the same dataset.

3.1. Number of Kernel Parameters Comparison among Different Steganography Networks

Model parameter is an index to measure the complexity of the model. The larger number of parameters the model has, the larger the graphics card memory it requires, and the more time one-time forward operation spends. Constructing an effective model with a small number of parameters plays an important role in the portability, run speed, and memory requirements of the algorithm.

The parameters and occupied memory of different networks are displayed in Table 1. Note that PyraGAN_F32_2 is the model for which the hidden feature channels are 32 and the down sampling network channels are 2, as shown in Figures 4 and 5, where the channel of the blue box is removed, PyraGAN_F16_3 is the model for which the hidden feature channels are 16, and the down sampling network channels are 3. Model HiDDeN [21] has the maximum number of parameters on the treen components, and the total memory it occupies is up to 1.8 M. Proposed PyraGAN_F32_2 has the same number parameters as model SteganoGAN [22] on the discriminator component and almost equivalent parameters for the decoder and encoder components, and the total memory occupied of model is one fifth that of HiDDeN. By increasing the diversity of the down sampling and reducing the features of the hidden features, the proposed PyraGAN_F16_3 is a light-weight model that occupies only 220 K memory, as shown in bold in Table 1, and has minimum parameters among four networks.

Table 1. Parameter comparison of different steganography CNNs.

Parameters	HiDDeN [21]	SteganoGAN [22]	PyraGAN_F32_2	PyraGAN_F16_3
Discriminator	76,097/30 K	19,873/88 K	19,873/88 K	5329/28 K
Encoder	169,347/676 K	31,998/136 K	34,973/148 K	23,439/108 K
Decoder	205,500/824 K	29,665/124 K	30,049/128 K	18,097/84 K
Total memory occupied	1808 K	348 K	364 K	220 K

3.2. Comparison of Performance in Capacity, Imperceptibility and Extraction Accuracy

A good steganography algorithm should take properties of capacity, security, imperceptibility, and accuracy into consideration. In this paper, capacity based on BPP (bit per pixel), imperceptibility based on PSNR (peak signal-to-noise ratio), and extraction accuracy of messages abbreviated to ACCU are demonstrated, respectively.

3.2.1. Capacity of Information Hiding

The capacity of information hiding is the basic index to measure the steganography algorithm. If the embedding capacity is too small, more covers are needed to realize the transmission of the same amount of information, which will greatly limit the practicability of the steganography algorithms.

BPP, defined in Equation (7), is the bits of hidden messages per pixel, where L is the length of the hidden information. H and W represent the height and width of the cover video frames, respectively. Since our stego frames and the cover frames have the same size, which is 256 × 256, the capacity of the proposed method is 1 BPP. In the second row of Table 2, the BPP of proposed PyraGAN_F32_2, PyraGAN_F16_3, and SteganoGAN [22] is the same, which is two hundred times that of HiDDeN [21] owing to the hidden messages reshaped to $H \times W$, which is the size of the cover frames. Such an information embedding capacity is at a high level in most steganography algorithms.

$$BPP = L/(H \times W) \tag{7}$$

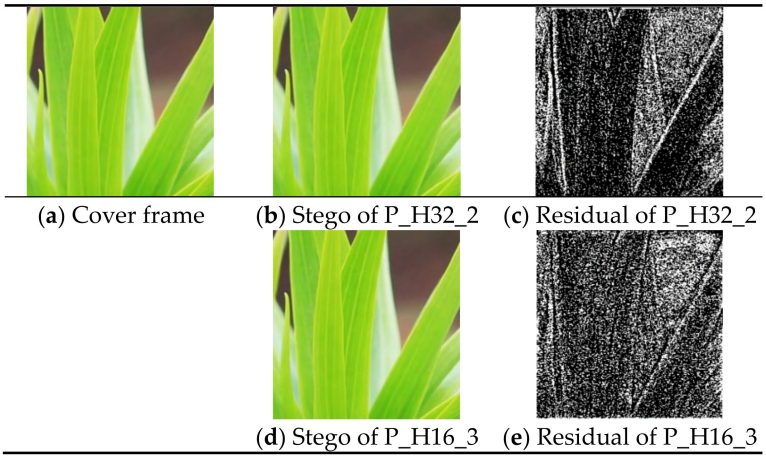
Table 2. Performance comparison of different steganography CNNs.

Models	HiDDeN [21]	SteganoGAN [22]	PyraGAN_F32_2	PyraGAN_F16_3
BPP(bit)	0.0018	1	1	1
PSNR(dB)	40.081	41.428	**43.109**	42.995
ACCU(%)	0.793	0.993	**0.994**	0.993

3.2.2. Imperceptibility of Stego Video Frames

The imperceptibility of the stego video frames is an important index to measure the steganography algorithm. The better the steganography algorithm, the higher the visual quality of stego video frames, which ensures that the secret information is invisible and not easy to be suspected. It can be evaluated from both qualitative and quantitative perspectives.

Qualitatively speaking, Figure 8 shows the visual quality of the steganographic video frames of P_F32_2 and P_F16_3. They both have high visual quality and do not appear to carry a hidden message. Then, on the right side of them, we can see their gray residuals obtained by subtracting the cover video frame pixel by pixel. Residual, in a sense, represents the hidden message. The residuals of P_F32_2 and P_F16_3 both reflect the outline of the video frame, where P_F32_2′s is much clearer, proving that the stego frames of PyraGAN achieve a good video content adaptation, and P_H16_3 achieves a large reduction in the number of parameters at a small cost in the visual quality of the residual.



(**a**) Cover frame (**b**) Stego of P_H32_2 (**c**) Residual of P_H32_2

(**d**) Stego of P_H16_3 (**e**) Residual of P_H16_3

Figure 8. The visual quality of steganographic video frame of PyraGAN.

Quantitatively speaking, PSNR, shown in Equation (8), is used to measure the similarity between two video frames pixel by pixel. First, MSE is calculated according to Equation (2) and the PSNR value is calculated from MSE, where A represents the peak signal.

$$PSNR = 20 \times log_{10}\left(\frac{A}{\sqrt{MSE}}\right) \tag{8}$$

The PSNR of different model is shown in the third row of Table 2. The proposed P_F32_2 achieves the best performance, for which the PSNR value in bold in Table 2 is 2 dB higher than that of SteganoGAN and 3 dB higher than that of HiDDeN, because the CU mask is a good assistant to help the model learn features of the cover video frames. In addition, the attention mechanism focuses on "which" features and "where" features are important for information hiding. Therefore, it realizes the adaptive information hiding based on the content of the cover video frames. Furthermore, at the expense that the PSNR value of P_F16_3 is 42.995 dB, underlined in Table 2, and only 0.1 dB lower than that of P_F32_2, the model P_F16_3 realizes the reduction of computational complexity greatly. Although P_F16_3 has the minimum parameters, its stego video frames have almost the best quality because its increase of multiscale feature extraction layers additionally boosts the performance of invisibility.

3.2.3. Extraction Accuracy of Hiding Secret Message

Ensuring the message of the receiver is the same as the sender is an important indicator to test the effectiveness of the steganography algorithm. ACCU is the accuracy of the bit-to-bit comparison of received and sent messages. From the fourth row in Table 2, the proposed P_F32_2 has the ACCU value of 0.994 in bold, which is the highest in the four models, and the proposed P_F16_3 and model SteganoGAN both have the ACCU value of 0.993, which is the second highest. Note that the decoder of P_F32_2 has more hidden features that can decode hidden messages more accurately. Compared with P_F32_2, the proposed P_F16_3 takes two third parameters, which achieves basically equivalent message extraction accuracy owing to its third feature extraction network channel, as shown in blue box of Figure 5. Therefore, PyraGAN has a good performance in extracting messages.

In general, the proposed model PyraGAN is the best in terms of capacity, imperceptibility, and accuracy compared with the existing two excellent models of HiDDeN [21] and SteganoGAN [22]. In addition, the proposed model P_F16_3 uses the minimum parameters, of which the occupied memory is only 220 K.

4. Conclusions

In this paper, we introduce an end- to-end video steganography architecture named as PyraGAN, which supports different-sized cover video frames and arbitrary binary data. Furthermore, we improve the performance of the model based on the CU mask made from VVC and attention mechanism. Experimental results of PyraGAN, HiDDeN [23], and SteganoGAN [23] demonstrate that proposed PyraGAN achieves a better performance in invisibility, capacity of hidden message, and accuracy of decoded message. This paper provides probability for end-to-end video steganography combined with coding unit of VVC. In future work, the robustness of the proposed steganography network will be the first important issue, such as adding a differentiable noise layer after encoder to resist some popular attacks. Moreover, the research will be conducted combined with other characteristics of video compression such as inter prediction. Furthermore, using a suitable error correction encoding method for perfect extraction of hidden messages which will be another research direction of our future work.

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

Funding: This work was founded by The Scientific Research Common Program of the Beijing Municipal Commission of Education (KM202110015004).

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

References

1. Hussain, M.; Wahab, A.W.A.; Idris, Y.I.B.; Ho, A.T.S.; Jung, K. Image steganography in spatial domain: A survey. *Signal Processing Image Commun.* **2018**, *65*, 46–66. [CrossRef]
2. Chanu, Y.J.; Singh, K.M.; Tuithung, T. Image steganography and steganalysis: A survey. *Int. J. Comput. Appl.* **2012**, *52*, 1–11.
3. Kadhim, I.J.; Premaratne, P.; Vial, P.J.; Halloran, B. Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research. *Neurocomputing* **2019**, *335*, 299–326. [CrossRef]
4. Alenizi, F.A. Robust Data Hiding in Multimedia for Authentication and Ownership Protection. PhD Thesis, University of California, Irvine, CA, USA, 2017.
5. Cheddad, A.; Condell, J.; Curran, K.; Kevitt, P.M. Digital image steganography: Survey and analysis of current methods. *Signal Processing* **2010**, *90*, 727–752. [CrossRef]
6. Pevný, T.; Filler, T.; Bas, P. *Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. Proceedings of the International Workshop on Information Hiding, Calgary, AB, Canada, 28–30 June 2010*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 161–177.
7. Filler, T.; Judas, J.; Fridrich, J. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Trans. Inf. Forensics Secur.* **2011**, *6*, 920–935. [CrossRef]
8. Holub, V.; Fridrich, J. Designing steganographic distortion using directional filters. In Proceedings of the 2012 IEEE International workshop on information forensics and security (WIFS), Costa Adeje, Spain, 2–5 December 2012; pp. 234–239.
9. Li, B.; Wang, M.; Huang, J.; Li, X. A new cost function for spatial image steganography. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 4206–4210.
10. Holub, V.; Fridrich, J.; Denemark, T. Universal distortion function for steganography in an arbitrary domain. *EURASIP J. Inf. Secur.* **2014**, *2014*, 1–13. [CrossRef]
11. Sedighi, V.; Cogranne, R.; Fridrich, J. Content-adaptive steganography by minimizing statistical detectability. *IEEE Trans. Inf. Forensics Secur.* **2015**, *11*, 221–234. [CrossRef]
12. Sahu, A.K.; Swain, G. An optimal information hiding approach based on pixel value differencing and modulus function. *Wirel. Pers. Commun.* **2019**, *108*, 159–174. [CrossRef]
13. Zhang, C.; Karjauv, A.; Benz, P.; Kweon, I.S. Towards Robust Data Hiding Against (JPEG) Compression: A Pseudo-Differentiable Deep Learning Approach. *arXiv* **2020**, arXiv:2101.00973.
14. Shi, H.; Dong, J.; Wang, W.; Qian, Y.; Zhang, X. SSGAN: Secure Steganography Based on Generative Adversarial Networks. In Proceedings of the Pacific Rim Conference on Multimedia, Harbin, China, 28–20 September 2017; Springer: Cham, Switzerland, 2017; pp. 534–544.
15. Volkhonskiy, D.; Nazarov, I.; Burnaev, E. Steganographic Generative Adversarial Networks. In Proceedings of the Twelfth International Conference on Machine Vision (ICMV 2019), Amsterdam, The Netherlands, 16–18 November 2019; International Society for Optics and Photonics: Bellingham, WA, USA, 2020; Volume 11433, p. 11433M.
16. Tang, W.; Li, B.; Tan, S.; Barni, M.; Huang, J. CNN-based adversarial embedding for image steganography. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2074–2087. [CrossRef]
17. Tang, W.; Tan, S.; Li, B.; Huang, J. Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Lett.* **2017**, *24*, 1547–1551. [CrossRef]
18. Yang, J.; Ruan, D.; Huang, J.; Kang, X.; Shi, Y. An embedding cost learning framework using GAN. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 839–851. [CrossRef]
19. Baluja, S. Hiding images within images. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *42*, 1685–1697. [CrossRef] [PubMed]
20. Hayes, J.; Danezis, G. Generating steganographic images via adversarial training. *arXiv* **2017**, arXiv:1703.00371.
21. Zhu, J.; Kaplan, R.; Johnson, J.; Fei-Fei, L. Hidden: Hiding Data with Deep Networks. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 657–672.
22. Zhang, K.A.; Cuesta-Infante, A.; Xu, L.; Veeramachaneni, K. SteganoGAN: High capacity image steganography with GANs. *arXiv* **2019**, arXiv:1901.03892.
23. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
24. Zhang, R.; Dong, S.; Liu, J. Invisible steganography via generative adversarial networks. *Multimed. Tools Appl.* **2019**, *78*, 8559–8575. [CrossRef]

25. Chan, C.K.; Cheng, L.M. Hiding data in images by simple LSB substitution. *Pattern Recognit.* **2004**, *37*, 469–474. [CrossRef]
26. Goel, A.K. An Overview of Image Steganography and Steganalysis based on Least Significant Bit (LSB) Algorithm. *Des. Eng.* **2021**, *2021*, 4610–4619.
27. Zhang, K.A.; Xu, L.; Cuesta-Infante, A.; Veeramachaneni, K. Robust invisible video watermarking with attention. *arXiv* **2019**, arXiv:1909.01285.
28. Luo, X.; Li, Y.; Chang, H.; Liu, C.; Milanfar, P.; Yang, F. DVMark: A Deep Multiscale Framework for Video Watermarking. *arXiv* **2021**, arXiv:2104.12734.
29. Woo, S.; Park, J.; Lee, J.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
30. Xu, B.; Wang, N.; Chen, T.; Li, M. Empirical evaluation of rectified activations in convolutional network. *arXiv* **2015**, arXiv:1505.00853.
31. Santurkar, S.; Tsipras, D.; Ilyas, A.; Mądry, A. How does batch normalization help optimization? In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; pp. 2488–2498.
32. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 448–456.
33. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; Springer: Cham, Switzerland, 2015; pp. 234–241.
34. 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.
35. Adler, J.; Lunz, S. Banach wasserstein gan. *arXiv* **2018**, arXiv:1806.06621.
36. Timofte, R.; Agustsson, E.; Gu, S.; Wu, J.; Ignatov, A.; van Gool, L. DIV2K Dataset: DIVerse 2K Resolution High Quality Images as Used for the Challenges @ NTIRE (CVPR 2017 and CVPR 2018) and @ PIRM (ECCV 2018). Available online: http://data.vision.ee.ethz.ch/cvl/DIV2K (accessed on 1 November 2020).
37. Available online: http://images.cocodataset.org/zips/train2017.zip (accessed on 1 November 2020).
38. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.