STNet: A Style Transformation Network for Deep Image Steganography

Zihan Wang[1,2,3], Neng Gao[1,3], Xin Wang[3](✉), Ji Xiang[3], and Guanqun Liu[2,3]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, CAS, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangzihan,gaoneng,wangxin,xiangji,liuguanqun}@iie.ac.cn

Abstract. Image steganography is the technique of hiding information within images in plain sight. With the rapid development of deep learning in the field of steganalysis, it becomes a tremendous challenge to design a secure steganographic algorithm. To this end, we propose a novel steganographic network based on style transfer, named STNet. This network accepts the content and style images as input to synthesize art image with content from the former and style from the latter and embeds the secret information in style features. It can effectively resist most steganalysis tools. Steganalysis can identify stego images from cover images, but they cannot distinguish our stego images from other art images. Meanwhile, our method produces stego images of arbitrary size with 0.06 bit per pixel, improving over other deep steganographic models which only can embed fixed-length secret. Experiment results demonstrate that our STNet can achieve great visual effect, security, and reliability.

Keywords: Steganography · Style transfer · Deep learning

1 Introduction

Steganography and cryptography are both techniques used in secret communication. However, they have different goals that cryptography makes the secret data unreadable, while steganography hides the presence of secret communications. Image steganographic algorithm is designed to hide information into a static image. The carrier image for this purpose is called cover image, and the output image embedded with secret information is named stego image. On the contrary, steganalysis is the method to distinguish stego images from cover images.

In traditional image steganographic algorithms, the cover image embeds secret information by changing the values of some pixels, which are chosen by a suitably defined distortion function, such as HUGO [19] and WOW [7]. These methods design the distortion function with complex artificial rules. The secret information generally embeds into noisy regions or complex textures and avoids smooth regions of the cover image. They keep security by reducing the disturbances to the appearance and statistical features of the cover image. But they still face the risks of being detected, especially when the steganalysis algorithm is based on deep learning.

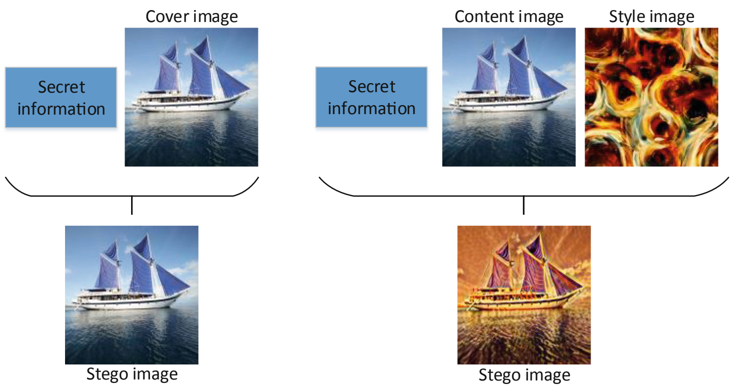

Fig. 1. The difference between the traditional steganographic method (left) and our STNet model (right). The traditional method generates the stego image by embedding secret information into the cover image. Our STNet accepts the content image and stego image as input and embed secret information into the style feature.

Nowadays, many researchers have attempted to introduce deep learning into the field of steganography, such as SteGAN [6] and HiDDeN [28]. These methods can automatically learn the steganographic strategy without any domain knowledge. They use neural networks instead of distortion functions to find pixels that are suitable for embedding secret information. However, they also face the security issue. Besides, one disadvantage of the deep learning steganographic algorithm is that it can only conceal fixed length of secret information. The size and capacity of stego images are fixed when the model has been trained.

To improve these situations, we attempt to find an innovative steganographic algorithm. CycleGAN [29] is a well-known image-to-image domain transformation model. However, recent research finds that CycleGAN is vulnerable to adversarial examples attacks [3]. It conceals information about source image into the target images imperceptibly, and can recover the source image from target images end-to-end. We explore whether it can be beneficial. We connect this phenomenon with steganography that neural networks can learn to encode a mass of secret information in the process of image domain transformation. In addition, nowadays makeup and art effect applications on images are widely used in our daily life, which makes it possible to transmit style transformation images unobtrusively in public media. So we propose a novel deep image steganographic model based on style transfer, which is called STNet (Fig. 1).

STNet is an end-to-end unsupervised training model without any domain knowledge. It accepts the content image and style image as input and recomposes content image in the feeling of style image. The secret information embeds in the style features of the stego image. We transmit the stego image in the public network, and the receiver will decode the secret information from the stego image successfully. Compared with previous steganographic methods, the main contributions of our work are as follows:

- To the best of our knowledge, STNet is the first steganographic method based on style transfer. It generates a new image that matches the content and style representation of two different source images. We design the model structure elaborately that secret information is embedded in style features of stego image.
- Our model has high security and can resist most existing steganalysis tools. The traditional steganographic algorithm has the cover and stego images pair so that the steganalysis tool can be trained to detect stego images. However, we generate the new art image as stego image directly. It is useless to detect out stego images from cover images because our style transformation images are indistinguishable from countless art images on the internet.
- Once the model has completed training, most deep learning steganographic algorithms can only output fixed size stego images with a fixed capacity. However, our model is a fully convolutional network and can accept arbitrary size content and style images as input. We introduce adaptive instance normalization (AdaIN) to perform style transfer, and the size of the stego image is the same as content image. Therefore, we can embed arbitrary bits of secret information.
- We conduct experiments on COCO and wikiart dataset. It produces visually pleasant style transformation results with high security and reliability.

The rest of the paper is organized as follows. Section 2 introduces the related work of steganography and style transfer. Section 3 describes our model architecture and learning objectives. Section 4 presents our experimental results and analysis. The conclusion and future works are presented in Sect. 5.

2 Related Work

2.1 Steganography

In the traditional steganographic algorithm, least significant bit (LSB) is a general steganographic algorithm that it hides the information by replacing the least significant bits of the cover image pixels. The changes of the image are imperceptible, but it does not preserve the statistical characteristic of the image and makes it vulnerable to the steganalysis. In order to solve this problem, most image steganographic methods use a distortion function that finds the pixel with the least disturbance to the image, thereby preserving image statistical characteristic. Highly Undetectable Steganography (HUGO) [19] defines the distortion function as the weighted difference of feature vectors, which already used in steganalysis algorithms. Wavelet Obtained Weights (WOW) [7] uses directional high-pass filters to force the embedding focus on highly textured or noisy regions. Universal wavelet relative distortion (UNIWARD) [8] can embed in an arbitrary domain, it computers the relative changes of directional filter bank decomposition of the image.

In recent years, a large number of steganographic algorithms try to combine with deep learning. ASDL-GAN [22] and VT-SCA-GAN [26] first propose to use generative adversarial network instead of distortion function to decide the embedding position. SGAN [24] and SSGAN [20] generate new cover images that are most suitable for embedding by the classical steganographic algorithm. For further performance, SteGAN and HiDDeN propose an end-to-end model that neural network directly outputs the final stego image. They achieve the best performance of current steganography with generative adversarial networks. SSteGAN [25] directly produces realistic and secure stego images with the noise and secret information as input instead of cover images. There also have some studies attempt to embed color or grayscale image in another color image of the same size [1,27]. Compared with other steganographic algorithms, they reduces the requirements for recovering secret data accurately, which does not interfere with image comprehension.

2.2 Style Transfer

Style transfer is an art of modifying the style of an image while preserving its content. Gatys et al. [5] first propose a way that iteratively updates the input image by minimizing the content loss and style loss based on features extracted from a pre-trained network. This method is computationally expensive since each iteration optimization requires both forward and backward propagation. [10,12,23] proposes a variety of fast feed-forward network models to reduce the heavy computational cost. However, these networks are trained to transfer multiple content images to a fixed style. To address this problem, Dumolin et al. [4] introduce a single network with conditional instance normalization that can represent 32 styles at the same time. Huang et al. [9] propose arbitrary style transformation network by matching the mean and variance of content and style features. Li et al. [13] boost the style transformation performance by learning linear transformations between content feature and transformation matrix.

3 Model

As shown in Fig. 2, our model consists of three parts: style transformation network, reveal network, and loss network. The style transformation network is also the steganographic network. It embeds secret information in the high-frequency style features of the output image while the style is being converted. The reveal network accepts the stego image as input and accurately recovers secret information. The loss network is a pre-training model used to calculate the style and content loss of the stego image. Next, we will detail each part of the model.

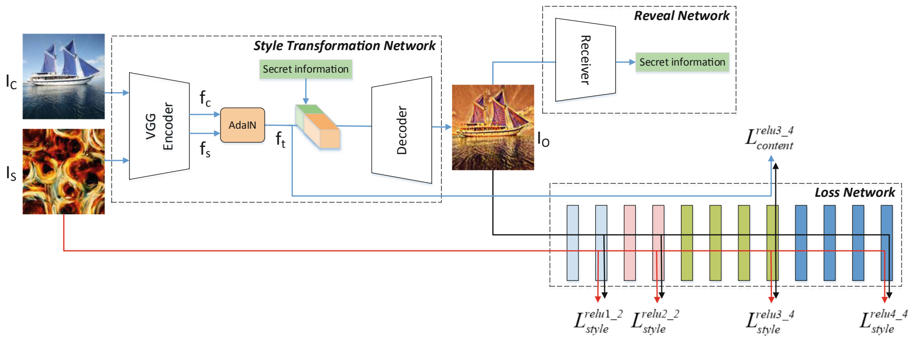

Fig. 2. Overview of the proposed method. Our model contains a style transformation network, a reveal network, and a loss network. We train the style transformation network to transform content and style image to stego image. The reveal network is used to decode secret information from the stego image. The loss network defines the loss function to train style transformation network.

3.1 Style Transformation Network

The VGG Encoder, Decoder, AdaIN layer, and secret information constitute the style transformation network. It takes an arbitrary content image I_c and style image I_s as inputs and produces the stego image I_o that simultaneously matches the content representation of the I_c and the style representation of the I_s. As we all know, convolutional neural networks are useful in image processing tasks. It uses multiple filters to extract specific features of the input image. In particular, when convolutional neural networks are trained for image classification, they convert the input image into a high-dimensional representation of the object information. That is to say, the output of a convolutional neural network contains the style or content characteristics of the input image. We use the Encoder to get the feature representation f_c and f_s of the content image and the style image. In our experiment, Encoder is the first fewer layers of VGG-19 [21] network pre-trained on the ImageNet dataset for image classification.

Then we use an adaptive instance normalization (AdaIN) layer [9] to perform style transformation in the feature space. It receives the feature map f_c and f_s as input and output target feature representation f_t. AdaIN has no learnable weight parameters, it aligns the mean μ and variance σ of content feature f_c to match those of style feature f_s in channel-wise.

$$AdaIN(f_C, f_S) = \sigma(f_S)\left(\frac{f_C - \mu(f_C)}{\sigma(f_C)}\right) + \mu(f_S) \tag{1}$$

Similar to the method of batch normalization (BN) and Instance Normalization(IN), we simply scale the normalized content input with $\sigma(f_s)$, and shift it with $\mu(f_s)$. Now, we get a new feature f_t which contains the high-level content feature of the I_c and the texture information of the I_s.

The Decoder is used to convert the feature map back to the image space. We first concat the secret information M with the output of AdaIN layer f_t as the input of the Decoder, and then the Decoder outputs the stego image I_o, which embeds the secret information into the image style feature. Its network structure is the opposite of Encoder. We use Nearest-Neighbor upsampling instead of fractionally-strided convolutions to replace all pooling layers because it can reduce checkerboard artifacts [18] effects. Meanwhile, we use reflection padding in both Encoder and Decoder to avoid border artifacts.

3.2 Reveal Network

The reveal network receives the stego image as input and recovers secret information $\hat{M}$ end-to-end. It uses a series of deep convolutional networks to extract secret information from the stego image. The reveal network is made up of six convolution layers with batch normalization, and Leaky ReLU activation function, except the final layer which uses Tanh and without batch normalization. The reveal network's goal is to recover secret information $\hat{M}$ accurately. Concretely, We use the squared Euclidean distance as secret information reconstruction loss. Note that L indicates the number of bits of the secret information.

$$L_m(M, \hat{M}) = \left\| M - \hat{M} \right\|_2^2 / L \tag{2}$$

3.3 Loss Network

The loss network also is the pre-training VGG-19 network. The parameters of VGG encoder remain unchanged during the training process. Different from the traditional steganographic mode, which changes the pixel value of the cover image as little as possible, we pay more attention to the content and style of the image compared to specific pixel values. We attempts to measure perceptual differences in content and style of input and output images, and define the content loss L_c and style loss L_s for training the style transformation network.

Content Loss. Let $\phi_l(x)$ be the feature maps of the specific layer l for the VGG network when the x as input. The content loss is the squared Euclidean distance between the target features and the features of the output image. We hope that the secret information should be as unrelated as possible to content features of the stego image. We use the AdaIN output f_t as the target feature, rather than the feature of the content image.

$$L_c = \frac{1}{C \times H \times W} ||\phi(f_t) - \phi(f_c)||_2^2 \tag{3}$$

Style Loss. There are two general methods to compute style loss and produce similar results: perceptual loss [10] and BN Statistics Matching [14]. We use the BN Statistics Matching for the convenience of calculation. We construct style loss by aligning the mean and standard deviation of two feature maps between the style image and the stego image.

$$\mathcal{L}_s = \frac{1}{N}\sum_{i=1}^{N}\left(\left\|\mu\left(\phi_i(I_o)\right) - \mu\left(\phi_i(I_s)\right)\right\|_2^2 + \left\|\sigma\left(\phi_i(I_o)\right) - \sigma\left(\phi_i(I_s)\right)\right\|_2^2\right) \tag{4}$$

Where $\mu(\phi_i(I_s))$ and $\sigma(\phi_i(I_s))$ is the mean and standard deviation of the loss network i-th layer. In our experiment, We adopt the $relu3\_4$ layer for the content loss, and $relu1\_2$, $relu2\_2$, $relu3\_4$, $relu4\_4$ for the style loss.

Total Loss. In total, we use the weighted sum of the content loss, style loss, and secret information reconstruction loss as the total loss function of STNet. α and β are weight factors that control the importance of each loss terms.

$$L_{total} = L_C + \alpha L_S + \beta L_m \tag{5}$$

4 Experiments

In this section, we first briefly describe some experimental implementation details. Then we present results of STNet and compare with other style transfer methods. Furthermore, we analyze the capacity and accuracy of secret information recovery. In the following, we evaluate the security and reliability of our steganographic model.

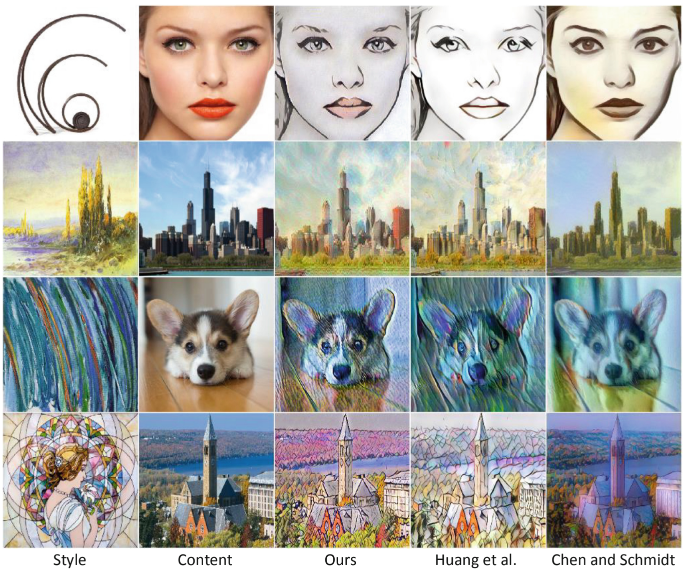

Fig. 3. Comparison between results by our style steganographic method and other style transfer algorithms. All images are resized to 512 × 512 pixels to facilitate presentation.

We conduct our experiment using COCO dataset [16] as content images and a dataset of paintings obtain from wikiart.org [17] as style images, following the setting of [2,9]. Each dataset has approximately 80,000 natural images or paintings as training examples. We resize each image to 512×512 pixels, then randomly crop regions of size 256×256. All model is trained with a batch size of 8 with epochs 8. We use Adam optimizer [11] with the learning rate of 1×10^{-4} and default hyperparameters. At each batch, we train style transformation network and reveal network at the same time.

In Fig. 3, we show the output image of our model and compare with other comparative models. Our results are qualitatively similar to comparative models. Due to the difference in architecture, loss function, style weight, etc. The output images produced by different models are slightly different. Overall, the secret information is embedded in the style feature and does not affect the content of the output image. The change of the style result is imperceptible to humans.

In order to evaluate the accuracy of secret information recovery, we randomly select 10,000 content and style images to generate 10,000 stego images as test images. Note that all the test images never appear in the training set. Reveal network can successfully decode 99.8% of the secret information. In practical application, in order to achieve 100% accuracy, we can introduce error correction coding (ECC) to correct the error bits. Different from other deep steganographic algorithms, our model is a fully convolutional network. Thus, we can input content and style images of arbitrary size, and the output image has the same size as the content image with the capacity of 0.06 bpp. We can choose the appropriate size image according to the length of the secret information. In comparison, other deep learning steganographic models can only embed fixed-length secret information when the model has completed training. The maximum stego image size and capacity of different models are shown in Table 1.

Table 1. The maximum image size and capacity of different steganographic algorithms.

Model	Our model	SteGAN	HiDDeN
Stego image size (max)	Arbitrary size	32×32	16×16
Capacity (max)	0.06 bpp	0.4 bpp	0.2 bpp
Secert information (max)	Arbitrary length	409 bit	51 bit

4.1 Security

The traditional image steganographic algorithm hides secret information by modifying the value of some specific pixels and minimize the disturbance to image appearance and statistical properties. Most steganalysis assumes that it knows the steganographic algorithm and get a large number of cover and stego images pair for training. Then the model can detect whether the image contains secret information, when it receives an image as input. Different from the traditional steganographic algorithm, we directly generate a new art image with secret information embed in style feature. There exists an essential difference between our method and the traditional steganographic algorithm so that our method can resist most existing steganalysis tools. The task of steganalysis has become to distinguish our style transformation images from countless art images.

To evaluate the security of our model, we built a steganalysis model to distinguish our stego image from other style transformation images. We denote the combination of the convolutional neural network, batch normalization and Leaky ReLU as a Conv-BN-LReLU block. The steganalysis model consists of five Conv-BN-LReLU blocks, three maxpooling layers and a dense layer. For the training dataset, positive samples are the stego images generated by our STNet. The outputs of the other three style transformation models [9,13,15] are regarded as negative samples. After training, we test the classification accuracy of the steganalysis model.

There are four sets of different data for testing, generated according to the following experimental conditions: C1, We use the model used in training to generate positive and negative samples. C2, We train a new STNet with different random seed and weight factors (α and β) to generate images as positive samples. C3, We train a new STNet with different hyperparameter (whether to use BN or not) to generate images as positive samples. C4, We train a new STNet with different training dataset to generate images as positive samples. At the same time, we use another style transformation model [2] to generate the negative samples and used in C2, C3 and C4.

The experiment results are shown in Table 2. C1 achieves a detection accuracy of 0.993, which indicate that our steganalysis model performance well. The steganalysis model can precisely distinguish the stego image from the image generated by the model used in training. However, in case C2, C3, and C4, when we detect images generated by models that do not appear in training, the accuracy of detection has dropped rapidly. There are a variety of style transformation networks and countless art images on the internet. It is a great challenge to train a steganalysis model to distinguish our stego images from all art images. What is more, the steganalysis knows our algorithm but does not have access to the trained model itself (it does not know the hyperparameters and dataset). Experimental results demonstrate that our model has high security, and our stego images are indistinguishable from art images on the internet.

Table 2. The accuracy of the steganalysis algorithm according to experimental conditions C1-C4. C1: baseline. C2: different random seed and weight factor. C3: whether to use BN or not. C4: different training data.

Experimental conditions	$C1$	$C2$	$C3$	$C4$
Classification accuracy	0.993	0.786	0.592	0.753

4.2 Reliability

In particular, there has a threat scenario that the adversary knows STNet is being used and tries to decode the stego image. To test the reliability of our model, we assume that the adversary knows the network structure, hyperparameters, and dataset, but does not have the random seed (which is used to initialize the parameters and can affect the values of final model parameters).

We have two separately STNet. One model is used to embed secret information and produces the stego images. The other model attempts to predict secret information from these stego images. We first train two models with the same training conditions (hyperparameters and dataset). The accuracy of secret information recovery can reach 0.99. In the second experimental case, we use different random seed while other training conditions stay the same. It cannot get any useful information with the decoding accuracy of 0.39 (0.5 equivalent to random guess). This result demonstrates that our model has a high reliability. Even if the adversary masters the algorithm and the stego image, they still can not get the secret information. To further improve steganographic security, we can encrypt the secret information before concealing it into cover images (Table 3).

The neural network of our model can automatically learn the unique steganographic strategy, and the strategy is similar when we train the model with identical training conditions. In the actual application scenario, it is very danger and trouble to transmit the entire model parameters directly in the public channel. Therefore, the two communicating parties can share the training conditions instead of transmitting the entire model parameters. They train the model locally with the same training conditions. Then the sender uses its style transformation network generates a stego image, and the receiver uses its reveal network decode the secret information successfully. What is more, we can always get a new steganographic system by merely altering training conditions.

Table 3. The accuracy of secret information recovery on experimental conditions C5 and C6. C5: two models have the same training conditions. C6: two models have different random seed value.

Experimental conditions	$C5$	$C6$
Decoding accuracy	0.99	0.39

5 Conclusion

In this paper, we propose a novel image steganographic method based on the style transfer. It is an end-to-end unsupervised neural network without any steganographic domain knowledge. The secret information embeds in the style features of the stego image. Our STNet can resist the most existing steganalysis algorithm, and its stego images are indistinguishable from art images on the internet.

For capacity, STNet can produce stego images of arbitrary size with 0.06 bit per pixel. That is to say, we can conceal the arbitrary length of secret information. The experiment results demonstrate that our method is competitive with existing steganographic algorithms on visual effect, capacity, security, and reliability. However, our model has the same limitations as other deep steganographic models. Compared with the traditional steganographic method, it needs extra cost to train the whole neural network. In future work, we will explore to apply our proposed scheme to other domains, such as text and video.

Acknowledgement. This work is supported by the National Key Research and Development Program of China, and the National Natural Science Foundation of China (No. U163620068).

References

1. Baluja, S.: Hiding images in plain sight: deep steganography. In: Advances in Neural Information Processing Systems, pp. 2066–2076 (2017)
2. Chen, T.Q., Schmidt, M.: Fast patch-based style transfer of arbitrary style. arXiv preprint arXiv:1612.04337 (2016)
3. Chu, C., Zhmoginov, A., Sandler, M.: CycleGAN, a master of steganography. arXiv preprint arXiv:1712.02950 (2017)
4. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artisticstyle. In: Proceedings of ICLR, vol. 2 (2017)
5. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2414–2423 (2016)
6. Hayes, J., Danezis, G.: Generating steganographic images via adversarial training. In: Advances in Neural Information Processing Systems, pp. 1954–1963 (2017)
7. Holub, V., Fridrich, J.: Designing steganographic distortion using directional filters. In: 2012 IEEE International Workshop on Information Forensics and Security (WIFS), pp. 234–239. IEEE (2012)
8. Holub, V., Fridrich, J., Denemark, T.: Universal distortion function for steganography in an arbitrary domain. EURASIP J. Inform. Secur. **2014**(1), 1 (2014)
9. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1501–1510 (2017)
10. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Li, C., Wand, M.: Precomputed real-time texture synthesis with markovian generative adversarial networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 702–716. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_43
13. Li, X., Liu, S., Kautz, J., Yang, M.H.: Learning linear transformations for fast arbitrary style transfer. arXiv preprint arXiv:1808.04537 (2018)

14. Li, Y., Wang, N., Liu, J., Hou, X.: Demystifying neural style transfer. arXiv preprint arXiv:1701.01036 (2017)
15. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Advances in Neural Information Processing Systems, pp. 386–396 (2017)
16. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
17. Nichol, K.: Painter by numbers (2016). https://www.kaggle.com/c/painter-by-numbers
18. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016). http://distill.pub/2016/deconv-checkerboard/
19. Pevný, T., Filler, T., Bas, P.: Using high-dimensional image models to perform highly undetectable steganography. In: Böhme, R., Fong, P.W.L., Safavi-Naini, R. (eds.) IH 2010. LNCS, vol. 6387, pp. 161–177. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16435-4_13
20. Shi, H., Dong, J., Wang, W., Qian, Y., Zhang, X.: SSGAN: secure steganography based on generative adversarial networks. In: Zeng, B., Huang, Q., El Saddik, A., Li, H., Jiang, S., Fan, X. (eds.) PCM 2017. LNCS, vol. 10735, pp. 534–544. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77380-3_51
21. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
22. Tang, W., Tan, S., Li, B., Huang, J.: Automatic steganographic distortion learning using a generative adversarial network. IEEE Signal Process. Lett. **24**(10), 1547–1551 (2017)
23. Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.S.: Texture networks: feed-forward synthesis of textures and stylized images. In: ICML, p. 4 (2016)
24. Volkhonskiy, D., Nazarov, I., Borisenko, B., Burnaev, E.: Steganographic generative adversarial networks. arXiv preprint arXiv:1703.05502 (2017)
25. Wang, Z., Gao, N., Wang, X., Qu, X., Li, L.: SSteGAN: self-learning steganography based on generative adversarial networks. In: Cheng, L., Leung, A.C.S., Ozawa, S. (eds.) ICONIP 2018. LNCS, vol. 11302, pp. 253–264. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04179-3_22
26. Yang, J., Liu, K., Kang, X., Wong, E.K., Shi, Y.Q.: Spatial image steganography based on generative adversarial network. arXiv preprint arXiv:1804.07939 (2018)
27. Zhang, R., Dong, S., Liu, J.: Invisible steganography via generative adversarial networks. Multimed. Tools Appl. **78**(7), 8559–8575 (2019)
28. 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), pp. 657–672 (2018)
29. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232 (2017)