PixInPix: Hidding Pixels in Pixels

Anonymous ECCV submission

Paper ID 24

Abstract. PixInPix is an steganography hidding system of images within other images. The system designed is able to create, from an cover image and a message, a new steganography image. This new stego-image looks as similar as possible as the cover but has the message hidden in it. Our approach adopts the U-net architecture and combines two reconstruction losses to provide a simple yet effective approach tested in low resolution images from MNIST, CIFAR and ImageNet.

Keywords: Steganography, Deep Learning, U-Net

1 Introduction

Steganography and watermarking are two well known techniques to hide information in media content such as images, videos or audio files. The main difference between them is related to what is hidden in them. While in watermarking the information hidden is related to its content, in steganography the hidden part usually do not have any relation with its content. The work we present tackles the field of steganography, however it could be easily reformulated into any watermarking problem.

Some works with convolutional neural networks [13, 16] have addressed the lossless case in which binary messages were hidden in a steganography image, studying the perturbations with respect to the original cover image. Our work addresses the scenario in which perturbations in the message are also acceptable, as this would be the case of visual information. Images are already typically encoded with lossy compression algorithms whose distortions are unnoticeable for humans. Our *PixInPix* model solves the task of hiding a message (or secret) image into a cover images, as task previously addressed in [1, 3]. In our case we provide a lightweight implementation that does not require any pre-processing of the message image. Hiding images into other images has applications in encoding depth, multi-view or short animations withing n a standard RGB image.

PixInPix follows a classic encoder-decoder paradigm which are trained together to hide images into other images. The encoder takes care of hiding a message into a cover image in such a way that pixel perturbations are hardly noticeable by a human eye. In our experiments, we apply different distortions and transformation on the encoded image and assess their impact in the recovery of the message image.

Our source code and trained models will be publicly available upon acceptance[1].

2 Related work

Hiding information in image pixels is a task extensively addressed in the broad field of steganography with multiple and diverse approaches [4–6, 10, 11, 15]. Recently, deep neural networks have achieved outstanding results, in parallel with several other task in the field of computer vision. They mostly follow the same basic architecture we adopt, with a convolutional neural network as encoder to hide the message with the cover image, and a convolutional decoder that outputs the message. A baseline system would define a reconstruction loss for the cover image at the output of the encoder, and a second reconstruction loss for the message at the output of the decoder.

Hidden [16] encodes the message by replicating and concatenating it to the pixel embeddings obtained by a 2D convolutional encoder, while in our case we concatenate the message directly into the cover image. *Hidden* incorporates an adversarial loss to improve the realistic appearance of the stego image. The work provides a detailed study of the trade-offs between capacity, secrecy and robustness of the method. *Stegastamp* [14] follows a similar approach, but focuses in the specific application of encoding hyperlinks into image pixels, reporting satisfactory results in practice for up to 56 bits per message.

Both *Hidden* and *Stegastamp* limit their study to the encoding of binary messages in the form of vector, while our work addresses scenarios in which some losses are acceptable. *PixInPix* presents many points in common with *Deep-Stego* [1], which also hides images into images, but requires a specific pre-processing of the message. A similar pre-processing scheme was later adopted by Duan et al. [3] and extended by adopting a U-Net [12] neural architecture and a BEGAN adversarial training, a similar set up as the one adopted in the popular *Pix2Pix* [7] model for image translation. *PixInPix* also benefits from the U-Net [12] architecture but adopts a simpler approach because it does not apply any pre-processing neither adversarial loss.

3 System architecture

The architecture of our system is shown in Figure 1. It is composed of an encoder and a decoder deep neural network that are trained together to hide a message image in the cover image.

We feed the network with a 6 channel 32x32 pixel input. Three channels for each image (cover and message). The output of the network is a 32x32 RGB image. We used low resolution images to ease the training process, as larger images take more computational time and resources.

[1] http://anonymous.url

Fig. 1: Overall system structure

The message encoder is the convolutional neural network in charge of creating the new image, the steganography image. This stego image should look as similar as possible to the cover image, despite containing the message image hidden within.

The first layer of the encoder the down-sampling process was designed with multiple 2D convolutional layers followed by a ReLU activation function. In each down-sampling step, we double the number of feature channels and we add a max pooling layer at the end. In particular, we used five down-sampling steps to go from a 6 channels to 256 feature channels. In the up-sampling phase, we used the same structure. Several 2D convolutional layers preceded by 2D transposed convolution operations. At each up-sampling step the feature map from the down-sampling process is add as part of the input to improve the final results. At the end, the output we obtain is a 3 channel 32x32 RGB image.

The message decoder contains almost the same layers as the image encoder, with the only difference that the input is the stego image and the output the reconstructed message image.

Both encoder and decoder networks follow a U-Net-like architecture [12] that allow a better recovery of the spatial resolution thanks to the skip connections between the down-sampling and up-sampling layers. The details of the message encoder are depicted in Figure 2.

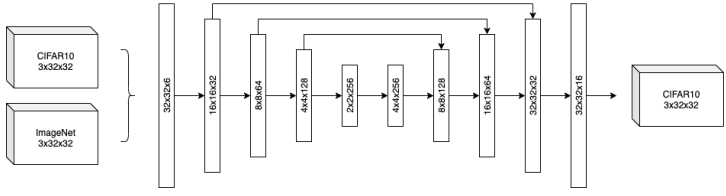


Fig. 2: Image encoder architecture

4 Experiments and results

In our experiments we used three well-known low resolution datasets for fast prototyping: MNIST [9], CIFAR10 [8] and a down-sampled ImageNet [2]. We used 50,000 images for training and 10,000 images for testing. We trained the model for 100 epochs and set the learning rate to 0.01 and used the SGD optimizer and mean square error (L2) as loss.

The particularity of the training procedure is that two loss terms needed to be balance: a reconstruction loss for the stego-image compared to the cover image, and a reconstruction loss for the message image. The two L2 were linearly weighted and their parameters were set after a naive hyper-parameter search.

We performed two types of experiments hiding both graysalce and RGB images into RGB images. First, an MNIST image was hidden into a CIFAR10 image (Figure 3) and an ImageNet image was later hidden into a CIFAR10 image (Figure 4).

5 Conclusions

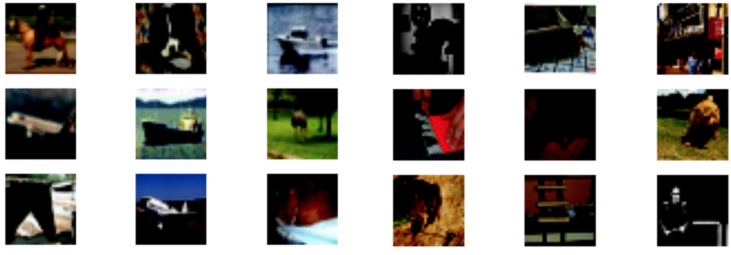
PixInPix offers a lightweight solution for hiding grayscale and color images into RGB images by exploiting the U-Net architecture and combining two reconstruction losses: one for the stego-image and another one for the message image. Our qualitative results show how this simple scheme can successfully hide images in low resolution set ups and provide a solid ground to extend the work to more complex scenarios.

Future work should address generating higher quality images and exploring other applications that may benefit from hiding in images, or the opposite.

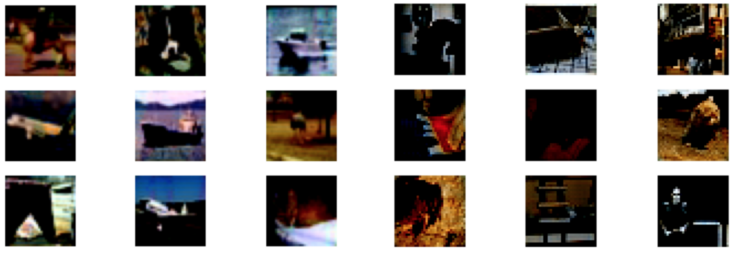
(a) Cover image. (b) Message.

(c) Stego- image. (d) Message reconstruction.

Fig. 3: Outputs using CIFAR10 and MNIST.



(a) Cover image. (b) Message.



(c) Stego- image. (d) Message reconstruction.

Fig. 4: Outputs using CIFAR10 and ImageNet.

References

1. Baluja, S.: Hiding images in plain sight: Deep steganography. In: Advances in Neural Information Processing Systems. pp. 2069–2079 (2017)
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
3. Duan, X., Jia, K., Li, B., Guo, D., Zhang, E., Qin, C.: Reversible image steganography scheme based on a u-net structure. IEEE Access **7**, 9314–9323 (2019)
4. Guo, L., Ni, J., Shi, Y.Q.: An efficient jpeg steganographic scheme using uniform embedding. In: 2012 IEEE International Workshop on Information Forensics and Security (WIFS). pp. 169–174. IEEE (2012)
5. 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)
6. Holub, V., Fridrich, J., Denemark, T.: Universal distortion function for steganography in an arbitrary domain. EURASIP Journal on Information Security **2014**(1), 1 (2014)
7. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
8. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-10 (canadian institute for advanced research) http://www.cs.toronto.edu/ kriz/cifar.html
9. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), http://yann.lecun.com/exdb/mnist/
10. Mielikainen, J.: Lsb matching revisited. IEEE signal processing letters **13**(5), 285–287 (2006)
11. Pevnỳ, T., Filler, T., Bas, P.: Using high-dimensional image models to perform highly undetectable steganography. In: International Workshop on Information Hiding. pp. 161–177. Springer (2010)
12. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
13. Tancik, M., Mildenhall, B., Ng, R.: Stegastamp: Invisible hyperlinks in physical photographs. CoRR **abs/1904.05343** (2019), http://arxiv.org/abs/1904.05343
14. Tancik, M., Mildenhall, B., Ng, R.: Stegastamp: Invisible hyperlinks in physical photographs. In: CVPR (2020)
15. Yedroudj, M., Comby, F., Chaumont, M.: Steganography using a 3 player game. CoRR **abs/1907.06956** (2019), http://arxiv.org/abs/1907.06956
16. 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)