Conceal and Reveal: Exploring Image Steganography and Steganalysis

Ermina Ashraf Izma Khurram Aaryaa Moharir Lerich Osay Daniel Suh
Victoria Vynnychok Dr. Murat Kantarcioglu

Introduction

Steganography has various forms and uses, whether it be as invisible ink messages or digital watermarks. Although it is often confused with cryptography, steganography does not hide a message's meaning but rather conceals its existence. Steganalysis is then used to determine whether steganography has been used on content, such as an image, and reveals the hidden message.

Our research consisted of using various techniques to implement steganography and steganalysis, and experimenting with cryptography to hide a message's meaning. We created and ran various machine learning models using both convolutional neural networks (CNN), such as ResNet50 and EfficientNetV2-S, and support vector machines (SVM) to detect whether or not an image had been steganographically modified.

Steganographic Methods

The main methods we researched were least-significant-bit (**LSB**) steganography, **JMiPOD**, **UERD**, and **J-UNIWARD**. The LSB algorithm simply changes the least significant bits of pixel values to hide information as the human eye barely notices small changes in color values. More advanced algorithms tend to use the quantized DCT (discrete cosine transform) coefficients for hiding information while minimizing statistical differences. DCT coefficients are created during the JPEG compression process, generally in 8x8 blocks. Small changes in these coefficients (which are then converted back into pixels when a JPEG is opened) are harder to detect. J-UNIWARD selectively embeds in higher frequency areas, UERD attempts to uniformly embed across all DCT coefficients, and JMiPOD focuses on minimizing the power (true positive likelihood) of an optimal detector.

Our Dataset

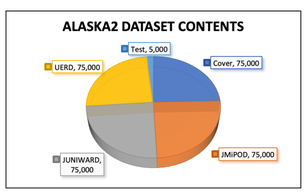

Figure 1. ALASKA2 Dataset Images

The dataset we used was the ALASKA2 Image Steganalysis dataset on Kaggle. It contains **over 300,000** images total, consisting of 75,000 cover images without steganography and over 200,000 steganographic images using UERD, JMiPOD, and JUNIWARD algorithms. The dataset also contains 5,000 test images.

Model Methodology

SVM Model: The first machine learning model we implemented was scikit-learn's SVM vector classifier. SVMs are a type of shallow learning model that calculates a 'hyperplane' that best separates data into classifications within an n-dimensional space based on n features. Because of the shallow nature of SVMs, we decided to use a sample of 1000 steganographic images and 1000 cover (non-steganographic) images to train and test our model using a **support vector classifier (SVC)**, a type of SVM model within the scikit-learn Python library.

CNN Model: We used Tensorflow and Keras to experiment with two different CNN models for binary classification: **ResNet50** and **EfficientNetV2-S**. These models have top scores on ImageNet, and they were chosen to better pick up on the weak steganographic signals left by the algorithms. We also briefly tried multi-class classification with an EfficientNetV2-S backbone in PyTorch, but it was more challenging than expected to get good accuracy, and we decided to not include the results in the poster.

Results

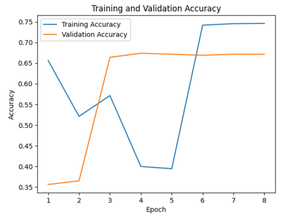

Figure 2. EfficientNetV2-S Accuracy Graph

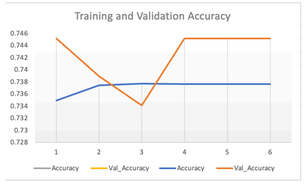

Figure 3. ResNet50 Accuracy Graph

Figure 4. EfficientNetV2-S Loss Graph

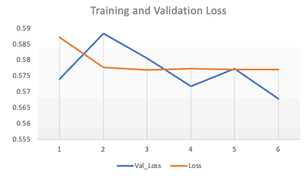

Figure 5. ResNet50 Loss Graph

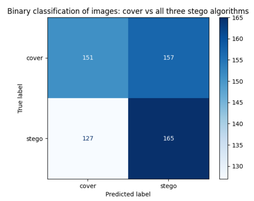

Figure 6. SVC test results' confusion matrix

Analysis and Comparison

Our most accurate model was the ResNet50 model with an accuracy of **74%**. The EfficientNetV2-S model had **68%** accuracy, and our SVC model had the lowest accuracy of **53%**. Our EfficientNetV2-S model seemed to over-fit after the sixth epoch and had trouble learning in general. The ResNet50 was more stable, although a similar plateau occurred near epoch 4. The SVC model had the lowest accuracy, as expected, likely because of its smaller parameter size and dataset.

Conclusion

We found that steganalysis for the more advanced algorithms was *particularly* challenging. Even with deep CNNs, it was hard to reach an accuracy above 80%. Employing techniques like cross-validation, hyperparameter tuning, and ensemble learning would be good next steps, along with experimenting with different hand-crafted features (YCbCr, DCT). Overall, we loved learning about the intricacies of different steganographic algorithms and steganalysis techniques.

References

Holub, V., Fridrich, J., Denemark, T. (2014). Universal distortion function for steganography in an arbitrary domain. Eurasip Journal on Information Security, 2014(1).

Płachta, M., Krzemień, M., Szczypiorski, K., Janicki, A. (2022). Detection of image steganography using deep learning and ensemble classifiers. Electronics, 11(10), 1565.

Rémi Cogranne, Quentin Giboulot, Patrick Bas. (Jun 2020). Steganography by Minimizing Statistical Detectability: The cases of JPEG and Color Images.. ACM Information Hiding and MultiMedia Security (IHMMSec), , Denver, CO, United States