Neural Steganalysis with Spatial Rich Models for Image Steganography Detection

Jonathan Lwowski, Isaac Corley, Justin Hoffman
Booz Allen Hamilton
3133 General Hudnell Drive, San Antonio, Texas 78226
lwowski_jonathan@bah.com, corley_isaac@bah.com, hoffman_justin@bah.com

Abstract

Digital image steganalysis is the process of detecting if an image contains concealed data embedded within its pixel space inserted via a steganography algorithm. The detection of these images is highly motivated by Advanced Persistent Threat (APT) groups, such as APT37 Reaper, commonly utilizing these techniques to transmit malicious shellcode to perform further post-exploitation activity on a compromised host. Performing detection has become increasingly difficult due to modern steganography algorithms advancing at a greater rate than the steganalysis techniques designed to combat them. The task of detection is challenging due to modern steganography techniques that embed messages into images with only minor modifications to the original content which varies from image to image. In this paper, we pipeline Spatial Rich Models (SRM) feature extraction, Principal Component Analysis (PCA), and Deep Neural Networks (DNNs) to perform image steganalysis. Our proposed model, Neural Spatial Rich Models (NSRM) is an ensemble of DNN classifiers trained to detect 4 different state-of-the-art steganography algorithms at 5 different embedding rates, allowing for an end-to-end model which can be more easily deployed at scale. Additionally our results show our proposed model outperforms other current state-of-the-art neural network based image steganalysis techniques. Lastly, we provide an analysis of the current academic steganalysis benchmark dataset, BOSSBase, as well as performance of detection of steganography in various file formats with the hope of moving image steganalysis algorithms towards the point they can be utilized in actual industry applications.

1. Introduction

Steganography is the method of sending hidden data such that only the sender and the intended recipient suspect the existence of the hidden message [5]. More specifically, digital image steganography is the usage of image files as the host for hidden steganographic messages. Advanced Persistent Threat (APT) groups, such as APT37 Reaper [17], commonly utilize image steganography as a means to transmit additional shellcode or scripts to an already compromised system. Once the image is received, the adversary then extracts the malicious payload and executes it to perform further post-exploitation activity on the target machine. Most recently, adversaries have used anonymous Twitter accounts as a command and control (C2) mechanism to download images containing concealed malware to update their capabilities on the victim machines [26]. Due to the highly undetectable nature of the current state of the art image steganography algorithms, adversaries are able to evade defensive tools such as Intrusion Detection Systems (IDS) and/or Antivirus (AV) software which utilize heuristic and rule-based techniques for detection of malicious activity.

To prevent the passing of hidden data, the detection of steganography using analytical methods, Steganalysis, has been studied for many years. Existing steganalysis methods typically perform well when detecting traditional steganography algorithms, such as Least Significant Bit (LSB) steganography [2]. However, many modern steganography algorithms, such as Highly Undetectable steGO (HUGO) [19], have been shown to evade these same steganalysis techniques. With the emergence of machine learning, many modern steganalysis algorithms are now incorporating deep neural networks as the classification backbone to detect modern steganography algorithms.

Traditional machine learning methods for steganalysis typically consist of ensemble classifiers of algorithms such as Support Vector Machines and Random Forests [15], [14], [8] which take input features extracted using methods such as Spatial Rich Models (SRM) [8], maxSRM [6], CFA-aware CRM [9], and many others. With the resurgence of neural networks, a slew of steganalysis techniques utilizing Deep Neural Networks (DNN) on the above mentioned

Patent Applied for in the United States

feature extractors [25] and Convolutional Neural Networks (CNN) on the raw images [3], [23], [20], [24] have been developed to attempt to solve identification of steganography at the pixel level.

In this paper, we propose an ensemble model which utilizes SRM [8] for feature extraction, Principal Component Analysis (PCA) [18], and DNNs to perform JPG image steganalysis across various state of the art steganography algorithms and embedding rates. To the best of our knowledge, our proposed model, Neural Spatial Rich Models (NSRM), outperforms the currently existing machine learning based JPG image steganalysis models which are trained on the same dataset.

The contents of this paper are organized as follows. Section 2 details the dataset used to train NSRM. Our proposed NSRM will be discussed in Section 3, followed by an analysis of the results in Section 4. The implications of trying to deploy this model in the real world are presented in Section 5. Finally, the conclusions and future works will be discussed in Section 6.

2. Dataset

2.1. Dataset Creation

To train our proposed NSRM, we used the BOSSBase dataset [1] because it is used as a benchmark dataset for many machine learning based steganalysis papers. The BOSSBase dataset contains 10,000 grayscale images of size 512x512 and of the portable gray map (PGM) format. An example of one of the BOSSBase images can be seen in Figure 1. The images were taken with seven different cameras. The images were then converted to JPG format with a quality factor of 95%, to be able to compare to other steganalysis techniques which are typically modeled on JPG images, and resized to 256x256 dimensions.

As seen in Figure 2, 4 different steganography algorithms are used to encode the 10,000 images. These algorithms (HUGO [19], HILL [16], S-UNIWARD [11], and WOW [10]) are considered to be some of the state of the art steganography algorithms are difficult to detect by modern steganalysis algorithms. These algorithms are open source and made available by the Digital Data Embedding Laboratory of Binghamton University [7]. For each of these steganography algorithms, 5 different embedding rates were used. The embedding rates used were 10%, 20%, 30%, 40%, and 50%, with 10% being the most difficult to detect, and 50% being the easiest to detect. This process created a total of 210,000 images consisting of 200,000 steganographic images, and 10,000 cover images.

2.2. Dataset Analysis

Before performing steganalysis on this dataset, an exploratory data analysis was performed to assess the difficulty of detection of the steganographic content embedded into the pixel space of the images. The first analysis that was performed was to compare the cover images to their steganographic counterparts. For example, the cover image in Figure 1 was subtracted from its steganographic counterpart at multiple embedding rates of 10% and 50%. This reveals the minor pixel differences that the steganography embedding has caused. As seen in Figure 3, the pixel differences for both 10% and 50% embedding rates are visually significant. Additionally, upon closer inspection, the steganographic content appears to have a square-like, checkerboard pattern. These patterns notably appear in all steganographic JPG images we analyzed, but do not appear in other non-JPG image file formats. This is potentially due to JPG being a lossy form of compression that removes high frequency noise components within the image.

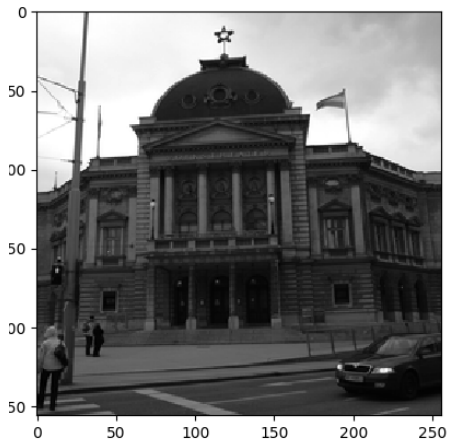


Figure 1: Example image from the BOSSBase dataset

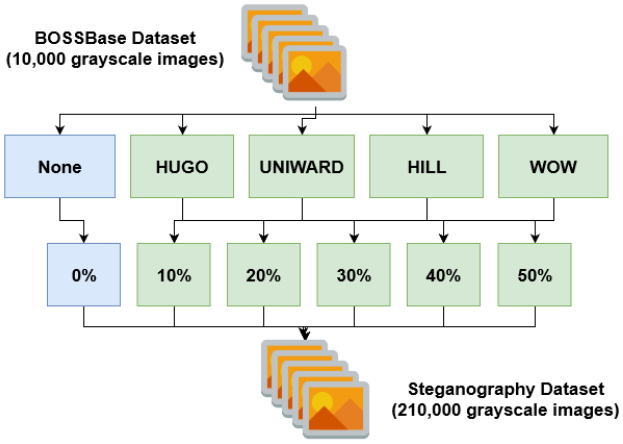
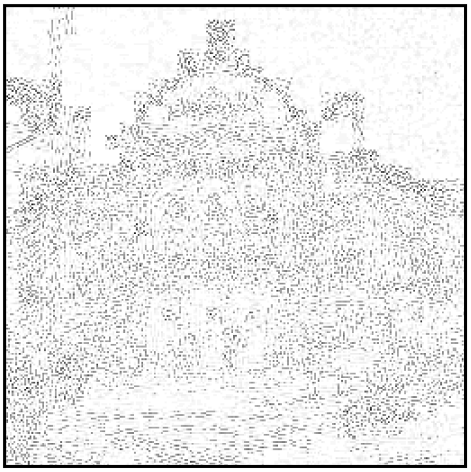


Figure 2: Steganography dataset creation process

This pattern likely serves as the primary features which a machine learning model could learn to more easily detect whether an image contains steganography.

(a) 10% Encoding (JPG)



(b) 50% Encoding (JPG)

Figure 3: Pixel differences between a cover image and its corresponding steganographic image at 10% and 50% embedding rates

The next analysis that was performed, was to subtract the 10% embedded images from their corresponding 50% embedded images. This analysis was performed to contrast how the steganographic content of the images vary with different embedding rates. As seen in the example in Figure 4, there was a significant visual difference between the images. This analysis shows that increasing the embedding rate makes the checkerboard pattern even more apparent within the image and it can be inferred that steganalysis of images encoded with lower embedding rates make detection of steganographic content more difficult. Additionally, in Figure 5, the displays a closer look at the checkerboard pattern in Figure 4.

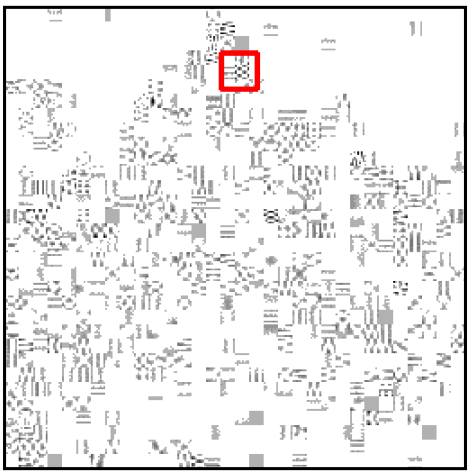


Figure 4: Pixel differences between a 10% embedded image and its corresponding 50% embedded JPG image

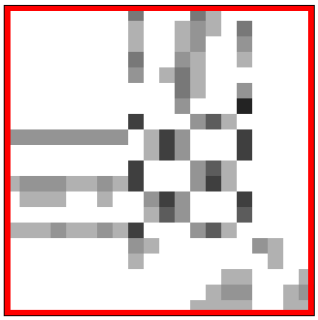


Figure 5: Checkerboard pattern revealed in the red square in Figure 4

3. Neural Spatial Rich Models (NSRM)

To detect steganographic images, our proposed NSRM utilizes a pipeline of Spatial Rich Models (SRM), Principal Component Analysis (PCA), and Deep Neural Networks (DNN) to detect each of the 4 steganography algorithms. The four models were then ensembled together by feeding the outputs of each individual model to an additional DNN model which results in a single model that detects all of the steganography algorithms used in our experiments.

3.1. Spatial Rich Models

Spatial Rich Models [8] was used as a preprocessing feature extractor because it uses the statistics of neighboring

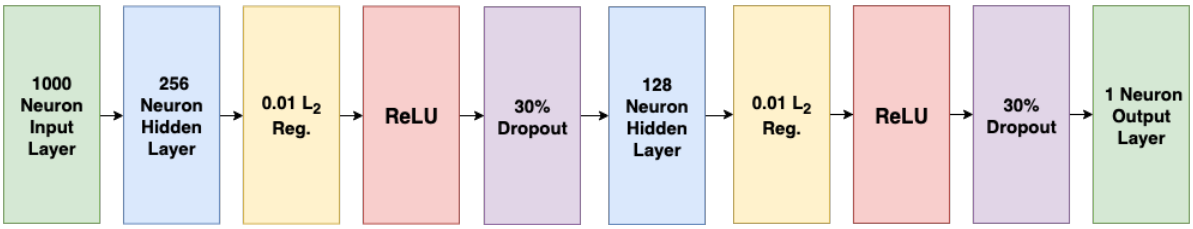


Figure 6: Individual Steganographic Detection DNN Model Architecture

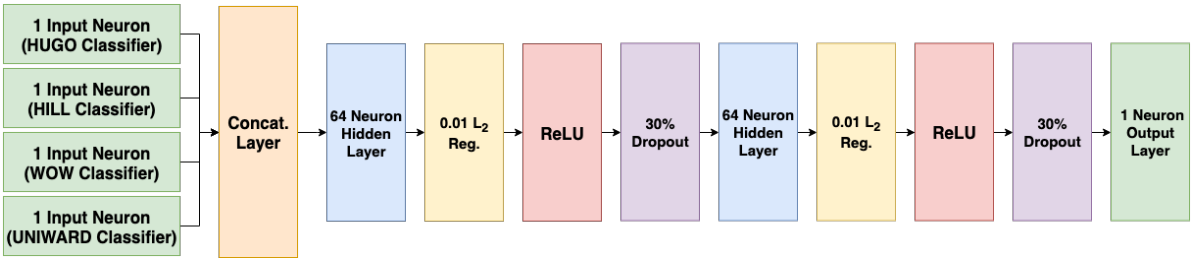


Figure 7: Multi-Steganographic Detection Ensemble DNN Model Architecture

noise residuals. These noise residuals capture the dependency changes caused by the steganographic embeddings. Since noise residuals are the high frequency components of the image, SRM is an effective feature extractor for detecting steganography [8]. The SRM begins by first computing the residuals (R) as follows:

$$R_{i,j} = \hat{X}_{i,j}(N_{i,j}) - X_{i,j} \tag{1}$$

where $N_{i,j}$ is the local neighborhood of pixel $X_{i,j}$, and $\hat{X}_{i,j}(N_{i,j})$ is the predictor for said pixel. The SRM then quantizes and truncates these residuals as follows:

$$R_{i,j} \leftarrow trunc_T\left(round\left(\frac{R_{i,j}}{q}\right)\right) \tag{2}$$

where $q > 0$ is the quantization step, and T is the truncation threshold. SRM produces an output vector with 37,561 residuals, which are then used as features for the classification models. Changes in the cover images due to the embedded payload will violate some of the local structures of the residuals. Along with the changes in the local residual, the neighboring residuals in the horizontal and vertical directions capture local dependency changes [8]. These properties of SRM make it an ideal feature extractor for steganalysis. Another particularly important property of SRM is that it computes a vector of residuals of the same 37,561 dimensions for images of arbitrary sizes. Therefore, any size image can be used as the input into our proposed model.

3.2. Individual Steganography Algorithm Classification

To make the learning easier for NSRM, we create a submodel to detect each of the different steganography algorithms individually. All of the individual algorithm classifiers use the same model, but are trained using data produced using the individual steganographic algorithm. The individual algorithm classification model, seen in Figure 6, begins by performing the SRM feature extraction which produces 37,561 input features. To assist with model convergence, each of the input features are scaled using standard normalization to have zero mean and unit variance. After scaling the SRM features, Principal Component Analysis (PCA) [18] was used to reduce the input features from 37,561 to 1000 principal components. This was done to avoid the "Curse of Dimensionality" typical of high dimensional data. Reducing the dimensionality has the additional benefit of reducing model training and inference time. Finally, the 1000 principal components were used as input into a DNN. The DNN has an input layer of 1000 neurons, because the output size of the PCA is 1000 principal components. Next, the DNN has a hidden layer of 256 neurons, l_2 regularization, Rectified Linear Unit (ReLU) activation [12], and 30% dropout [22]. The last hidden layer has 128 neurons, l_2 regularization, ReLU activation, and 30% dropout, as well. The output layer has 1 neuron with a sigmoid activation, which is treated as the probability the image contains steganography for the given algorithm.

3.3. Ensemble Classifier

After training each of the individual algorithm DNN classifiers, we ensemble each model together to create the NSRM, as seen in Figure 8. To ensemble the models together, each of the individual model weights are frozen. The same SRM, standard scaling, and PCA process is performed on the input images. Next, the 1000 components from the

PCA is fed into each of the individual DNNs. The output of each DNN is then fed into another DNN, which is used to ensemble each of the models together. The ensemble DNN model, seen in Figure 7, has four inputs. Each input contains one neuron, which is the probability of the image having steganography for each of the individual algorithms. Following the input layer is a concatenation layer, which combines all of the inputs into a single layer. The ensemble DNN has two hidden layers, each with 64 neurons, l_2 regularization, ReLU activations, and 30% dropout. The output layer has one neuron which represents the probability that the image contains any steganography from any of the 4 steganographic algorithms.

4. Results

4.1. Experimental Results

During experiments, the BOSSBase dataset was split into training and testing sets. The training set contained 75% of the images, and the testing set contained the remaining 25%. The NSRM was then trained using cover image and steganographic image pairs. This means for every steganographic image the model was trained on, the model also saw the corresponding cover image. This was done to balance the dataset since it contained only 10,000 cover images, and 200,000 steganographic images. Each individual algorithm classifier was trained for 25 epochs using the Adam optimizer [13] with a learning rate of 0.001. We noted that each algorithm required less than 25 epochs to converge. After training the individual model classifiers, all of the models had obtained both a training and testing accuracy above 99%. The testing accuracy for the individual models are provided in Table 1.

Table 1: Testing Accuracy for the Single Algorithm Classifiers

	HUGO	HILL	WOW	S-UNIWARD
Cover	1.0000	0.9992	0.9973	0.9976
10%	0.9993	0.9970	0.9913	0.9996
20%	0.9993	0.9976	0.9916	0.9996
30%	0.9993	0.9977	0.9903	0.9996
40%	0.9993	0.9975	0.9913	0.9997
50%	0.9993	0.9973	0.9923	0.9993

Once the individual algorithm classifiers have been trained, their weights are then frozen and pipelined into the NSRM ensemble classifier. The NSRM was trained using the same method above, except it was only trained for 5 epochs. The model only needed 5 epochs to converge since the individual models had already learned to produce adequate results, the ensemble just needed to learn how to combine their results together. After training the NSRM ensemble classifier, both the training and testing accuracy was above 99.5% for all 4 steganography algorithms on the JPG images in the BOSSBase dataset. More discussion is provided in Section 5 about the real world implications of operationalizing JPG steganalysis models. The testing accuracy for each of the algorithms is provided in Table 2.

Table 2: Testing Results for the NSRM Ensemble Classifier

	HUGO	HILL	WOW	S-UNIWARD
Cover	0.9997	0.9997	0.9997	0.9997
10%	0.9956	0.9956	0.9950	0.9956
20%	0.9960	0.9956	0.9950	0.9959
30%	0.9960	0.9956	0.9950	0.9995
40%	0.9956	0.9956	0.9960	0.9996
50%	0.9956	0.9956	0.9957	0.9996
Precision	0.9958	0.9957	0.9957	0.9959
Recall	0.9997	0.9997	0.9997	0.9997

Along with analyzing the training and testing accuracy, the Receiver Operating Characteristic (ROC) curves were also computed. As seen in Figure 9, the true positive rate (TPR) is very high, false positive rate (FPR) is very low, and the area under the curves are both very close to one.

Our results outperform other state of the art steganalysis models which were also trained on BOSSBase dataset, such as Gaussian Neuron Convolutional Neural Network (GNCNN) [20] and the CNN model developed by Xu et. al [25]. The test set accuracy for steganographic images embedded using the S-UNIWARD algorithm with a 0.4 embedding rate for the GNCNN and the CNN proposed by Xu et. al was 69.1% and 74.19%, respectively. The test set accuracy for steganographic images embedded using the S-UNIWARD algorithm with a 0.1 embedding rate for the GNCNN and the CNN proposed by Xu et. al was 54.08% and 56.08%, respectively. The test set accuracy for steganographic images embedded using the HILL algorithm for the CNN proposed by Xu et. al for an embedding rate of 0.4 and 0.1 was 73.54% and 56.01%, respectively. These models were not tested on the HUGO and WOW steganography algorithms, so comparisons to our results could not be performed. Along with the GNCNN and the model produced by Xu et. al., there are several other machine learning based steganalysis models [15], [3], however these models mix the BOSSBase dataset with other datasets, namely the BOWS2 [4] and CAMERA datasets, making comparisons not possible.

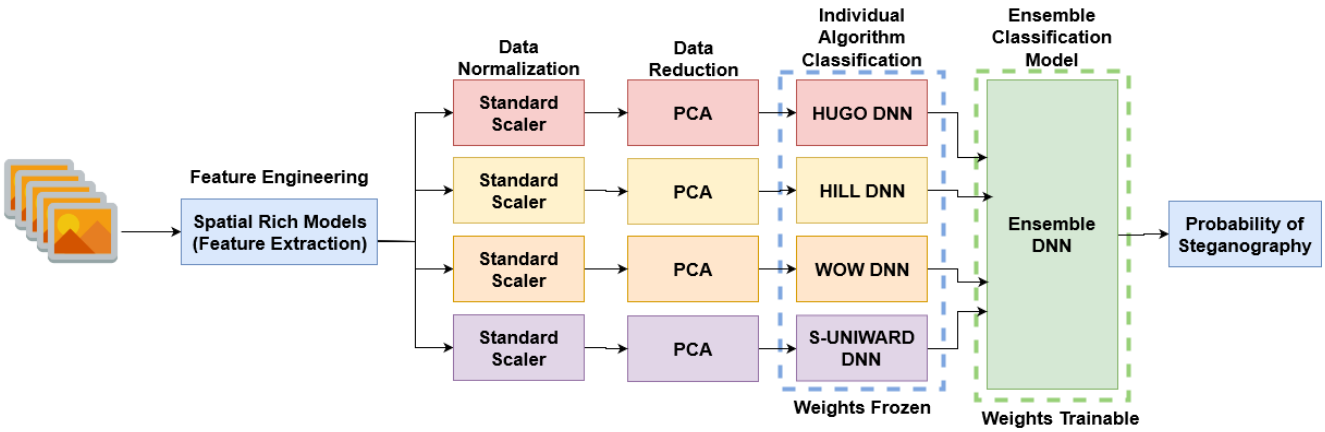


Figure 8: Neural Spatial Rich Model Architecture

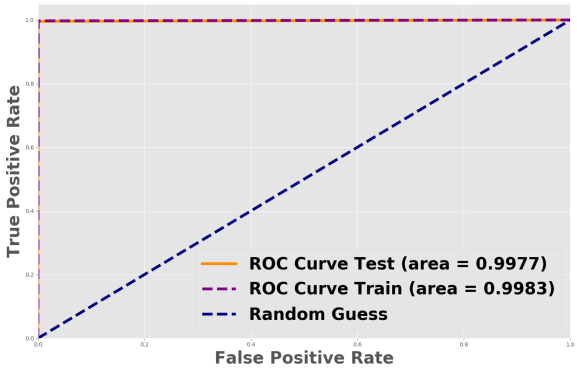


Figure 9: ROC curve for train and testing sets

5. Real World Implications

Although the NSRM model achieved greater than 99.5% accuracy on both the train and test sets, this does not indicate that the current state of the art steganalysis models are able to currently be deployed in a real world environment at scale. In the following sections we provide a detailed analysis of the difficulty of deploying steganalysis models which are able to detect steganography in a variety of image formats, contents, sizes, and color spaces.

5.1. Image Format Implications

To the best of our knowledge, the NSRM model and majority of other machine learning based steganalysis models are all trained and tested on images of the JPG format, which indicates that adversaries could simply use image formats other than JPG to evade detection. The reason the NSRM model was trained on JPG images was to compare the performance of our NSRM model to other state of the art models.

5.1.1 PGM Image Format Results

To analyze how the NSRM model performs on non-JPG image formats, the NSRM individual models were re-trained using the PGM image format which is the original raw image format of the BOSSBase dataset. The individual classifiers were trained on the same 210,000 images with the same training process as stated previously. As seen in Table 3, the individual models perform notably worse on the PGM image format. Interestingly, the model still obtained 99.9% accuracy detecting the HUGO steganography algorithm for all embedding rates, and can detect some of the steganography of other algorithms if the embedding rate is greater than approximately 30%. This results in a model which is not very useful to deploy in a real world environment to detect various types of images formats and confirms that detecting non-JPG image steganography is a harder problem than detecting JPG images. The NSRM ensemble model was not trained due to the poor performance of the individual steganalysis models.

Table 3: Testing Accuracy for the Individual Algorithm Classifiers on PGM Images

	HUGO	HILL	WOW	S-UNIWARD
Cover	.9999	.4696	.6531	.7681
10%	.9999	.5628	.3901	.2825
20%	.9999	.6400	.4923	.3965
30%	.9999	.7436	.5930	.5132
40%	.9999	.8236	.6631	.6272
50%	.9999	.8796	.7119	.6523

5.1.2 PGM versus JPG Steganalysis

As previously shown, JPG steganalysis is a much easier problem to solve due to the inherent nature of how JPG compression works. JPG compression is a lossy form of image compression which effectively acts as a low-pass filter by removing the high frequency content within an image. However, this process is reasonable for compression purposes due to high frequency components of images only containing approximately 5% of the image information [21]. As seen in Figure 10, the greater the amount of compression, or the lower the quality factor, the more pixelated and less detailed the image appears.

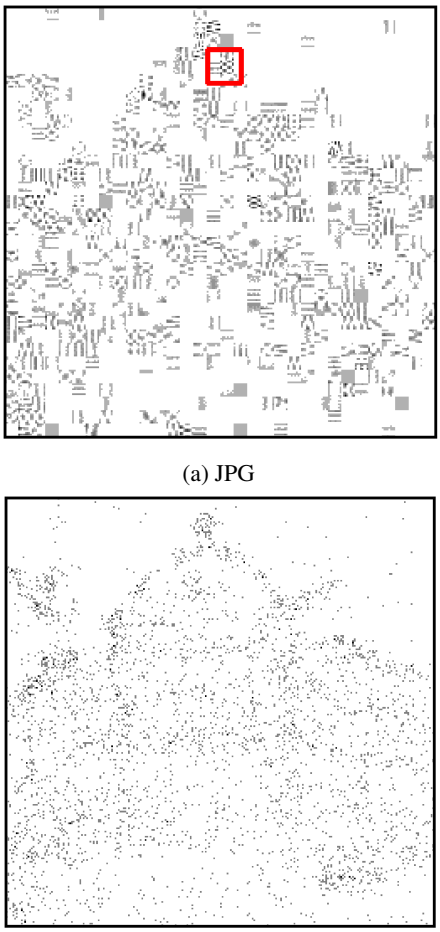
(a) Original Image (b) 10:1 JPG Comp. (c) 45:1 JPG Comp.

Figure 10: Examples of Different Amounts of JPG Compression [21]

Since other non-lossy file formats do not perform this smoothing, non-JPG steganalysis becomes increasingly difficult to detect in comparison to JPG steganalysis because the steganographic algorithms embed the hidden payload as high frequency noise within the other high frequency content. This makes it difficult for steganalysis methods to distinguish the embedded payload from other natural high frequency noise within the image. Due to non-JPG images containing high frequency content, the checkerboard like patterns embedded by the steganographic algorithms, previously discussed in Section 2.2, are not noticeable, whereas for JPG images it is visually apparent. A comparison of the cover and steganographic counterpart subtraction output for an example PGM and JPG images from the BOSSBase dataset is provided in Figure 11.

5.2. Image Size Implications

Another potential problem when implementing the NSRM and other machine learning steganalysis models is that the BossBase dataset only contains images that are all the same size. Even though the SRM feature extractor can compute features for any size input, this does not necessarily mean the model will have high accuracy on any arbitrarily sized image. Therefore the training dataset of a model must contain images of various dimensions. This will allow our NSRM model to be more generalized and be size invariant.



(a) JPG

(b) PGM

Figure 11: Comparison of Subtraction Cover from Steganographic Counterpart for PGM and JPG

5.3. Image Color Space Implications

Along with the image size implications, the dataset is only composed of grayscale images. This means steganalysis models trained on the BOSSBase dataset will only be capable of detecting steganographic grayscale images. In the real world, an adversary could evade these models simply using an image of another color space, such as RGB. Therefore, an optimal steganalysis dataset must contain images of a variety of color spaces.

5.4. Other Dataset Implications

Along with the problems listed above, the dataset also suffers from a couple other notable problems. One such problem is that the image content is primarily the same. Most of the image content is very similar with the majority of them being landscape or travel photos. Ideally the dataset would have a large variety of image content. Another potential problem is that the dataset was only captured using ten

different cameras. Every camera has its own fingerprint that it places on each image. This fingerprint could potentially be learned by a steganalysis model, and could be one of the features learned to distinguish cover images from steganographic images. Ideally the dataset would contain images taken by a large variety of cameras to ensure any model will generalize and not just learn the camera's fingerprint.

Due to the reasons discussed above, machine learning based steganalysis models will need to be improved and trained on larger datasets with a variety of image formats, contents, sizes, and color spaces to make them applicable to deploy in the real world.

6. Conclusion

In this paper, we developed an ensemble model that utilizes SRM for feature extraction, PCA for dimensionality reduction, and DNNs to perform JPG image steganalysis across various state of the art steganography algorithms and embedding rates. To the best of our knowledge, our proposed model, Neural Spatial Rich Models (NSRM), outperforms the currently existing machine learning based JPG image steganalysis models with a testing accuracy of greater than 99.5% on all four of the steganography algorithms. However since the benchmark dataset was lacking in several aspects such as image format, image size, and image color space, our future work includes creating an improved dataset to measure steganalysis model performance and generalization. This dataset will fill in the gaps left by the BOSSBase dataset. We then plan to continue our research on neural networks for image steganalysis to develop a model that can perform well on this new dataset and will allow us to create a more general model which can be deployed for active defense applications.

7. Acknowledgements

The authors would like to thank Google for providing us with cloud based computation for training and testing our models on the Google Cloud Platform (GCP). We'd also like to thank the other members of our team for their assistance, guidance, and review of our research.

References

[1] Patrick Bas, Tomáš Filler, and Tomáš Pevnỳ. "break our steganographic system": the ins and outs of organizing boss. In *International workshop on information hiding*, pages 59–70. Springer, 2011.

[2] Walter Bender, Daniel Gruhl, Norishige Morimoto, and Anthony Lu. Techniques for data hiding. *IBM systems journal*, 35(3.4):313–336, 1996.

[3] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 14(5):1181–1193, 2018.

[4] BOWS-2. Break our watermarking system - 2nd ed. http://bows2.ec-lille.fr/.

[5] Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.

[6] Tomas Denemark, Vahid Sedighi, Vojtech Holub, Rémi Cogranne, and Jessica Fridrich. Selection-channel-aware rich model for steganalysis of digital images. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 48–53. IEEE, 2014.

[7] Jessica Fridrich. Steganographic algorithms. http://dde.binghamton.edu/download/stego_algorithms/.

[8] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[9] Miroslav Goljan and Jessica Fridrich. Cfa-aware features for steganalysis of color images. In *Media Watermarking, Security, and Forensics 2015*, volume 9409, page 94090V. International Society for Optics and Photonics, 2015.

[10] Vojtěch Holub and Jessica Fridrich. Designing steganographic distortion using directional filters. In *2012 IEEE International workshop on information forensics and security (WIFS)*, pages 234–239. IEEE, 2012.

[11] Vojtěch Holub, Jessica Fridrich, and Tomáš Denemark. Universal distortion function for steganography in an arbitrary domain. *EURASIP Journal on Information Security*, 2014(1):1, 2014.

[12] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Jan Kodovskỳ and Jessica Fridrich. Steganalysis of jpeg images using rich models. In *Media Watermarking, Security, and Forensics 2012*, volume 8303, page 83030A. International Society for Optics and Photonics, 2012.

[15] Jan Kodovsky, Jessica Fridrich, and Vojtěch Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2011.

[16] Bin Li, Ming Wang, Jiwu Huang, and Xiaolong Li. A new cost function for spatial image steganography. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 4206–4210. IEEE, 2014.

[17] MITRE. Apt37. https://attack.mitre.org/groups/G0067/.

[18] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[19] Tomáš Pevnỳ, Tomáš Filler, and Patrick Bas. Using high-dimensional image models to perform highly undetectable steganography. In *International Workshop on Information Hiding*, pages 161–177. Springer, 2010.

[20] Yinlong Qian, Jing Dong, Wei Wang, and Tieniu Tan. Deep learning for steganalysis via convolutional neural networks.

In *Media Watermarking, Security, and Forensics 2015*, volume 9409, page 94090J. International Society for Optics and Photonics, 2015.

[21] Steven W Smith et al. The scientist and engineer's guide to digital signal processing, 1997.

[22] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

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

[24] Guanshuo Xu. Deep convolutional neural network to detect j-uniward. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pages 67–73. ACM, 2017.

[25] Xiaoyu Xu, Yifeng Sun, Guangming Tang, Shiyuan Chen, and Jian Zhao. Deep learning on spatial rich model for steganalysis. In *International Workshop on Digital Watermarking*, pages 564–577. Springer, 2016.

[26] Aliakbar Zahravi. Cybercriminals use malicious memes that communicate with malware. https://blog.trendmicro.com/trendlabs-security-intelligence/cybercriminals-use-malicious-memes-that-communicate-with-malware/, Dec 2018.