SteganoGAN Perturbation Testing

Mohammad Khan (msk295) **Alperen Komban** (ak2533) **Camilo Garcia** (cg665) **Aidan Adams** (aaa289)

https://github.com/Sayeem2004/steganoGAN

1 Introduction to SteganoGAN

Our project involves the task of steganography, which is the process of sending hidden messages within other media (images, in our case), with the goal of making it look like there is nothing hidden at all. Cryptography is the standard way of sending hidden messages, but it may suffer due to legal restrictions of certain countries, or it may invite attackers due to its detectable nature. Image steganography offers an alternative that avoids the above issues, and could be particularly useful in the case of medical or copyright data being stored easily in related images. Note that the traits that define a more successful steganography are undetectable, lossless (all data is recovered), and highe0[p9-0=] information density.

Our project is an extension of the work of Zhang et al. [2019]: "High Capacity Image Steganography with GANs", SteganoGAN for short. Traditional (non-machine learning) steganographic methods usually suffer from low information density to remain undetectable, while previous modern (deep learning) approaches suffer from other requirements like constant image sizes and noise. SteganoGAN uses a GAN-like CNN architecture of an encoder, decoder, and critic with an adversarial training regime to maintain being undetectable (usually associated with high-quality, realistic images) while still having high information density.

Furthermore, Zhang et al. invented a new metric called Reed-Solomon Bits Per Pixel (RS-BPP) for estimating the amount of data that can be stored in a given pixel using their network. This is necessary because deep learning approaches are not inherently lossless, some extra work (Reed Solomon encryption in this case) needs to be put in to reconstruct the original input, which translates to reduced data payload size. Using their SteganoGAN network, the authors achieve an RS-BPP of 4.4. Finally, they confirm that their SteganoGAN is mostly undetectable to various steganographic analysis tools as indicated with auROC score of 0.59. For classification problems, an auROC of 0.5 indicates random guessing, and an auROC of 1 indicates perfect guessing.

2 Narrowing the Scope

The most significant result from the SteganoGAN paper seems to be the high RS-BPP values despite also having SSIM scores close to 1 (i.e. original and encoded images are very similar). These results are summarized in Figure 1 below, which we will be recreating and comparing against. The original paper actually included a second section on this table that involved the COCO dataset by Lin et al. [2014], but because it has 330,000 images, it was unfeasible to use, and became a lower priority benchmark.

Dataset	**D**	**Accuracy**			**RS-BPP**			**PSNR**			**SSIM**		
		Basic	Resid.	Dense	Basic	Resid	Dense	Basic	Resid.	Dense	Basic	Resid.	Dense
Div2K	1	0.95	0.99	1.00	0.91	0.99	0.99	24.52	41.68	41.60	0.70	0.96	0.95
	2	0.91	0.98	0.99	1.65	1.92	1.96	24.62	38.25	39.62	0.67	0.90	0.92
	3	0.82	0.92	0.94	1.92	2.52	2.63	25.03	36.67	36.52	0.69	0.85	0.85
	4	0.75	0.82	0.82	1.98	2.52	2.53	24.45	37.86	37.49	0.69	0.88	0.88
	5	0.69	0.74	0.75	1.86	2.39	2.50	24.90	39.45	38.65	0.70	0.90	0.90
	6	0.67	0.69	0.70	2.04	2.32	2.44	24.72	39.53	38.94	0.70	0.91	0.90

Figure 1: Metrics from 18 models (3 encoders x 6 data depths) run on the Div2K dataset by Agustsson and Timofte [2017].

These results are significant because previously people thought that steganography had achieved a Pareto frontier, where any more payload size would force lower image quality. But, this paper shows this is not the case by 10x the data payload with high image quality. In this project, we first want to confirm that breaking past the previous Pareto frontier is possible (by recreating the above results), and second explore where the next frontier might be. To achieve these two goals we also did the following:

- First, we wanted to confirm that the RS-BPP statistic is actually useful. We were worried since RS-BPP is based on the mean accuracy of the network that it may be unrepresentative if the variance of the network is big. If this were the case, the paper might not have actually broken the Pareto frontier. So we calculated the variance of the network to confirm.
- Second, the paper makes a claim that Div2K had worse metrics than COCO because of structural differences in image content. We believe that this is a misrepresentation, and is instead due to the differences in image quality: Div2k being 4 megapixels, while COCO being 1/4 megapixels. To confirm our hypothesis (and to reduce training time), we used the 4x compressed Div2K dataset to train and test our model (4x compressed on each side leads to 1/4 megapixels).
- Finally, we wanted to do various perturbation testing on the model including: doubling the training epochs, trying out data depths up to 12, and switching the order of the LeakyReLU to be after BatchNorm. For the last choice, we noticed that in the SteganoGAN architecture, the LeakyReLU was placed before the BatchNorm. We thought that this would be a hindrance because the LeakyReLU's introduced nonlinearities might be reduced or reversed by the BatchNorm.

3 Experiment Methodology

The original paper splits up the network into three CNNs: an encoder, a decoder, and a critic. The encoder takes in an image and data bits (reshaped to be same size as image, but with channels = fixed data depth) and outputs the encoded image. The decoder takes in the encoded image and outputs the reconstruction of the original data bits. Finally, the critic takes in the encoded image and outputs a realism score (higher is better). The paper has three different types of encoders: basic, residual, dense and two different types of decoders: basic and dense based on the type of connections between CNN blocks. Similar to the paper, we tested across all three encoder type, only tested the dense decoder, and didn't vary the critic architecture. We also invoked the same two phase training process, where every epoch the encoder/decoder is trained first keeping the critic constant, and then vice versa.

We trained and validated on the 4x compressed bicubic Div2K dataset (800, 100 images respectively). For testing, we were not able to find the testing set on the Div2K website, so we instead used the 4x compressed unknown Div2k dataset (100 images) and the COCO 2017 validation set (100 images). There is the worry of overfitting to the testing set because the validation and first testing set is very similar, so we have to take that into account in our analysis. The data bits used in training, validation, and testing were generated randomly. We used the AdamW optimizer instead of the Adam optimizer as we thought more modern techniques would achieve better performance. For similar reasons, we used the Pillow image normalization functionality (range $[0, 1]$)instead of the manual normalization of $I/127.5 - 1$ (range $[-1, 1]$), where I represents the RGB values from $[0, 255]$. Exactly as stated in the paper, we trained for 32 epochs with a learning rate of $1e^{-4}$, clipping the gradient norm to 0.25 and critic weights to $[-0.1, 0.1]$.

To evaluate the performance of our SteganoGAN implementation and compare it with the paper, we implemented the four metrics mentioned in the paper: Accuracy, RS-BPP, PSNR, and SSIM. The first two assess the capacity of our SteganoGAN to hide information and the next two assess the quality of the resulting steganographic images. Accuracy refers to the bitwise accuracy of the reconstructed input versus the original input. The RS-BPP metric as described in the original paper measures the effective payload capacity using Reed-Solomon codes. This calculation models the error correcting behavior, where given a bit inaccuracy of p, the ratio of real data that can be reliably transmitted is $1 - 2p$. By multiplying this ratio with the data depth, we obtain an effective bits-per-pixel measurement.

To measure image distortion, we have implemented PSNR which calculates the ratio between the maximum possible power of a signal and the power of noise. Higher PSNR values indicate better image quality and less distortion in the steganographic image. Note that since our data normalizations methods are different than the papers the PSNR scales differently. Lastly, we have implemented SSIM to evaluate the perceptual similarity between the cover and steganographic images. SSIM can capture the perceptual changes in structural information like luminance and contrast, providing a complementary measure to PSNR, that the SteganoGAN paper views as more accurate. Finally, to measure how well our model can avoid steganographic analysis tools, we use the traditional analysis tool StegExpose by Boehm [2014]. The paper actually tested other Neural Network based tools, but due to time constraints we could not do the same.

4 Results & Analysis

Overall, the training process was very smooth with no discrepancies from our methodologies section. The only real issues that we ran into where mistakes in our implementation code and the long training times. We solved the latter problem by borrowing a RTX 5070 Ti GPU causing our time per epoch to go from 10 minutes to 15 seconds. This allowed us to experiment more, and also retrain when errors in our implementation occurred. In the testing process, we faced difficulty in encoding messages longer than 50 characters. However, we think this is a bug with our Reed-Solomon encoding code not the network, and it shouldn't affect our results because we used random data bits in both training and testing instead of actual messages. Our recreation of the table in Figure 1, is shown below in Figure 2 along with our results of testing traditional steganographic analysis tools.

Dataset	D	Accuracy			RS-BPP			PSNR			SSIM		
		Basic	Resid.	Dense	Basic	Resid.	Dense	Basic	Resid.	Dense	Basic	Resid.	Dense
Div2K pretrained	1	1.00	1.00	1.00	1.00	1.00	1.00	24.42	41.44	43.06	1.00	1.00	1.00
	2	0.99	1.00	1.00	1.97	1.99	2.00	27.23	38.55	37.34	1.00	1.00	1.00
	3	0.98	0.99	1.00	2.87	2.95	2.97	20.33	33.64	34.38	0.99	1.00	1.00
	4	0.85	0.86	0.95	2.78	2.88	3.60	18.29	34.64	33.64	0.99	1.00	1.00
	5	0.78	0.78	0.84	2.79	2.84	3.45	24.19	36.21	34.30	1.00	1.00	1.00
	6	0.73	0.73	0.74	2.77	2.82	2.84	28.62	35.99	35.13	1.00	1.00	1.00
COCO	1	1.00	1.00	1.00	0.99	1.00	1.00	23.78	39.45	40.70	1.00	1.00	1.00
	2	0.99	0.99	1.00	1.96	1.97	1.99	25.80	37.01	35.78	1.00	1.00	1.00
	3	0.97	0.98	0.99	2.80	2.88	2.95	19.75	31.91	33.05	0.99	1.00	1.00
	4	0.84	0.85	0.93	2.69	2.77	3.45	18.08	33.02	32.46	0.99	1.00	1.00
	5	0.77	0.77	0.83	2.67	2.70	3.29	23.25	34.75	32.78	1.00	1.00	1.00
	6	0.72	0.72	0.73	2.65	2.69	2.71	27.01	34.52	33.65	1.00	1.00	1.00

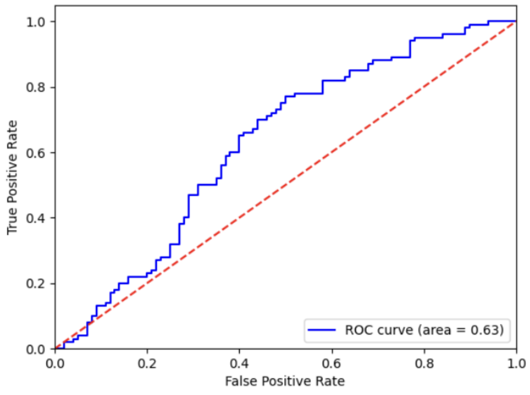
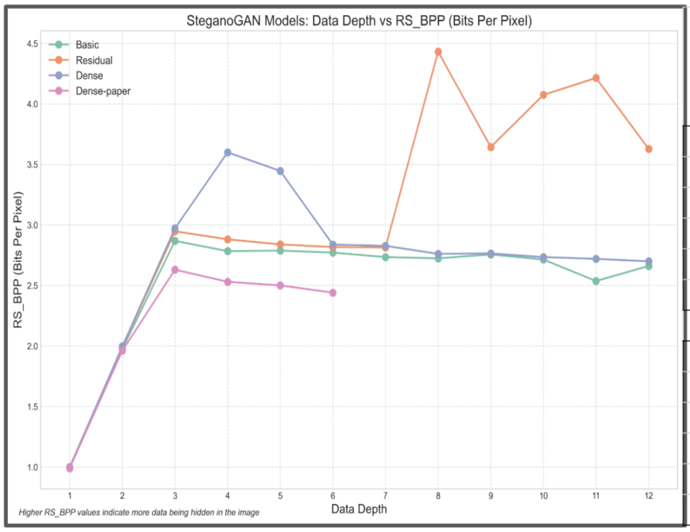
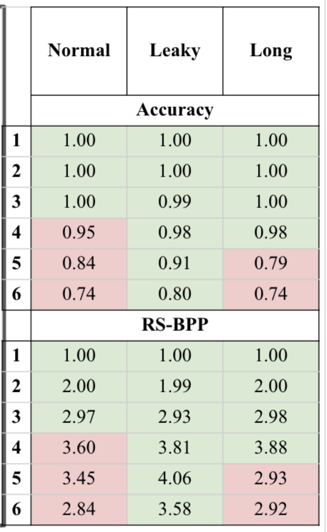


Figure 2: (a) Our recreation of Figure 1 (green indicates better than original paper) (b) Receiver Operating Curve of StegExpose

As seen from the majority green in the accuracy and RS-BPP sections, we have successfully obtained the high data density demonstrated in the papers. We can notice that the COCO dataset has a lot more red, and this is likely because our model wasn't trained on COCO data, thus being more unfamiliar to it. The vast majority of our PSNR scores are lower than the original paper (but they are comparable in magnitude), and this is likely due to the normalization differences mentioned above. The SSIM scores are all higher than the paper's scores and most of them reach the upper bound of 1. This is suspicious for the small amount of training we gave, and thus we don't think the SSIM scores are trustworthy. We think that this discrepancy in SSIM scores might also be caused by the data normalization differences, but we didn't have the time to check this hypothesis out further.

Now for the ROC curve, the original paper evaluated their model's steganographic images using StegExpose, reporting an auROC of 0.59. Our re-implementation achieved a slightly higher auROC of 0.63, suggesting only a minor difference in detectability. This statistic and the fact that we couldn't detect any changes visually indicate that we have surpassed the previous Pareto frontier, successfully replicating the results from the paper. One further thing to note is that although our compressed Div2K RS-BPP results are significantly higher than the original paper's, it does not reach the level of 4.4 that the original paper achieved on COCO. This could indicate that the paper's hypothesis of structural difference in image content has some merit, and that we can't disprove it. Now we can move on to the results of our perturbation testing shown in Figure 3 below.





	Normal	Leaky	Long
	Accuracy		
1	1.00	1.00	1.00
2	1.00	1.00	1.00
3	1.00	0.99	1.00
4	0.95	0.98	0.98
5	0.84	0.91	0.79
6	0.74	0.80	0.74
	RS-BPP		
1	1.00	1.00	1.00
2	2.00	1.99	2.00
3	2.97	2.93	2.98
4	3.60	3.81	3.88
5	3.45	4.06	2.93
6	2.84	3.58	2.92

Figure 3: (a) RS-BPP vs Data Depth for various encoders (b) RS-BPP for switching leaky and longer training time perturbations.

Looking at the purple line, we can see that the original paper observes a steady decline after a data depth of 3 on the Div2K dataset, which might be why they don't experiment past a data depth of 6. However, when we explored higher depths, we actually achieved an RS-BPP of 4.4 at a data depth of 8, 50% greater than what the original paper achieved. It is also important to note that this was done with a Residual encoder and not the Dense encoder that the paper claimed did the best. Similarly, we can notice that our leaky model (switching the order of LeakyReLU and BatchNorm) also achieved an RS-BPP of 3.58 at a data depth of 6, a 25% increase over what the normal architecture achieved. These massive increases in RS-BPP without too much of a decrease in image quality, likely indicate that there is a further Pareto frontier to find.

Finally, for the normal dense architecture with data depth of 6, we found a mean accuracy of approximately 0.725, with a standard deviation of 0.011. This means we can only guarantee an accuracy of 0.70 with 95% confidence. If we were to use this as our baseline instead of the mean, the RS-BPP would go from 2.7 to 2.436. This is almost a 10% decrease which is significant, but even with this decrease SteganoGAN goes way beyond the older Pareto frontier from traditional steganographic methods.

5 Reflections

We successfully recreated a GAN-based model capable of reliably hiding messages within images while mostly avoiding detection. Our implementation closely matched—and in some cases exceeded—the original SteganoGAN model's performance on several metrics. Additionally, our experiments with perturbations revealed significant trends that might be useful for further exploration to find the next Pareto frontier. However, all our results have to be taken with a grain of salt because of the many sources of noise including: 4x compressed dataset, no independent testing set, different metrics due to data normalization, etc.

The biggest lesson we learned throughout this project is the importance of computational resources, without a fast GPU, we might not have even gotten 1/4th the results we found. We also believe that with more computational resources and training on a broader range of less-compressed datasets, our results could be further validated and potentially extended. Our results imply that a GAN that is able to make steganographic images with dense data nearly indistinguishable to even other models is not an impossibility.

6 References

Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

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

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014.

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