Embedding Free, Adversarially Trained, Generative Steganography

Trevor Maxfield ~ maxfit@stanford.edu

Abstract

Using a generative adversarial network design we illustrate a method for embedding free, adversarially trained, image steganography; hiding a message in the process of generating a novel image rather than embedding the message into an existing cover image. This process is trained against a detection adversary and alongside a decoder, encouraging high-fidelity reconstruction of the message and low rates of detection. Furthermore, without the natural existence of a cover image in which the data is embedded, a real-world adversary will have a significantly harder time discovering the hidden message, even with open access to the dataset.

1 Introduction

Image steganography is the process of encoding a hidden message into a cover image in an attempt to avoid detection of the hidden information. Traditionally this is achieved though methods like placing the message in dark regions of the image or in the least significant bit(s) of each pixel, however these types of encodings are often detectable through statistical methods, such as StegExpose Boehm [2014]. With the capabilities of deep networks to encode data and generate novel images, a variety of research has been done on adapting these types of networks to encode/decode hidden information.

This project uses a generative adversarial network (GAN), Goodfellow et al. [2014] to simultaneously train an encoder/decoder while being subject to a detection adversary that attempts to identify hidden-payload containing images. This will not require the use of a cover image, meaning that the hidden message is directly input to the generator to create the image containing the message instead of the hidden message being encoded in an existing image. This provides the benefit that no cover image explicitly exists, as the generator maps messages to images directly, restricting the amount of information about the encoded message an adversary can obtain.

Explicitly, a bit-string message is mapped to an image through the generator. A discriminator accepts these images alongside real images from the dataset to determine real or fake, simultaneously learning the presence of an embedded message. Finally a decoder takes the images from the generator and learns a mapping back to the hidden bit-string message. See Figure 1 for a diagram.

2 Related work

Related work in deep steganography includes Baluja [2017], a deep steganography model with no detection adversary, Wu et al. [2018] a deep residual network design for steganalysis (detection of messages in images), and Wang et al. [2019] which uses style transfer techniques for embedding messages into covers. These works provide good insight into the modern state of deep steganography, both in embedding and detection techniques.

In Hu et al. [2018], a deep steganography without embedding (SWE) model is given. A standard GAN is trained on a dataset, and in a second step an extractor is trained to recover the input noise to the generator. Message data is then manually encoded into the random generator input to "seed" the generator output. This model is not trained against a detection adversary and achieves a relatively low bpp of ≈ 0.07 to ≈ 0.21, although it has the advantage of working without a cover image.

In Zhang et al. [2019], cover images are input to a generator along with a hidden message. The generator is trained against a detector and a decoder, so that high fidelity encodings and low-detection embeddings of the message into the cover image are achieved. A high rate of bits per pixel of up to 4.4 is achieved, although a cover image (a potential vulnerability) is still required.

We develop a model, that combines aspects of both of these papers. Using the idea of embedding free steganography from Hu et al. [2018], no input cover image will be required. Instead the generator input will be the data to be hidden. However, instead of separately training a network to recover this, a decoder will simultaneously be trained to retrieve the hidden message, while a detector is trained to encourage stealthier embeddings, both ideas from Zhang et al. [2019].

3 Data

Four datasets are used. Preliminary models use MNIST digits,Deng [2012], fashion-MNIST, Xiao et al. [2017], (both $28 \times 28 \times 1$, 60,000 images), and CIFAR10, Krizhevsky et al. [2009], ($32 \times 32 \times 3$, 50,000 images). The more sophisticated model uses the Eurosat Helber et al. [2019] dataset of satellite images ($64 \times 64 \times 3$, 24,000 images, excluding "SeaLake" category). All datasets can be be easily loaded through Tensorflow Abadi et al. [2015] and/or Keras Chollet et al. [2015].

Initial results from both MNIST datasets offer insight into how the model uses relative structure of the images. Experiments using CIFAR10 and the preliminary model showed a lack of generalization to three channel images. This prompted the design of a two-stage generator, operating on the Eurosat data, consisting of a cover image generator network and an embedding network as to further encourage the generation of realistic images by the model while maintaining a high-fidelity embedding.

4 Methods and Experiments

The first approach uses a generator, discriminator, decoder design to directly map messages to an image and back through the generator and decoder, while the discriminator enforces realism and low detectability of the message in generated images. Various sized networks were trained for each of the datasets using this method. See Figure 1 for one implementation.

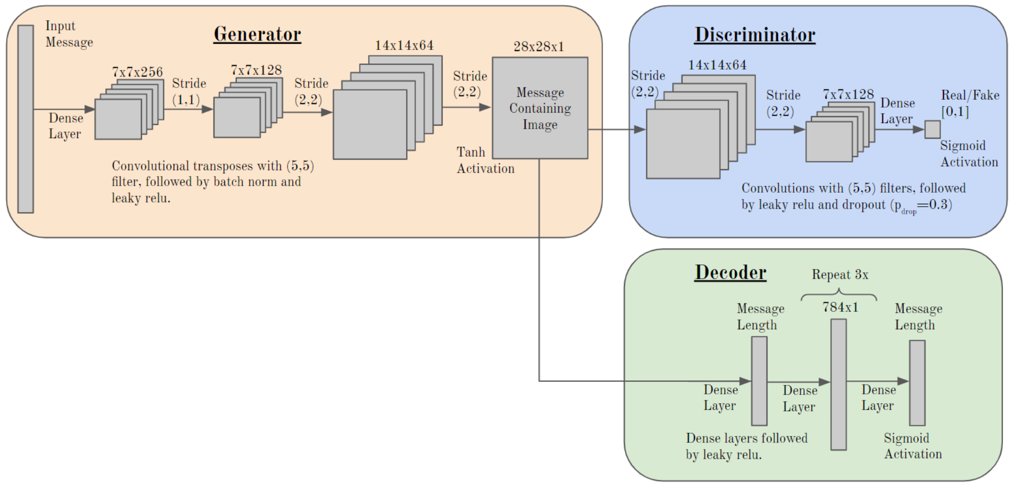

Figure 1: MNIST Digits DCGAN network architecture.

The second approach, dubbed the "Temporary Cover Model", Figure 2, was designed for the Eurosat dataset. This model splits the generator into a cover model and an embedding model. The cover model produces a novel cover image from the distribution of the data (akin to a standard generator in a GAN), and the embedding model encodes the message into this cover. The discriminator operates against real, cover, and embedded images, and the decoder is the same as previous models.

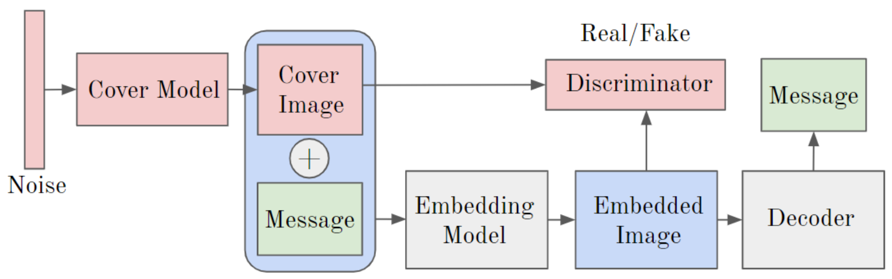

Figure 2: Eurosat DCGAN Outline for the "Temporary Cover Model". A version with implementation details is given in Appendix A in Figure 6

This eliminates issues with the complexity of mapping several thousand bits to an image in an invertible manner, which arose when training on CIFAR10 and Eurosat in the previous design. Note that this model is still embedding free, as the cover image does not exist outside of the model pipeline, as it is a novel image generated in a GAN-like manner and should not exist in the dataset.

4.1 Embedding Free Method

The model for MNIST Digits and Fashion can be seen in figure 1. The input message is a randomly generated array of 1's and 0's representing bits of the message to be encoded. The generator uses a dense layer and then a series of convolutional transposes with batch norm and leaky relu to scale the data up to 28x28, the size of the images. A discriminator takes real and generated images and uses convolutions to return a scalar value of probability that the input is real or fake. The decoder accepts generated images and uses dense layers to output the input bit message.

We have three loss functions, one for each network. The discriminator uses cross entropy loss with the real or fake label. The decoder uses cross entropy between the output message and the input message. The generator uses cross entropy with fooling the discriminator, typical to GANs, and added to that is a scalar multiple of the decoder loss. Thus the generator optimizes towards faithful message reconstruction by the decoder *and* fooling the discriminator.

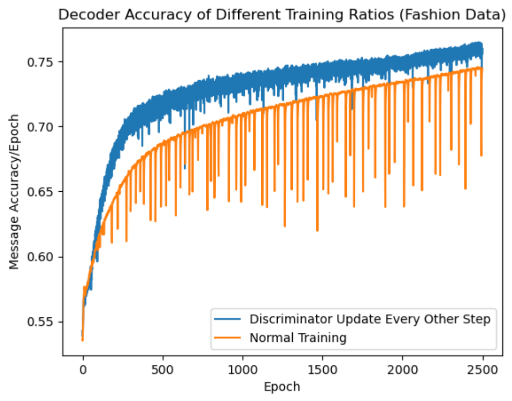

Figure 3: Accuracy of message reconstruction per epoch for the two training paradigms.

One trick that is used in literature to help with GAN training is to not train the discriminator every step. We implement this, where after epoch 40 the discriminator has its weights updated every other step. In these discriminator free steps, the generator and decoder will be updated, however we modify the generator loss to add 4 to 8 times the decoder loss. This emphasizes training towards an embedding, at the risk of sacrificing some image quality. Depending on which training epoch is last completed, results can be more realistic image-wise (normal epoch) or more accurate in embedding (generator/decoder only epoch). We see this improves accuracy more quickly over a training paradigm that updates all of the networks at each step, Figure 3.

Results from MNIST Fashion and CIFAR10 are in Figure 4, with fashion embedding 450 bits and CIFAR 1024 bits. Both saw higher accuracies in message reconstruction after the generator/decoder only training epoch with only moderate decreases in quality for fashion images. Note that since the fashion images are one channel, we can argue that the bpp should be scaled by three, as these pixels hold 1/3 the information of a color image.

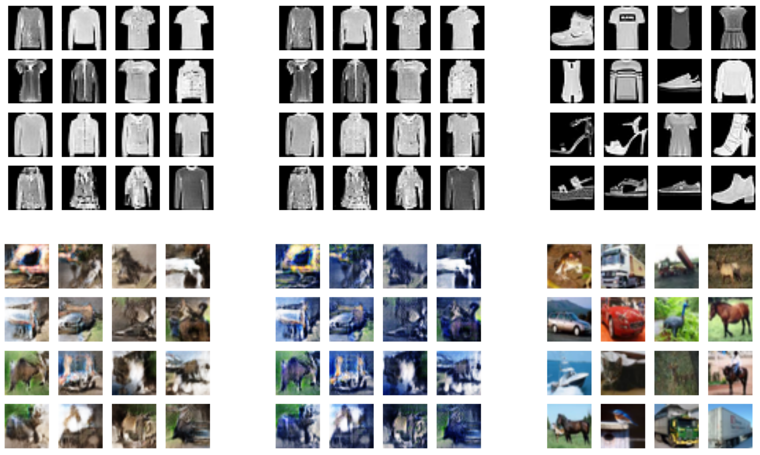

Figure 4: Left is normal epoch, center is generator/decoder only epoch, right is data.
Fashion accuracy: left 92.68%, center 94.47%, message size of 450 bits (0.57 bpp)
CIFAR10 Accuracy: left 98.91%, center 99.33%, message size of 1024 bits (1 bpp)

Extending this network architecture to color images (CIFAR10), allows for larger message payloads. However, the number of parameters when using dense layers for encoding becomes prohibitively large. Instead convolutions are used in the generator to break down the input message to a localized embedding space, then convolutional transposes are then applied to bring this embedding up to the correct image dimensions. This led to high accuracy embeddings with low visual quality, as seen in the CIFAR10 images in the bottom row of Figure 4. This indicates that the mapping from message $\rightarrow$ image $\rightarrow$ message through a localized encoding (due to convolutions), such that the image is realistic and the message is accurate, is perhaps more difficult over the space of color images than greyscale.

4.2 Temporary Cover Model

To overcome the lack of realism in the generated CIFAR10 images, a new model was designed that splits the generator into two separate networks, Figure 2. The first, denoted a cover model, is a typical DCGAN generator. Noise is transformed into an image that represents a point in the distribution of the dataset, but does not directly come from the dataset. We use this image as our "temporary cover"; embedding the message into this cover through a CNN that operates as an embedding network. If this temporary cover does not persist outside of the model pipeline then the method is still *technically* embedding free.

The discriminator and decoder follow similar design principles as the previous models, with slight modifications to increase network depth. The main difference is that the discriminator now accepts three images for its loss, cross entropy with the temporary cover, the embedded image, and real images from the dataset. The cover model and embedding model both use cross entropy with fooling the generator, however the embedding network adds in the decoder loss. Thus the temporary cover strives to match the dataset while the embedding network attempts to maintain this realism while incorporating the message. The decoder uses cross entropy loss to minimize message error. Implementation details of a model designed around the Eurosat dataset are given in Figure 6.

Note that we no longer implement the alternating training scheme as before, however looking at Figure 2, we see that the cover model and discriminator network are just a normal DCGAN (red components in the figure). Thus we can pre-train these two networks to develop realistic covers before we start training the embedding model and decoder. This provided the benefit of faster initial training and seemed to produce better results in the long run. It also opens up the possibilities of introducing transfer learning to the model design.

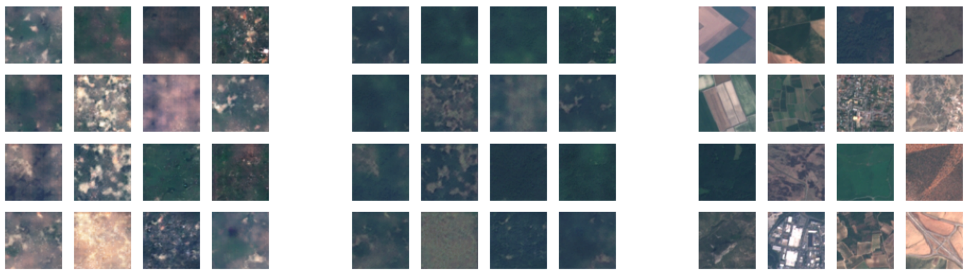

Figure 5: Left is cover, center is with embedding, right is data.
Eurosat dataset. Message accuracy (epoch 3750) 92.16%, message size of 8192 bits (2 bpp)

Results with Eurosat data are given in Figure 5. The cover model and discriminator were trained for the first 1,000 epochs and for the last 2,750 epochs all four networks were trained. We see relatively realistic images (barring mode collapse) with high fidelity embeddings for a lot of data, 8192 bits.

5 Results

To give quantitative results, we test the embedding against the statistical methods implemented in StegExpose. Using the default detection threshold we summarize our quantitative results in Table 1. Note the results on the first three datasets are with the first embedding free model and the Eurosat results are from the temporary cover model.

Dataset	bits/pixel	Decoder Accuracy	Epochs	Detection %	Real Detection %
MNIST Digits	0.19	98.05%	9000	0.11%	0.38%
MNIST Fashion	0.57	94.47%	9500	19.98%	23.47%
CIFAR10	1	99.33%	8000	85.89%	63.23%
Eurosat	2	92.16%	3750	0.62%	12.97%

Table 1: StegExpose results, percent of embedded and real images detected as message containing.

As for qualitative results, in the first model we saw extremely realistic images and accurate message results from Fashion MNIST as well as MNIST (Figure 7 in Appendix B if the reader is interested). For both of these datasets we saw a detection rate lower than the false positive rate for the real images. In CIFAR10 we saw this realism disappear, and the detection accuracy was quite high as well (85%).

The updated temporary cover model for the Eurosat data produced extremely realistic images, high message accuracy with a high bpp (2 bpp), and low detection rate (0.6%). However these realistic images seemed to only come from one of the categories in the dataset, green areas of plains and hills. We can see this mode collapse even more explicitly with the fashion results, as most of the outputs are shirts. This makes sense in the context of our loss functions: if there is more information typically contained one category of the data it will likely be easier to encode the message more accurately into these types of images.

6 Conclusion and Future Work

With the two model designs developed through this work we investigated the idea of embedding information into the process of generating an image rather than into an existing cover, providing heightened security in image steganography. However, a direct mapping becomes difficult when moving past one channel. This can be remedied by generating a realistic temporary cover, which should never exist outside of the model pipeline, to embed the information into. This improved the realism of generated color images and increased the capacity of embedded data. As well, this method almost entirely avoided statistical detection, making it a viable candidate for image steganography.

Future work should look at using transfer learning on pre-trained image generators to replace the cover model and hopefully speed up training and increase the realism of the output images. Extending the model to be similar to a conditional GAN, Mirza and Osindero [2014], might allow one to fight against the mode collapse seen here as well. Finally, investigation into where exactly the message is embedded in the Eurosat images could provide useful insight to improve embedding density.

References

Benedikt Boehm. Stegexpose-a tool for detecting lsb steganography. *arXiv preprint arXiv:1410.6656*, 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Shumeet Baluja. Hiding images in plain sight: Deep steganography. In *Advances in Neural Information Processing Systems*, pages 2069–2079, 2017.

Songtao Wu, Shenghua Zhong, and Yan Liu. Deep residual learning for image steganalysis. *Multimedia tools and applications*, 77(9):10437–10453, 2018.

Zihan Wang, Neng Gao, Xin Wang, Ji Xiang, and Guanqun Liu. Stnet: A style transformation network for deep image steganography. In *International Conference on Neural Information Processing*, pages 3–14. Springer, 2019.

Donghui Hu, Liang Wang, Wenjie Jiang, Shuli Zheng, and Bin Li. A novel image steganography method via deep convolutional generative adversarial networks. *IEEE Access*, 6:38303–38314, 2018.

Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. Steganogan: High capacity image steganography with gans. *arXiv preprint arXiv:1901.03892*, 2019.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

François Chollet et al. Keras. `https://keras.io`, 2015.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

A Eurosat DCGAN Implementation

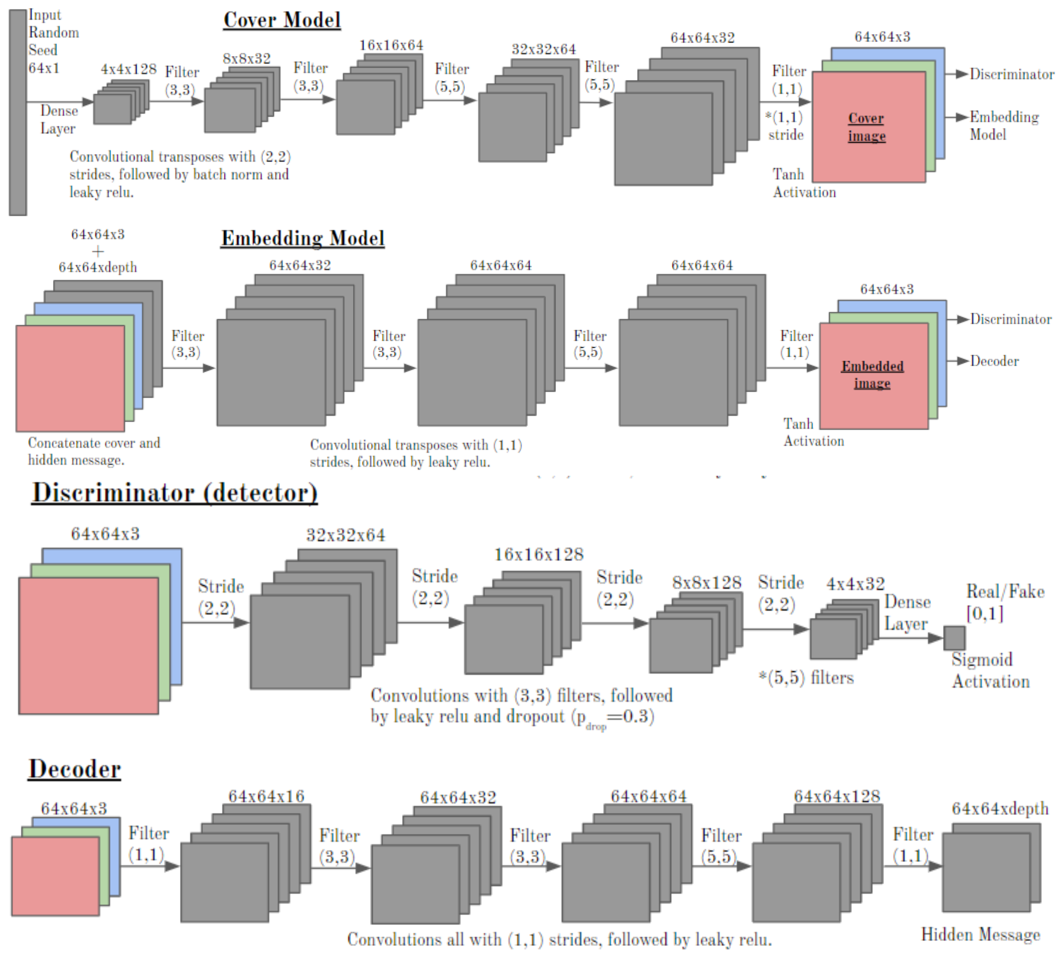

Figure 6: Implementation details for the Eurosat DCGAN. See Figure 5 for the context/general design.

B MNIST Digits Results

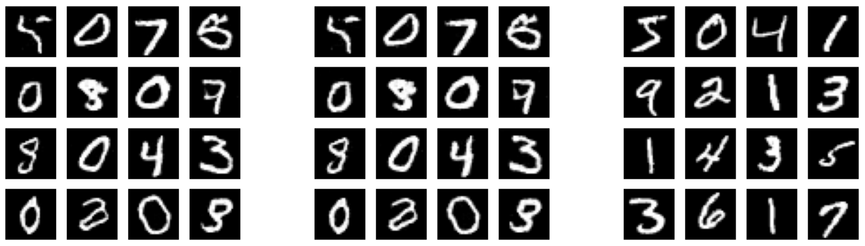

Figure 7: Digits accuracy: left 95.30%, center 98.05%, message size of 150 bits (0.19 bpp)