Convolutional Video Steganography with Temporal Residual Modeling

Xinyu Weng$^{1,2}$, Yongzhi Li$^{1,2}$, Lu Chi$^1$, Yadong Mu$^{1,2}$

$^1$Institute of Computer Science & Technology
$^2$Big Data Scientific Research Center
Peking University, China

wengxy@pku.edu.cn, yongzhili.pku@gmail.com, chilu@pku.edu.cn, myd@pku.edu.cn

Abstract

Steganography represents the art of unobtrusively concealing a secret message within some cover data. The key scope of this work is about visual steganography techniques that hide a full-sized color image/video within another. A majority of existing works are devoted to the image case, where both secret and cover data are images. We empirically validate that image steganography model does not naturally extend to the video case (i.e., hiding a video into another video), mainly because it completely ignores the temporal redundancy within consecutive video frames. Our work proposes a novel solution to the problem of video steganography. The technical contributions are two-fold: first, the residual between two consecutive frames tends to zero at most pixels. Hiding such highly-sparse data is significantly easier than hiding the original frames. Motivated by this fact, we propose to explicitly consider inter-frame residuals rather than blindly applying image steganography model on every video frame. Specifically, our model contains two branches, one of which is specially designed for hiding inter-frame difference into a cover video frame and the other instead hides the original secret frame. A simple thresholding method determines which branch a secret video frame shall choose. When revealing the concealed secret video, two decoders are devised, revealing difference or frame respectively. Second, we develop the model based on deep convolutional neural networks, which is the first of its kind in the literature of video steganography. In experiments, comprehensive evaluations are conducted to compare our model with both classic least significant bit (LSB) method and pure image steganography models. All results strongly suggest that the proposed model enjoys advantages over previous methods. We also carefully investigate key factors in the success of our deep video steganography model.

1. Introduction

The term steganography [7, 2] can date back to some ancient technique developed in the 15th century. The goal of steganography is to encode a secret message in some transport medium (called cover in this paper) and covertly communicate with a potential receiver who knows the decoding protocol. Essentially different from cryptography, steganography aims to hide the presence of secret communications, allowing only the target recipient to know. State differently, the covering medium can be publicly visible and yet only the target receiver can perceive the presence and decode the secret message. In practice, any steganography model should conceal a secret message by concurrently optimizing two criteria: minimizing the change of the covering medium that leads to suspect from an adversary, and reducing the residual between decoded secret message and its ground truth. The research on steganography has practical implications. For example, a number of nefarious applications of steganography techniques are known, such as hiding commands that coordinate criminal activities through images posted on social media websites. In the industry of digital publishing, a common tactic to claiming authorship without compromising the integrity of the digital content is to embed digital watermarks. For some brief introduction to steganography, one can refer to [16, 1, 12, 14].

Let us first explain the process of a typical steganography system, which is shown in Fig. 1. In classic steganography, the process involves three parties: Alice, Bob and Eve. Al-
Figure 2: Exemplar results generated by an image steganography model. The role of each image is depicted in bold yellow text located in the top-left of each image. To depict how container image deviates from the original cover image, we choose two local patches and contrast them for these two images. Indeed, for the local patch delimited by the green box, from the container image one can observe the ghost image of specific building in the secret image (in blue box). Better viewing after enlarging.

ice first conceals a secret message into a cover to obtain a container message (or steganographic message), and then sends the container message to Bob. Eve is an adversary (the steganalyzer) to both Alice and Bob. Each message that Eve observes is either cover or container. And Eve makes a binary classification on each message. His goal is to judge whether a message is steganographic or not. However, this steganalyzer is not requested to decode the hidden secret message. In this scheme, we say Alice performs perfectly if she ensures: 1) Bob receives the container message and successfully recover secret message at high accuracy using a decoding protocol; and 2) Eve, who always attempts to detect the presence of secret message, has exactly 50% chance of correctly judging a container or cover message. It is similar to the expectation in adversarial training [6, 4].

To accomplish both goals, the container message should not deviate from the original cover too much, avoiding that abnormal pattern appears and is detected by Eve. Meanwhile, it should also be in a good shape to be accurately deciphered by the decoder model at Bob’s hand.

Hiding messages in an image has been a long-standing research task of salient practical interest. One can gauge the amount of concealed information through bits-per-pixel (bpp), namely the amortized bits hidden at each pixel in the cover image. A recent research trend is hiding a full-sized color image into another same-sized image as exemplified in [2]. We hereafter term the task image steganography. This represents a highly challenging task since it pursues a bpp level of 1 (i.e., each pixel in the cover hides a complete RGB color). Fig. 2 illustrates a group of typical results calculated from an image steganography model, faithfully following the scheme depicted in Fig. 1. The steganography model can hardly accomplish both of Alice’s two goals in the container. As shown in Fig. 2, artifacts can often be observed in container, making it easily detected by an adversary. In recent year, to improve the performance of an image steganography model, researchers have explored deep neural networks for learning both encoding model (cover + secret → container) and decoding model (container → decoded secret) in above process. Successful applications are found in [7, 2].

In this work, our major focus is video steganography. The task aims to hide a full-sized video clip into another. Considering the increasing popularity of video data across the Internet, the research of video steganography, though currently rarely found in the literature, represents a nascent research topic of key practical implications. One may argue that image steganography model can be readily used to solve the video steganography problem, by pairing frames in cover / secret videos and feeding them into an image model. We argue that this tactic is not optimal, because it does not fully consider the temporal redundancy within
consecutive video frames. Our work proposes a novel solution to video steganography. Briefly speaking, the technical contributions are two-fold:

First, the residuals between two consecutive frames are highly sparse. Critically, compared with hiding frame into another frame, hiding such sparse residual in another video frame defines a much easier task. Motivated by this fact, instead of blindly applying image steganography model on all frames, we propose to split frames into two sub-sets: reference frames and residual frames. Each residual frame is obtained by differencing with specific reference frame. Correspondingly, our model contains two branches at both the encoding and decoding stages, tackling either type of frames respectively. We empirically validate this treatment can significantly boost the container’s perceptual quality and increase the possibility of fooling an adversary.

Secondly, our model is fully based on deep convolutional neural networks, which is the first of its kind in video steganography. Specifically, our deep video steganography model consists of two H-networks for hiding references or residuals, and two R-networks for revealing the secret video. The full model is trained without any human annotations and network parameters are optimized from scratch. In experiments, comprehensive evaluations are conducted to validate the powerful modeling of deep networks. We also carefully design ablation investigation to find key factors in our deep video steganography model.

The remainder of this paper is organized as following: We first briefly review the related work in Section 2. Section 3 details the proposed two-branch deep neural networks for the video steganography task. All experimental evaluations and in-depth analysis are found in Section 4. Finally, Section 5 concludes this work and points out several future research directions.

2. Related Work

Least significant bit (LSB) [15, 19] is a classic steganographic algorithm. It adopts a simple idea to embed a secret image into another. In digital images, each pixel in an image is comprised of three bytes (i.e., 8 binary bits), representing the RGB chromatic values respectively. The LSB algorithm replaces the least 4 significant bits of the cover image by 4 most significant bits of the secret image. For each byte, the significant bits dominate the color values. This way, the chromatic variation of the container image (altered cover) is minimized. Decoding the concealed secret image can be simply accomplished by reading the 4 least significant bits and performing bit shift. Despite that its distortion is not often visually observable, LSB is unfortunately highly vulnerable to steganalysis [5] - statistical analysis can easily detect the pattern of altered pixels. Recent works have been devoted to more sophisticated methods that preserve the image statistics or design special distortion functions, such as HUGO [17], WOW (wavelet obtained weights) [8], S-UNIWARD [9], and ATS [13].

The most relevant works to ours are two deep learning based image steganography methods in [7, 2]. Much earlier works [10, 18] adopts deep neural networks to elevate accuracies, yet mostly in the decoding process, such as determining which bits to extract from the container images. Both of [7, 2] build the whole system based on deep networks, including encoding (hiding), decoding (revealing) and adversarial networks. The quantitative evaluations strongly corroborate the superior modeling ability of deep networks. However, to our best knowledge, there is no prior work that explore deep networks for the hiding-video-in-video setting. This paper provides clear evidence that direct adaptation of image steganography model to video data is not an optimal choice and we are thus motivated to de-
vise special video steganography model based on temporal residual modeling.

3. The Proposed Model

Fig. 3 illustrates some motivating fact to our video steganography model. As seen, the residual values between consecutive video frames are dominated by near-zero values. Hiding such high-sparse data into a cover frame intuitively requires less effort compared with a full-colored secret frame, since hiding a zero value is trivial. This way, the cover image tends to be less altered, which potentially increases the chance of fooling an adversary. Using residuals as the secret message instead can ease Alice’s job (or the encoding model) in Fig. 1 and meanwhile does not make Bob’s task harder. However, to operate on residuals, there are two challenges that we should concern: how to determine encoding the original video frame or its residual with respect to the previous frame? And at the decoding stage, how the decoder knows the received image conceals a full-colored frame or a residual array?

To address above issues, we categorize all secret frames to be either reference frame or residual frame. Correspondingly, we propose to use two separate encoding / decoding networks for tackling different type of frames. The architecture of our proposed system is shown in Fig. 4. The system is comprised of five computational steps:

**Step-1: Reference/Residual Frame Labeling:** We adopt a simple thresholding approach for labeling a frame to be reference or residual type. Specifically, the first frame in a video is surely labeled as reference. The following frames in the same video sequentially calculate their average of the preceding frame.

![Diagram of computational pipeline](image)

**Figure 4:** The computational pipeline of our proposed video steganography model. See text for more details.
Table 1: Architecture of both Reference Hiding network and Residual Hiding network. There is a batch normalization layer (BN) and a Leaky Rectified Linear Unit (LeakyReLU) after each convolution layer. And there is a BN and a Rectified Linear Unit (ReLU) after each deconvolution layer except the last one. The output deconvolution layer is followed by a Sigmoid function.

| Index | Type     | Kernel | Stride | Padding | Input | Out | Concatenation |
|-------|----------|--------|--------|---------|-------|-----|---------------|
| 1     | Conv2d.  | 4×4    | 2      | 1       | 6     | 64  | N/A           |
| 2     | Conv2d.  | 4×4    | 2      | 1       | 64    | 128 | N/A           |
| 3     | Conv2d.  | 4×4    | 2      | 1       | 128   | 256 | N/A           |
| 4     | Conv2d.  | 4×4    | 2      | 1       | 256   | 512 | N/A           |
| 5     | Conv2d.  | 4×4    | 2      | 1       | 512   | 512 | N/A           |
| 6     | Conv2d.  | 4×4    | 2      | 1       | 512   | 512 | N/A           |
| 7     | Conv2d.  | 4×4    | 2      | 1       | 512   | 512 | N/A           |
| 8     | deConv2d.| 4×4    | 2      | 1       | 512   | 512 | N/A           |
| 9     | deConv2d.| 4×4    | 2      | 1       | 1024  | 512 | concat with layer #6 |
| 10    | deConv2d.| 4×4    | 2      | 1       | 1024  | 512 | concat with layer #5 |
| 11    | deConv2d.| 4×4    | 2      | 1       | 1024  | 256 | concat with layer #4 |
| 12    | deConv2d.| 4×4    | 2      | 1       | 512   | 128 | concat with layer #3 |
| 13    | deConv2d.| 4×4    | 2      | 1       | 256   | 64  | concat with layer #2 |
| 14    | deConv2d.| 4×4    | 2      | 1       | 128   | 3   | concat with layer #1 |

Table 2: Architecture of both Reference Reveal network and Residual Reveal network. Each layer is a Inception block with two kinds of kernel size (3×3 and 5×5), and the output is the concatenation of both feature maps. There is a batch normalization layer (BN) and a Rectified Linear Unit (ReLU) after each block layer except the last one. The output convolution layer is followed by a Sigmoid function.

| Index | Type     | Kernel | Stride | Padding | Input | Out |
|-------|----------|--------|--------|---------|-------|-----|
| 1     | Conv2d.  | 3×3 and 5×5 | 1 | 1 and 2 | 3 | 50×2 |
| 2     | Conv2d.  | 3×3 and 5×5 | 1 | 1 and 2 | 100 | 50×2 |
| 3     | Conv2d.  | 3×3 and 5×5 | 1 | 1 and 2 | 100 | 50×2 |
| 4     | Conv2d.  | 3×3 and 5×5 | 1 | 1 and 2 | 100 | 50×2 |
| 5     | Conv2d.  | 3×3 and 5×5 | 1 | 1 and 2 | 100 | 50×2 |
| 5     | Conv2d.  | 1×1    | 1      | 0       | 100  | 3   |

Step-2: Hiding Secret (encoding): This step does Alice’s job in Fig. 1. The key differentiator of our method to others is a divide-and-conquer scheme. Note that in Fig. 4, two hiding networks are devised, referred to as Reference H-net or Residual H-net respectively. Each frame is fed into the corresponding H-net by their label. It should be clarified that these two H-nets do not share any parameter. They are individually optimized for encoding specific type of frame only. In this paper we term the new frame, which appears similar to the cover yet conceals a secret somewhere, as container. In practice, we choose the U-net model [3, 11] for both H-nets. The network specifications are found in Table 1.

Step-3: Revealing Secret (decoding): It does Bob’s job in Fig. 1. The input is merely the container, and the output (we call it decoded secret) is another image which is desired to be exactly the secret in the perfect case. Otherwise the model is trained to minimize the discrepancy between the secret and its decoded version. Similar to H-nets, two R-nets (Reference R-net or Residual R-net) are introduced to reveal the frame or residual secret. However, unlike the encoding stage, Bob strictly has no access to the cover or secret, which implies that frame labels are missing. State differently, the decoder is not aware of which R-net is the optimal handler. We postpone this decision to the next step. The container frame will be sent to both R-nets and obtained two decoded secret images. The specification of R-nets is found in Table 2. It is clarified that two R-nets do not share parameters, despite the same network architecture.

Step-4: Frame-or-Residual Classification: Our proposed
temporal residual modeling raises new challenges to the classic scheme as depicted in Fig. 1 - Bob receives two copies of decoded secret messages in Step-3, from Reference R-net or Residual R-net respectively. Clearly, only one of the secret messages is true. Bob needs to pick out the real message. In fact, we can exhaustively enumerate all possible messages: the real reference and fake residual (container with a true reference secret gets through Reference and Residual R-nets respectively), real residual or fake reference (similar to above, but containers now carry residuals), totalling four valid cases. Therefore, we formulate it as a four-way classification problem. As seen in Fig. 2, a Reference-or-Residual (RoR) Net is devised for judging an input decoded message.

Similar to the R-nets, we let RoR Net have a mainframe of five convolutional layers, each of which is paired with BN layer and LeakyReLU. The key difference to R-nets is that the network head is a linear fully-connected layer followed by some softmax layer. Given an input image, the softmax eventually returns a 4-d probabilistic vector that categorizes the decoded information.

**Step-5: Residual Frame Reconstruction**: This step is optional if Step 4 judges a message as real reference. However, for a residual frame, it is not visually understandable per se. One need to add decoded residuals to the correct reference frame for obtaining the concealed video frame. Since we always process video frames in temporal order, we can record the latest reference frame for reconstructing residuals.

In our proposed system, H-nets / R-nets are jointly trained before the RoR net. The overall loss function for learning H-nets / R-nets is composed as individual loss defined on each networks. Recall that H-nets output container frame and R-nets return decoded reference or residuals. Following a typical treatment in image segmentation, for H-nets we define a loss on H-nets as summing all pixel-wise difference between container / cover, and a loss on R-nets for comparing decoded references / residuals and the original copies. For learning the RoR net, we adopt the standard cross-entropy loss to enforce label consistency.

### 4. Experiments

#### 4.1. Dataset Description and Experimental Setting

There is no available benchmark used for video steganography research. We therefore construct a new benchmark as follows: TRECVID Multimedia event detection (MED) is a yearly competition about retrieving specific semantic events (such as “birthday party” or “parking”) from a huge pool of videos. The MED 2017 video corpus consists of more than 0.3 Million videos with high-quality annotation. Since our task is essentially unsupervised, we ignore the video semantic labels and randomly sample 12,000 videos from the whole set. For each video, a 2-second clip is randomly cropped and 24 frames are extracted using the tool of FFmpeg. We generate a data split of training / validation / testing subsets, with 10,000, 1000, and 1,000 video clips respectively.

On all 10,000 training videos, our simple thresholding scheme generates 43,610 reference frames and 196,840 residuals. Videos are randomly drawn to form the (cover, secret)-pair. The Reference H-net is trained using all reference frames, and Residual H-net utilizes the residuals. All decoded messages collectively trains the four-way RoR net. We tune the network parameter following common tactics, such as decaying the learning rate after a fixed number of iterations and using momentum to keep the solution stable. The best model on the validation set is kept as the final model.

#### 4.2. Empirical Evaluation and Analysis

Fig. 5 shows the steganography results on selected videos. For each video, we show both the results of Reference H/R-nets and Residual H/R-nets. It is observed that secret videos often have different color tone and textures from the covering video, constructing a challenging task. By investigating the residuals between container-cover and secret-decoded secret pairs as in Fig. 5, we can observe that the container frames still look visually natural. In Table 3 we report the APD scores for the baseline least significant bits (LSB). Since our Reference H/R-net is alike to the work in [2], we separately report the APD scores of Reference H/R-nets (image model in the table) and the full architecture in Fig. 4. It is seen that our full model enjoys few distortions for the container frames. This may attribute to the separate processing of reference / residual frames. The decoded secret message by the full model is slightly worse, potentially caused by the residual reconstruction step. Both the image model and our video-based model significantly outperforms LSB. We also perform visual comparison with LSB and [2] in Fig. 6 and clear superiority goes to our model.

---

2The results are based on an unofficial re-implementation of this image steganography model in https://github.com/harveyslash/Deep-Steganography.
4.2.1 Where is the Secret Frame Encoded?

Following [2], we manually adjust the pixel values in containers and re-feed the modified container to R-net. By checking the changes in decoded secret frame one may discover where the secret is hidden. As illustrated in Fig. 7, we adjusted four 16×16 areas in a container. The left top one sets all pixels to white color, and other three set either of the RGB channels to 0, respectively. We have several observations: first, the decoded secret changes accordingly, with a large spatial field. It implies our encoding / decoding models are non-local, unlike LSB. Secondly, we find that image gradient matters in visual steganography. Zeroing a single color channel does not change much the image gradient of the inner pixels, which explains that the inner region is still accurately reconstructed. However, the boundary pixels, which suffers from severe gradient change, lose the correct secret message.
4.2.2 Investigation on Adversarial Learning

In steganography, a prominent goal is to fool the adversary, Eve in Fig. 1. An interesting problem to us is: after collecting how many labeled cover / container data, the adversary will become accurate enough to detect the presence of secret message? Without loss of generality, we assume the adversary uses a 6-layer CNN for learning a binary classification from labeled data. We investigate both LSB and our video model, as shown in Fig. 8. Interestingly, both methods tend to have zero probability of fooling the adversary after about 2,000 labeled data are leaked.

To increase the resistance to the adversary, we explore an idea of adding an adversarial learning sub-model, similar to [7]. Specifically, besides H/R-nets we incorporate an adversarial discriminator (we assume it has a common CNN architecture). It can receive frames from the cover and container to make a judgment. If the discriminator cannot distinguish a cover or container, it means that the container generated by H-nets is able to fool this discriminator. We adopt the classic GAN [6] loss for this discriminator. In one of our experiments, we combine this GAN loss with H/R losses, modulated via a hyper-parameter (set to 3e-4 empirically), and re-train the video model. The APD scores of the re-trained H/R-nets are 4.13 and 6.21, respectively, which are slightly worse than the original video model in Table 3. This implies that the new loss sacrifices APD in H/R-nets for lower detectability by the adversary.

After re-training, we plot the curve of the new model in Fig. 8 (denoted as “w/ GAN”). The apparent gap between the curves of original model (denoted as “w/o GAN”) and new model indicates that the latter is enhanced for anti-steganalysis. For example, for the original model, leaking 400 training pairs can enable the attacker to correctly distinguish 80% testing samples. While for the adversarially-trained new model, to achieve this accuracy, more than 600 pairs are required. This experiment serves a strong evidence that incorporating a GAN-style adversarial discriminator can lead to a more steganalysis-secure message embedding. It is also noted that, for LSB the adversary can easily perform shift operations on covers and containers to distinguish, making it less secure.

4.2.3 The Good and Bad Cover / Secret

It is an important problem to predict whether a pair of cover / secret has the potential to generate low APD score. To this end, we randomly select 100 videos as covers, and other 100 videos as secret videos to be hidden. They are exhaustively paired and tested by our video model. We record the pairwise APD scores and plot them on the left matrix in Fig. 9. Interestingly, we find that some videos tend to be a “good” cover or secret whatever the other party in the pair is, and vice versa. It becomes clear by looking at the white or black stripes in Fig. 9. For better understanding, we select the top 3 best or worse cover / secret and show them in the right of Fig. 9. It is observed that “good” ones tend to have few textures and lower saturation.
Figure 7: Finding how the secret message is concealed. Column 1: cover image. Column 2: modified container by manually adjusting four patches of original container image. Column 3: original decoded secret. Column 4: decoded secret image by re-feeding modified container to R-net.

Figure 8: Investigation on adversarial learning. The curves indicate how many leaked training samples are needed for an adversary to distinguish a container from a cover, which reflects the anti-steganalysis ability of a model.
Figure 9: Investigation of the “goodness” of covers or secret videos. Left: the matrix of inter-video APD scores (darker ones indicate lower APD scores). Right: examples of “good” or “bad” covers or secret.

Figure 10: Left: APD score between all 24 frames in a video with respect to the first. Right: the distribution of the lengths of video segments obtained via thresholding.

4.2.4 Effect of Frame Locations

We adopt a thresholding scheme to split reference and residual frames. However, choosing a proper threshold is non-trivial. In our experiment, we randomly selected 1000 short videos (24 frames each one) and calculated the average of every pixel’s residual from each frame to the first frame. The curve is showed in Fig. 10. A large threshold will generate more residuals, which tends to lead improved container quality yet may degrade the decoded secret. If we set a smaller threshold, there will be more reference frames, making the video model quickly converge to the image steganography. To search a good balance between H/R-nets, we finally selected a threshold of 30.68. Let us term a reference frame and its following residuals as a video segment. The right one in Fig. 10 plots the distribution of video segment lengths under the chosen threshold.

4.2.5 Reference-or-Residual (RoR) Network

As stated earlier, to categorize the decoded message we train a four-class CNN Reference-or-Residual (RoR) classifier. In practice, we use the trained Reference H/R-nets and Residual H/R-nets to collect training data, and use these data to train the RoR network. On the testing set, an accuracy of 99.9625% was achieved, which is nearly perfect yet the RoR network is still fooled by some hard samples. To attack this issue, we propose an improved judgment method. Because there are only two combinations of reference and residual values, i.e. real reference and fake residual or fake reference and real residual. Therefore, we add the probability of real reference and probability of fake residual as P1, the probability of fake reference and the probability of real residual was added up as P2. If P1 is larger than P2, we suppose that this container conceals reference information.
otherwise it hides residuals. This simple scheme brings a 100% accuracy on the test set.

5. Concluding Remarks

In this paper we present a novel deep neural network for the task of video steganography. To fully utilize the sparse property of inter-frame differences, we develop a temporal residual modeling technique, separately treating reference and residual frames during generating steganographic videos. Comprehensive evaluations and studies show the superiority of our method. The future work shall include the exploration of more sophisticated deep models, such as C3D [20], which may better handle failure cases as in Fig. 11.

References

[1] G. Abboud, J. S. Marean, and R. V. Yampolskiy. Steganography and visual cryptography in computer forensics. In SADFE, 2010.
[2] S. Baluja. Hiding images in plain sight: Deep steganography. In NIPS, 2017.
[3] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. CoRR, abs/1606.00915, 2016.
[4] G. F. Elsayed, S. Shankar, B. Cheung, N. Papernot, A. Kurakin, I. J. Goodfellow, and J. Sohl-Dickstein. Adversarial examples that fool both human and computer vision. CoRR, abs/1802.08195, 2018.
[5] J. J. Fridrich, M. Goljan, and R. Du. Detecting LSB steganography in color and gray-scale images. IEEE MultiMedia, 8(4):22–28, 2001.
[6] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.
[7] J. Hayes and G. Danezis. Generating steganographic images via adversarial training. In NIPS, 2017.
[8] V. Holub and J. J. Fridrich. Designing steganographic distortion using directional filters. In WIFS, 2012.
[9] V. Holub, J. J. Fridrich, and T. Denemark. Universal distortion function for steganography in an arbitrary domain. EURASIP J. Information Security, 2014:1, 2014.
[10] S. Husien and H. Badi. Artificial neural network for steganography. Neural Computing and Applications, 26(1):111–116, 2015.
[11] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In CVPR, 2017.
[12] G. C. Kessler and C. Hosmer. An overview of steganography. Advances in Computers, 83:51–107, 2011.
[13] D. Lerch-Hostalot and D. Megías. Unsupervised steganalysis based on artificial training sets. Eng. Appl. of AI, 50:45–59, 2016.
[14] B. Li, S. Tan, M. Wang, and J. Huang. Investigation on cost assignment in spatial image steganography. IEEE Trans. Information Forensics and Security, 9(8):1264–1277, 2014.
[15] J. Mieliokainen. LSB matching revisited. IEEE Signal Process. Lett., 13(5):285–287, 2006.
[16] T. Morkel, J. H. P. Eloff, and M. S. Olivier. An overview of image steganography. In ISSA, 2005.
[17] T. Pevný, T. Filler, and P. Bas. Using high-dimensional image models to perform highly undetectable steganography. In Information Hiding, 2010.
[18] L. Pihre, J. Pasquet, D. Ienco, and M. Chaumont. Deep learning for steganalysis is better than a rich model with an ensemble classifier, and is natively robust to the cover source-mismatch. CoRR, abs/1511.04855, 2015.
[19] K. Qazanfari and R. Safabakhsh. An improvement on LSB matching and LSB matching revisited steganography methods. CoRR, abs/1709.06727, 2017.
[20] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. C3D: generic features for video analysis. CoRR, abs/1412.0767, 2014.