EXPLAINING AND IMPROVING THE JPEG COMPATIBILITY ATTACK WITH STATISTICAL HYPOTHESIS TESTING AND DEEP LEARNING

BY

ELI DWORETZKY

BS, Binghamton University, 2020
BA, Binghamton University, 2020

THESIS

Submitted in partial fulfillment of the requirements for
the degree of Master of Science in Electrical and Computer Engineering
in the Graduate School of
Binghamton University
State University of New York
2021

Accepted in partial fulfillment of the requirements for
the degree of Master of Science in Electrical and Computer Engineering
in the Graduate School of
Binghamton University
State University of New York
2021

December 17$^{\text{th}}$, 2021

Jessica Fridrich, Chair and Faculty Advisor
Department of Electrical and Computer Engineering, Binghamton University

Scott Craver, Member
Department of Electrical and Computer Engineering, Binghamton University

Emrah Akyol, Member
Department of Electrical and Computer Engineering, Binghamton University

Abstract

The JPEG compatibility attack is a steganalysis method for detecting messages embedded in the spatial representation of decompressed JPEG images. This thesis focuses on a novel approach that improves the detection accuracy for the difficult case of high JPEG qualities and content-adaptive stego algorithms. Close attention is paid to the robustness of the detection with respect to the JPEG compressor and DCT coefficient quantizer. A likelihood ratio detector derived from a model of quantization errors of DCT coefficients is used to explain the main mechanism responsible for detection and to understand the experimental results. The most accurate detector is an SRNet trained on a two-channel input consisting of the image and its SQ error. Additional mathematical formulations are also discussed to gain insight into and offer alternative perspectives on the attack.

Acknowledgments

I would like to thank my advisor Prof. Jessica Fridrich for her mentorship, wisdom, and generosity with her time. My experience working with her has been an invaluable part of my education. She inspired and motivated me to work ambitiously, think creatively, and also have fun.

I would also like to thank the members of the Digital Data Embedding laboratory — in particular, Jan, Yassine, and Edgar — for their assistance and camaraderie. My appreciation also goes out to my friends and extended family, especially to my friend Matt for his companionship during the pandemic.

Finally, I am sincerely grateful to my parents and my brother Aaron for their support and encouragement all throughout my studies.

Eli Dworetzky

Contents

List of Tables

List of Figures

Preface

This thesis is largely based on the author's manuscript "JPEG Compatibility Attack Revisited" with J. Butora and J. Fridrich which is currently in peer review for publication. The digital images used in figures and experiments originate from the BOSSbase 1.01 [2] and BOWS2 [3] datasets.

Chapter 1 provides relevant background material for and establishes the relevance of the JPEG Compatibility Attack. Chapter 2 conveys a robust, novel approach to the JPEG Compatibility Attack which is the main contribution of the thesis. Chapter 3 studies other mathematical formulations of the attack as a means to gain insight and perspective on the problem. Appendix A contains supplementary findings and analysis.

Chapter 1

Introduction

Steganography is the practice of concealing secret information within physical or digital objects as a means of covert communication.

In this thesis, we are concerned with a subclass of steganalysis techniques called compatibility attacks. In particular, we study and modernize the JPEG Compatibility Attack (JCA) in light of the most recent advances in image steganography and steganalysis.

Steganography is often formulated and motivated by the famous prisoners' problem [42]. Suppose Alice and Bob are prisoners locked in separate cells. Their only means of communication is through a channel completely monitored by the warden. If the warden discovers that the prisoners are communicating for devious purposes such as devising an escape plan, she will cut their communication and punish them accordingly. A *malicious* warden hopes to deceive the prisoners by fabricating fake messages or modifying authentic ones. An *active* warden may distort the channel by lossy compression, noise adding, etc. A *passive* warden simply eavesdrops. Indeed, if the prisoners wish to concoct an escape plan, they cannot use cryptography alone since encrypted messages are often in the form of unintelligible garble. The warden will be suspicious of an encrypted message in plain sight no matter how unbreakable it is. Therefore, the prisoners should send ostensibly benign messages, concealing the fact that secret information is being exchanged at all.

First, we briefly review the modern foundation of steganography and steganalysis in Section 1.1 and 1.2 as well as the history of compatibility attacks in Section 1.3. In Section 1.4, we introduce the notation used throughout the thesis and briefly discuss relevant background material from the field of directional statistics. Section 1.5 describes the JCA processing pipeline considered in this work, which involves the initial JPEG compression of the cover image, decompression, embedding, and subsequent recompression and decompression used by the steganalyst.

1.1 Steganography

We constrain ourselves to image steganography through cover modification; that is, a *cover* image is drawn from a source (e.g., a camera, database, or website) and subtly altered to embed the secret message, producing a *stego* image. Cover images serve as camouflage for secret messages. We assume the warden is passive which is common in literature and in

practice [13], and we are only concerned with spatial-domain steganography, which is often accomplished by modifying the Least Significant Bits (LSBs) of the cover's pixel values.

The prototypical steganographic scheme is LSB Replacement (LSBR). Given a message in the form of a bitstream, LSBR synchronizes the LSBs of a pseudo-random path of pixels with the bitstream by flipping LSBs whenever a mismatch occurs. We assume the bitstream is unbiased which is approximately the case when encryption precedes embedding. Without coding, embedding a relative payload of α, measured in bits per pixel (bpp), yields the overall change rate $\beta = \alpha/2$. Thus, LSBR can be effectively simulated by flipping LSBs with probability β.

Modern algorithms, however, are built using LSB Matching (LSBM); i.e., whenever a mismatch in parity occurs, pixel values are changed by $+1$ or -1 each with change rate β (so that the rate of no change is $1 - 2\beta$). Additionally, modern schemes utilize matrix embedding [13] (eliminating the need for a pseudo-random path) which means the change rate now adheres to the rate-distortion bound $\beta \geq H_3^{-1}(\alpha)$, where H_3 is the ternary entropy function given by

$$H_3(x) = -2x \log_2 x - (1 - 2x) \log_2(1 - 2x). \tag{1.1.1}$$

The rate-distortion bound is attained $\beta = H_3^{-1}(\alpha)$ for optimal codes. Moreover, content-adaptive schemes use change rates β_i that differ across pixels, since it is generally desirable for the embedding changes to occur in regions of noisy or textured content rather than regions of smooth or singular content. The β_i are often called the selection channel. An optimal coding scheme will then minimize a distortion measure between cover and stego objects while satisfying the payload constraint

$$\sum_{i=1}^{n} H_3(\beta_i) = n\alpha. \tag{1.1.2}$$

The so-called Syndrome Trellis Codes (STCs) [12], a version of sparse convolutional codes, are nearly optimal with respect to the rate-distortion bound and are the current state-of-the-art. In this thesis, all content-adaptive schemes are simulated by modifying pixels by ± 1 with the probabilities β_i as if optimal coding was used — the worst case for the warden. For more details, the reader is referred to [13].

Furthermore, modern schemes are typically either cost-based or model-based. Most cost-based schemes try to minimize a linear additive distortion function

$$\min \sum_{i=1}^{n} \rho_i \beta_i, \tag{1.1.3}$$

subject to the payload constraint in Eq. (1.1.2) where ρ_i is a heuristically designed cost that describes the impact of changing pixel i. Eq. (1.1.3) admits the closed form solution

$$\beta_i = \frac{e^{-\lambda \rho_i}}{1 + 2e^{-\lambda \rho_i}}, \tag{1.1.4}$$

where $\lambda > 0$ is a Lagrange multiplier dependent on α. Examples include S-UNIWARD [23], HILL [33], and WOW [21]. On the other hand, model-based schemes are designed using statistical models rather than heuristics. One such example is MiPOD [41] which tries to minimize the deflection coefficient of the optimal detector under a heteroscedastic Gaussian model for a noise residual. Figure 1.1.1 shows an example of embedding changes made when using the modern scheme HILL.

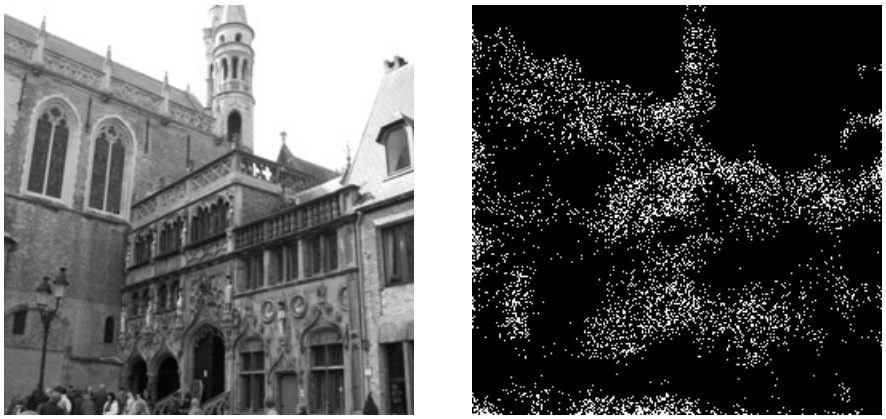

Figure 1.1.1: Left: A decompressed JPEG (quality factor 90) used as a cover image. Right: Stego changes (shown in white) made due to embedding a payload of 0.4 bpp using HILL.

1.2 Steganalysis

As portrayed in the prisoners' problem, the goal of the warden (or steganalyst) is to detect the presence of secret communication. Under the assumption that cover and stego images are random variables, the warden could perform a simple hypothesis test on the observed images if the cover and stego distributions are known. Due to the complex diversity and huge dimensionality of images, hypothesis testing is usually infeasible. Consequently, machine learning classifiers and data-driven techniques are often used in practice to detect steganography in single images.

The current state-of-the-art detectors for steganalysis are in the form of Convolutional Neural Networks (CNNs). Due to the design of convolutional filters, CNNs are exceptional at forming local statistics about image content. Since the signal of interest is the embedding changes, however, the pioneering CNNs designed for steganalysis, such as XuNet [47], used fixed high-pass filters in the first convolutional layers. Additionally, most CNNs include pooling layers to mitigate the influence of image noise which is counterproductive for steganalysis since embedding changes are a type of low energy noise pattern. To prevent suppression of the stego signal, a deep residual network called SRNet [6] contains seven convolutional layers at the front with pooling disabled. SRNet reaches state-of-the-art performance and will be used for building deep learning steganalyzers in this thesis.

Before the adoption of deep learning in steganalysis, heuristically designed feature sets would be manually collected and then fed to traditional machine learning tools for training and testing. The most popular feature set for spatial-domain steganalysis was the so-called Spatial Rich Model (SRM) [15]. Due to the dimensionality of SRM, classifiers such as Support Vector Machines (SVM) were relatively time consuming to train. The FLD-Ensemble classifier [31] was shown to ease the computational burden while achieving a similar performance.

1.3 Compatibility Attacks

Compatibility attacks are specialized steganalysis methods that exploit some inherent structure found in the cover source. Compatibility attacks typically become possible when covers are preprocessed using a many-to-one mapping that imposes strict constraints on pixel values or any other representation of the image. Hence, a violation of compatibility conditions can be used as proof of image tampering or steganography. Compatibility attacks typically achieve extremely high detection performance compared to standard steganalysis methods. In this section, we review some history about the JCA, the main focus of this thesis, as well as some other examples of attacks.

1.3.1 JPEG Compatibility Attack

The JCA is a specialized image steganalysis method that can reliably detect messages embedded with spatial-domain steganography under the assumption that the cover image is a decompressed JPEG. The compression imposes strict constraints on the spatial-domain representation, which allows very accurate detection of pixel modifications even for small payloads. For low enough qualities, it is even possible to extract the embedding changes from a stego image since the process of recompressing and decompressing the image will return the original cover. The assumption that the cover was originally stored as JPEG is feasible as the vast majority of images are stored in the JPEG format. Steganographers might hide data in the spatial domain because it offers a larger embedding capacity or simply because the data hiding program cannot handle the JPEG format.

The attack was originally conceived in [14] based on the idea that one could prove that a given image contains blocks of 8×8 pixels that could not be obtained by decompressing any combination of 64 quantized Discrete Cosine Transform (DCT) coefficients. A brute-force search in the form of a tree-pruning algorithm was proposed to obtain such proof. For larger quality factors (smaller JPEG quantization steps), the complexity of this search increases rapidly, which makes this attack impractical to use at scale. Moreover, since the original JPEG compressor is not available to the steganalyst, in practice the incompatibility of a block would also need to be verified with respect to all JPEG decompressors, which further increases the complexity and may not even be feasible.

A quantitative version of this attack that estimates the change rate introduced by Least Significant Bit (LSB) replacement was proposed in [4, 5], where a recompressed-decompressed version of the image was used as a pixel predictor in the weighted Stego-Image (WS) attack [28]. The detection accuracy of this attack is fairly robust with respect to errors in the estimated quantization table as well as different JPEG compressors. This approach is, however, fundamentally limited to LSB replacement and cannot detect embedding that uses LSB matching, which is the case of all modern content-adaptive stego algorithms. The same recompression predictor was also used in [36], where the number of pixels by which the stego image and its recompressed version differed was used as the detection statistic. The departure from the WS detector allowed detection of embedding operations other than LSB replacement.

An improved localized version of this attack was described in [30] by counting the number of different pixels between the image and the recompressed-decompressed version in each 8×8 block. An example of this difference image, called the *recompression residual*, is

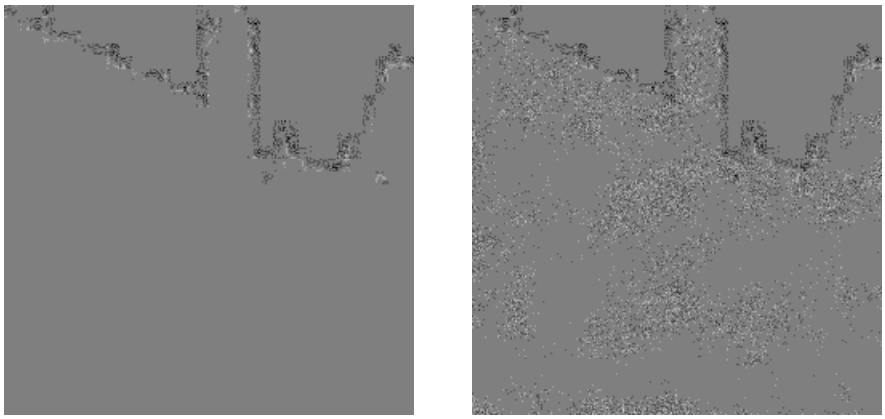

Figure 1.3.1: Left: The recompression residual for the cover image in Figure 1.1.1. Artifacts can appear when pixels are 0 or 255 (on the boundary of the 8-bit dynamic range). Right: The recompression residual for the stego image in Figure 1.1.1. The embedding changes are apparent for QF90.

visualized in Figure 1.3.1. A 65-dimensional histogram of these counts, which we call the *recompression residual histogram* (RRH), served as a feature vector for training a classifier. The authors reported a markedly improved detection accuracy especially for larger quality factors and small payloads.

In general, all forms of the JCA become less accurate for high qualities because the process of recompression-decompression, which is used as a powerful reference, is more affected by rounding in the spatial domain when decompressing the original cover image. The stego changes thus become harder to distinguish from recompression artifacts, which decreases the detection accuracy especially for content-adaptive steganography as the recompression artifacts and stego changes often occur in approximately the same areas of the image. Addressing these deficiencies is one of the main goals of this thesis.

1.3.2 Reverse JPEG Compatibility Attack

The reverse JPEG Compatibility Attack (RJCA) is a steganalysis method recently invented in 2019 that applies to JPEG-domain steganography instead [8]. In particular, the RJCA is concerned with JPEG images compressed using a quality 99 or 100 standard quantization table (or a custom quantization table that closely resembles 99 or 100). The driving mechanism of the attack is the observation that the (spatial-domain) rounding errors due to JPEG decompression follow what we call a wrapped Gaussian distribution in this thesis (see Section 1.4.2). A steganographic embedding will cause an unnatural increase in the variance of this distribution which is statistically detectable through hypothesis testing and machine learning classifiers. For lower JPEG qualities, however, the RJCA loses its detection power because the cover and stego rounding error distributions become indistinguishable. The statistical model used in [8] was one of the inspirations for the novel approach studied in Chapter 2.

A selection-channel aware (SCA) version of the RJCA was proposed in [10]. In particular, this work sought to improve the hypothesis test in [8]. The impact of embedding was modeled as a shift in the mean of a multivariate extension of the wrapped Gaussian distribution.

Under this statistical model, however, the likelihood ratio test is infeasible in practice since the stego distribution is a mixture over all 3^{64} possible shifts in the mean (for ternary embedding in an 8×8 block). Thus, the model was simplified to a shifted multivariate Gaussian distribution by ignoring the "wrapped" part. When the stego scheme is unknown, the test can be further modified into an energy detector whose terms are weighted by the L^1 norms of the DCT blocks (since adaptive schemes tend to embed in DCT blocks with large values). Both the SCA test and weighted test outperformed the prior art's hypothesis test based detector. The simplifying assumptions used in [10] do not have direct analogs for spatial-domain steganography so the mean-shift model is impractical for the JCA.

1.3.3 Compatibility constraints in rich models

For some image sources and rich models, powerful compatibility constraints are placed on the histogram / co-occurrence features of specific submodels. For example, a compatibility attack for color images exists for covers developed in 'dcraw' using the AHD and PPG demosaicking algorithms. In particular, the pixel constraints are so tight that only eight features in the 'minmax41' submodel of the Color Rich Model (CRM) are needed for extremely accurate detection [16].

1.4 Preliminaries

1.4.1 Notation

This section introduces the notation most frequently used throughout this work. Less important notation will be defined when needed.

The operation of rounding $x \in \mathbb{R}$ to the nearest multiple of a positive integer q is denoted by $[x]_q \triangleq q \cdot [x/q]$, where the square bracket is the operation of integer rounding $[x]_1 = [x]$. The quantization (rounding) error is defined as $\mathrm{err}_q(x) \triangleq x - [x]_q$. Rounding x "towards zero" is denoted as $\mathrm{trunc}(x)$ and is defined as $\mathrm{trunc}(x) = \lfloor x \rfloor$ for $x \geq 0$ and $\mathrm{trunc}(x) = \lceil x \rceil$ for $x < 0$, where $\lfloor x \rfloor$ and $\lceil x \rceil$ represent flooring and ceiling. Clipping x to a finite dynamic range $[0, 255]$ is denoted $\mathrm{clip}(x)$ with $\mathrm{clip}(x) = x$ for $x \in [0, 255]$, $\mathrm{clip}(x) = 0$ for $x < 0$ and $\mathrm{clip}(x) = 255$ for $x > 255$. The symbol $\triangleq$ is used whenever a new concept is defined. The uniform distribution on the interval $[a, b]$ will be denoted $\mathcal{U}[a, b]$ while $\mathcal{N}(\mu, \sigma^2)$ is used for the Gaussian distribution with mean μ and variance σ^2. If X is a random variable, then f_X, $\mathbb{E}[X]$, and $\mathrm{Var}[X]$ denote the probability density (PDF), expectation, and variance of X, respectively.

Boldface symbols are reserved for matrices and vectors. The symbols $'\odot'$ and $'\oslash'$ denote element-wise product and division between vectors / matrices of the same dimensions. For readability, we slightly abuse notion when referring to the (element-wise) matrix extensions of the above operations. For example, rounding $\mathbf{x} \in \mathbb{R}^{m \times n}$ with respect to a matrix $\mathbf{q}$ is defined by $[\mathbf{x}]_\mathbf{q} \triangleq \mathbf{q} \odot [\mathbf{x} \oslash \mathbf{q}]$ where $[\cdot]$ denotes element-wise integer rounding in this context. Similarly, we define $\mathrm{err}_\mathbf{q}(\mathbf{x}) \triangleq \mathbf{x} - [\mathbf{x}]_\mathbf{q}$.

1.4.2 Directional statistics

Here, we recall some results from directional statistics needed for the JCA in this thesis. For any real-valued random variable X and positive integer q, the distribution of the quantization error $\mathrm{err}_q(X)$ is obtained by wrapping the distribution of X onto a circle with circumference q. In other words, $\mathrm{err}_q(X)$ has a *wrapped* PDF of the form $\sum_{n\in\mathbb{Z}} f_X(x+qn)$ with a support confined to the half-open interval $[-q/2, q/2)$. In the case $X \sim \mathcal{N}(\mu, \sigma^2)$, the quantization error $\mathrm{err}_q(X)$ follows a wrapped Gaussian distribution $\mathcal{N}_{\mathcal{W}}(\mu, \sigma^2, q)$ whose PDF is given by

$$g(x; \mu, \sigma^2, q) \triangleq \frac{1}{\sqrt{2\pi\sigma^2}} \sum_{n\in\mathbb{Z}} \exp\left(-\frac{(x-\mu+qn)^2}{2\sigma^2}\right), \tag{1.4.1}$$

when $-q/2 \le x < q/2$ and $g(x; \mu, \sigma^2, q) = 0$ otherwise. We note that the wrapped Gaussian is equivalent to what was called a *folded* Gaussian in [8] and [10]. However, since the class of wrapped distributions is well studied and is the standard nomenclature in directional statistics [37], we use the term *wrapped* hereafter.

The wrapped Gaussian is adequately approximated by the truncated sum over the $2N+1$ terms for which $n \in \{0, \pm 1, \ldots, \pm N\}$; the choice of N depends on μ, σ^2, q and the desired precision [37]. For example, $g(x; 0, 1/12, q)$ is well-approximated by one term ($n = 0$) for $q \ge 2$ and three terms ($n = -1, 0, 1$) for $q = 1$. General bounds for the approximation error are found in [32, 8, 37].

Finally, we recall a fundamental asymptotic result known as Poincaré's Limit Theorem (PLT) [37]. If X is an absolutely continuous random variable and q is fixed, then the distribution of $\mathrm{err}_q(cX)$ tends to the uniform distribution $\mathcal{U}[-q/2, q/2)$ as $c \to \infty$. The following extension of the PLT is developed in [26] for wrapping a joint distribution onto a torus. Let M be a n-torus, that is the set $\prod_{i=1}^{n}[-q_i/2, q_i/2)$ where $\prod$ denotes the Cartesian product and $\mathbf{q} \in \mathbb{R}^n$. We can wrap $\mathbb{R}^n$ onto M via the map $\mathrm{err}_{\mathbf{q}}$. If X is an absolutely continuous random vector on $\mathbb{R}^n$, then the distribution of $\mathrm{err}_{\mathbf{q}}(cX)$ tends to the uniform distribution on M as $c \to \infty$.

1.5 JCA pipeline

In this section, we introduce the pipeline through which an originally uncompressed (raw) image is JPEG compressed and then decompressed for spatial-domain embedding, and possibly embedded with a secret message. For clarity, all objects included in this initial compression-decompression will be denoted with a superscript $'(0)'$. JPEG compression proceeds by dividing the image into 8×8 blocks, applying the DCT to each block, dividing the DCT coefficients by quantization steps, and rounding to integers. The coefficients are then arranged in a zigzag fashion and losslessly compressed to be written as a bitstream into the JPEG file together with a header. In this thesis, we constrain ourselves to grayscale images. More details about the JPEG format can be found in [40].

The original uncompressed 8-bit grayscale image with $N_1 \times N_2$ pixels is an element of $\{0, 1, \ldots, 255\}^{N_1 \times N_2}$. Throughout this thesis, $\mathbf{x}^{(0)} = (x_{ij}^{(0)})$ denotes one specific 8×8 block of uncompressed pixels where $0 \le i, j \le 7$. For clarity, we strictly use i, j to index pixels and k, l to index DCT coefficients.

During JPEG compression, the block of DCT coefficients before quantization, $\mathbf{y}^{(0)} \in \mathbb{R}^{8\times 8}$, is obtained using the formula $y^{(0)}_{kl} = \mathrm{DCT}_{kl}(\mathbf{x}^{(0)}) \triangleq \sum_{i,j=0}^{7} f^{ij}_{kl} x^{(0)}_{ij}$, $0 \le k, l \le 7$, where

$$f^{ij}_{kl} = \frac{w_k w_l}{4} \cos\frac{\pi k(2i+1)}{16} \cos\frac{\pi l(2j+1)}{16}, \tag{1.5.1}$$

are the discrete cosines and $w_0 = 1/\sqrt{2}$, $w_k = 1$ for $0 < k \le 7$. The pair (k, l) is called the kl^{th} DCT mode. Before applying the DCT, each pixel is adjusted by subtracting 128 from it during JPEG compression, a step we omit here since it has no effect on our analysis. For brevity, we will also use matrix notation and denote the DCT of a block $\mathbf{u}$ as $\mathbf{v} = \mathbf{D}\mathbf{u}$ where $v_{kl} = \mathrm{DCT}_{kl}(\mathbf{u})$ for all k, l. Here, $\mathbf{D}$ is a 64×64 matrix of discrete cosines and $\mathbf{u}$, $\mathbf{v}$ are the blocks rearranged as column vectors. Note that $\mathbf{D}^\top = \mathbf{D}^{-1}$ due to orthonormality.

The block of quantized DCTs is $\mathbf{c}^{(0)} = [\mathbf{y}^{(0)} \oslash \mathbf{q}]$, $c^{(0)}_{kl} \in \{-1024, \ldots, 1023\}$ where $\mathbf{q} = (q_{kl})$ is a luminance quantization matrix of quantization steps q_{kl} supplied in the header of the JPEG file. For a JPEG compressor that uses truncation instead of rounding, $\mathbf{c}^{(0)} = \mathrm{trunc}(\mathbf{y}^{(0)} \oslash \mathbf{q})$.

During decompression, the above steps are reversed. First, dequantizing $\mathbf{c}^{(0)}$ yields $\widetilde{\mathbf{y}}^{(0)} = \mathbf{q} \odot \mathbf{c}^{(0)}$. Applying the inverse DCT, the block $\widetilde{\mathbf{x}}^{(0)}$ of non-rounded pixels after decompression is obtained by $\widetilde{x}^{(0)}_{ij} = \mathrm{DCT}^{-1}_{ij}(\widetilde{\mathbf{y}}^{(0)}) \triangleq \sum_{k,l=0}^{7} f^{ij}_{kl} \widetilde{y}^{(0)}_{kl}$, where $\widetilde{x}^{(0)}_{ij} \in \mathbb{R}$, or in the matrix form $\widetilde{\mathbf{x}}^{(0)} = \mathbf{D}^\top \widetilde{\mathbf{y}}^{(0)}$. The pair (i, j) used to index $\widetilde{x}^{(0)}_{ij}$ is called the ij^{th} JPEG phase [22]. Finally, rounding $\widetilde{\mathbf{x}}^{(0)}$ to integers and clipping to a finite dynamic range $[0, 255]$ produces the fully decompressed block $\mathbf{x} = (x_{ij})$.

At this point, the steganographer may embed the cover image $\mathbf{x}$ with a secret message by introducing embedding changes $\boldsymbol{\eta}$ to produce the stego image $\mathbf{x}^{(s)} = \mathbf{x} + \boldsymbol{\eta}$. In the JCA, the (cover or stego) image is again JPEG compressed and decompressed to obtain a reference image. Since $\mathbf{q}$ is not available in a decompressed JPEG's file format, recompression is performed using a quantization matrix, $\widehat{\mathbf{q}}$, estimated directly from $\mathbf{x}$ or $\mathbf{x}^{(s)}$.

Figure 1.5.1 visually conveys the JCA pipeline considered in this thesis. As shown, the recompressed blocks $\mathbf{y}$, $\widetilde{\mathbf{y}}$, $\widetilde{\mathbf{x}}$ are all defined by repeating the compression process. We omit $\mathbf{c}^{(0)}$ and $\mathbf{c}$ from Figure 1.5.1 since the operation $[\cdot]_{\mathbf{q}}$ combines quantizing and dequantizing into one step. All stego versions of the objects considered in the recompression will be denoted with a superscript $'(s)'$ — the cover versions do not have a superscript.

Moreover, we denote the initial quantization error by $\boldsymbol{\varepsilon}^{(0)} \triangleq \mathbf{y}^{(0)} - \widetilde{\mathbf{y}}^{(0)}$, the decompression (rounding) error in the spatial domain by $\boldsymbol{\delta} \triangleq \widetilde{\mathbf{x}}^{(0)} - \mathbf{x}$, and the recompression quantization error by $\boldsymbol{\varepsilon} \triangleq \mathbf{y} - \widetilde{\mathbf{y}}$. For brevity, we often refer to $\boldsymbol{\varepsilon}$ as the *Q error* and $\mathbf{D}^{-1}\boldsymbol{\varepsilon}$ as the spatial-domain Q error, or *SQ error*. We refer to $\mathrm{clip}([\widetilde{\mathbf{x}}]) - \mathbf{x}$ as the *recompression residual* which was the object of focus in the previous art [30]. Ignoring clipping, the (negative) SQ error can be seen as the unrounded recompression residual since $\mathbf{x}$ is a block of integers:

$$[-\mathbf{D}^{-1}\boldsymbol{\varepsilon}] = [\widetilde{\mathbf{x}} - \mathbf{x}] = [\widetilde{\mathbf{x}}] - \mathbf{x}. \tag{1.5.2}$$

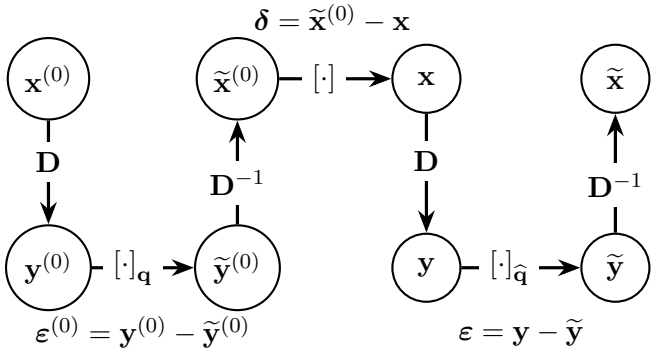

Figure 1.5.1: JPEG compression - decompression - recompression pipeline. Adjusting pixels to $[-128, 127]$ and clipping to $[0, 255]$ are ignored.

Chapter 2

Modern Approach

In this Chapter, we formulate the JCA using statistical and data-driven methods in order to solve the difficult case of high JPEG qualities and make it robust against a wide range of JPEG compressors. Specifically, this novel approach takes strides to make the JCA feasible in practice so that steganalysts may use it reliably in the wild.

First, the pipeline discussed in Section 1.5 is analyzed in Section 2.1 by modeling the quantization errors during the initial compression, which allows us to obtain a detector of steganography as a likelihood ratio test (LRT) in Section 2.2.[1] The LRT is used to obtain insight into the inner workings of the JCA and also explain the trends in detection accuracy observed for detectors in the form of a Convolutional Neural Network (CNN) considered in Section 2.3. Section 2.4 contains the results of the LRT and CNNs — contrasted with the performance of the previous art (RRH) — for a wide range of JPEG quality factors, payloads, and embedding schemes. Section 2.5 is devoted to an important practical aspect of the JCA, which is its robustness to various JPEG compressors and DCT quantizers, including the "trunc" quantizer in common use today [9, 1]. Since the JCA needs to estimate the quantization table of the original JPEG compression, in Section 2.6 we demonstrate that the table can be accurately estimated from the decompressed cover / stego image while pointing out an important fact that, for the purpose of the JCA, only divisors of quantization steps (the so-called sufficient steps) need to be estimated. The chapter is concluded in Section 2.7.

2.1 Pipeline Analysis

Equipped with the tools introduced in Section 1.4.2, we can now study the objects in Figure 1.5.1. We start by modeling the initial quantization error, $\boldsymbol{\varepsilon}^{(0)}$, as a random vector. We then derive the distributions of subsequent objects, ultimately formulating how a steganographic embedding impacts the distribution of the Q errors $\boldsymbol{\varepsilon}$. In summary, the analysis in this section will leverage these facts:

1. The (cover or stego) image is stored using integers which allows us to analytically isolate the rounding errors in each domain.

[1] Research on quantization noise during recompression with high quality factors is potentially relevant to the forensics community [35, 39].

2. The dimensionality of the blocks is high enough to use the Central Limit Theorem (CLT) to approximate the marginals using Gaussians when switching between domains.

3. Poincaré's theorem tells us the distribution of $\boldsymbol{\delta}$ tends to a uniform distribution with jointly independent components as quality factor decreases.

2.1.1 Rounding errors in the spatial domain

By the linearity of the DCT, we can express the non-rounded block of pixels $\widetilde{\mathbf{x}}^{(0)}$ as

$$\begin{aligned}\widetilde{\mathbf{x}}^{(0)} &= \mathbf{D}^{-1}\widetilde{\mathbf{y}}^{(0)} \\ &= \mathbf{D}^{-1}\mathbf{y}^{(0)} - \mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)} \\ &= \mathbf{x}^{(0)} - \mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)}. \end{aligned} \tag{2.1.1}$$

Consider the case of the round quantizer; the values of $\varepsilon_{kl}^{(0)}$ are contained within $[-q_{kl}/2, q_{kl}/2)$.

Assumption 2.1. *For all modes (k,l), the DCT quantization errors $\varepsilon_{kl}^{(0)}$ are jointly independent and satisfy*

$$\varepsilon_{kl}^{(0)} \sim \mathcal{U}[-q_{kl}/2, q_{kl}/2). \tag{2.1.2}$$

Assumption 2.1 has been studied in [44], used in [8, 10, 39], and can be justified directly by the Poincaré Theorem for small quantization steps q_{kl}. By the joint independence of $\varepsilon_{kl}^{(0)}$ and the fact that $\mathbb{E}[\varepsilon_{kl}^{(0)}] = 0$ and $\mathrm{Var}[\varepsilon_{kl}^{(0)}] = q_{kl}^2/12$, Lindeberg's extension of the CLT implies that the marginals of $\widetilde{\mathbf{x}}^{(0)}$ approximately follow the Gaussian distribution

$$\widetilde{x}_{ij}^{(0)} \sim \mathcal{N}(x_{ij}^{(0)}, s_{ij}^{(0)}), \tag{2.1.3}$$

with variance

$$s_{ij}^{(0)} = \frac{1}{12}\sum_{k,l=0}^{7} (f_{kl}^{ij})^2 q_{kl}^2. \tag{2.1.4}$$

The rounding error in the spatial domain has the form

$$\boldsymbol{\delta} = \widetilde{\mathbf{x}}^{(0)} - [\widetilde{\mathbf{x}}^{(0)}] = \mathrm{err}_1(-\mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)}), \tag{2.1.5}$$

because $\mathbf{x}^{(0)}$ is a block of integers. We conclude that the marginals of $\boldsymbol{\delta}$ are approximately distributed by $\delta_{ij} \sim \mathcal{N}_{\mathcal{W}}(0, s_{ij}^{(0)}, 1)$ for all JPEG phases.

We note that when quantization steps are large or when an alternate quantizer such as trunc is used, Assumption 2.1 may no longer hold. Nonetheless, the PLT still allows us to say something about the joint distribution of the rounding errors $\boldsymbol{\delta}$. Looking at Eq. (2.1.4), notice that the probability mass of $\boldsymbol{\varepsilon}^{(0)}$ spreads out as the entries of $\mathbf{q}$ increase. Thus, the distribution of $\boldsymbol{\delta}$ is well-approximated by the joint uniform distribution on $[-1/2, 1/2)^{64}$ for sufficiently low enough quality factors. We experimentally observed that the marginals δ_{ij} are uniform for QFs 98 and below, and thus, we infer that the PLT has applied for these qualities.

Note that if the quantizer is trunc, the variance $\mathrm{Var}[\varepsilon_{kl}^{(0)}]$ is larger compared to round regardless of the distribution of uncompressed DCT coefficients $\mathbf{y}^{(0)}$. Hence, we also conclude that the PLT has applied for QFs 98 and below in the case of trunc.

2.1.2 Cover images

By reasoning similar to that of Eq. (2.1.1), the linearity of the DCT implies

$$\mathbf{y} = \widetilde{\mathbf{y}}^{(0)} - \mathbf{D}\boldsymbol{\delta}. \tag{2.1.6}$$

Assumption 2.2. *The cover block* $\mathbf{x} = [\widetilde{\mathbf{x}}^{(0)}]$ *has rounded to pixels all within the dynamic range* $[0, 255]$. *The rounding errors* $\boldsymbol{\delta}$ *are jointly independent for all JPEG qualities.*

If $\widetilde{x}_{ij}^{(0)}$ is outside the dynamic range, δ_{ij} will belong to an interval potentially much larger than $[-1/2, 1/2)$ with bounds dependent on image content. Using Assumption 2.2, we may ignore the effects of clipping and approximate the marginals of $\mathbf{y}$ using the CLT:

$$y_{kl} \sim \mathcal{N}(\widetilde{y}_{kl}^{(0)}, s_{kl}), \tag{2.1.7}$$

$$s_{kl} = \sum_{i,j=0}^{7} (f_{kl}^{ij})^2 \mathrm{Var}[\delta_{ij}]. \tag{2.1.8}$$

Note that for QFs 98 and below, the approximate uniformity of $\boldsymbol{\delta}$ implies $\mathrm{Var}[\delta_{ij}] \approx 1/12$, which yields $s_{kl} \approx 1/12$ by the orthonormality of the DCT. The Q error computed via the true quantization matrix $\mathbf{q}$ can be expressed as

$$\boldsymbol{\varepsilon} = \mathbf{y} - [\mathbf{y}]_{\mathbf{q}} = \mathrm{err}_{\mathbf{q}}(-\mathbf{D}\boldsymbol{\delta}), \tag{2.1.9}$$

since $\widetilde{y}_{kl}^{(0)}$ is an integer multiple of q_{kl} for all (k, l). Thus, we conclude that $\varepsilon_{kl} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl}, q_{kl})$.

2.1.3 Stego images

We model the embedding changes η_{ij} as content-adaptive ± 1 noise in the spatial domain; we have $\mathbf{x}^{(s)} = \mathbf{x} + \boldsymbol{\eta}$. Specifically, we treat η_{ij} as a random variable supported on $\{-1, 0, 1\}$ with PMF $\mathbb{P}(\eta_{ij} = 1) = \mathbb{P}(\eta_{ij} = -1) = \beta_{ij}$, where β_{ij} are known as the *change rates* (or *selection channel*) determined by the stego scheme. Under this framework, the non-rounded recompressed DCTs have the form

$$\mathbf{y}^{(s)} = \widetilde{\mathbf{y}}^{(0)} - \mathbf{D}\boldsymbol{\delta} + \mathbf{D}\boldsymbol{\eta}. \tag{2.1.10}$$

Assumption 2.3. *The embedding changes* η_{ij} *are jointly independent and independent of the rounding errors* δ_{ij}.

This is a reasonable assumption for steganography that minimizes an additive distortion and does not use the rounding errors as side-information for embedding. Applying the CLT again, we have

$$y_{kl}^{(s)} \sim \mathcal{N}(\widetilde{y}_{kl}^{(0)}, s_{kl} + r_{kl}), \tag{2.1.11}$$

$$r_{kl} = \sum_{i,j=0}^{7} (f_{kl}^{ij})^2 \mathrm{Var}[\eta_{ij}]. \tag{2.1.12}$$

Thus, the Q error for a stego block can be written as

$$\boldsymbol{\varepsilon}^{(s)} = \mathrm{err}_{\mathbf{q}}(-\mathbf{D}\boldsymbol{\delta} + \mathbf{D}\boldsymbol{\eta}), \tag{2.1.13}$$

since $\widetilde{y}_{kl}^{(0)}$ is an integer multiple of q_{kl} for all modes. Hence, $\varepsilon_{kl}^{(s)} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl} + r_{kl}, q_{kl})$ which means the embedding increases the variance of the wrapped Gaussian.

2.2 Statistical Hypothesis Detector

The analysis carried out in the previous section allows us to formulate a statistical hypothesis test about the Q errors for detecting steganography. Then, we introduce rules for eliminating blocks from the test for a tighter fit of modeling assumptions in practice, which improves the detection accuracy. Afterwards, we briefly discuss other considerations for modeling assumptions. The analysis of this section is useful to obtain insight into why and how the JCA works and to explain trends observed for other types of detectors studied in Section 2.3.

All experiments in this section, and in this thesis in general, were conducted on the union of the BOSSbase 1.01 [2] and BOWS2 [3] datasets, each with 10,000 grayscale images resized to 256×256 pixels with `imresize` in Matlab using default parameters. We refer to the union as BOSSBOWS2. This dataset is a popular choice for designing detectors with deep learning because small images are more suitable for training deep architectures [48, 6, 49, 50, 46, 52]. The training set (TRN) contained all 10,000 BOWS2 images along with 4,000 randomly selected images from BOSSbase. The remaining images from BOSSbase were randomly partitioned to create the validation set (VAL) and the testing set (TST) containing 1,000 and 5,000 images, respectively.

2.2.1 Likelihood ratio test

Given a collection $\mathcal{B}$ of 8×8 blocks from an $N_1 \times N_2$ decompressed image, the steganalyst is faced with the following hypothesis test for all $0 \leq k, l \leq 7$ across all blocks $\mathbf{x} \in \mathcal{B}$:

$$\mathcal{H}_0 : \varepsilon_{kl} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl}, q_{kl}) \tag{2.2.1}$$

$$\mathcal{H}_1 : \varepsilon_{kl} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl} + r_{kl}, q_{kl}),\ r_{kl} > 0. \tag{2.2.2}$$

Assumption 2.4. *The Q errors ε_{kl} are jointly independent within and between blocks.*

This assumption allows us to construct a detector from the marginals; working with a joint density leads to similar computational complexity issues encountered in [14, 10]. Thus, the log-likelihood ratio test for an image is

$$\mathcal{L}(\mathcal{B}) = \sum_{\mathbf{x} \in \mathcal{B}} \sum_{k,l=0}^{7} \mathcal{L}_{kl}(\mathbf{x}) \tag{2.2.3}$$

$$= \sum_{\mathbf{x} \in \mathcal{B}} \sum_{k,l=0}^{7} \log \frac{g(\varepsilon_{kl}; 0, s_{kl} + r_{kl}, \widehat{q}_{kl})}{g(\varepsilon_{kl}; 0, s_{kl}, \widehat{q}_{kl})} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma. \tag{2.2.4}$$

Assuming the change rates (and thus r_{kl}) are known, the steganalyst is faced with a simple hypothesis, for which the LRT is uniformly most powerful in the clairvoyant case according to the NP-lemma [27]. As a remark, we remind the reader that the quantization matrix must be estimated from the image first — a preanalytical step discussed in Section 2.6. Until then, we assume the true quantization matrix is known, i.e. $\widehat{\mathbf{q}} = \mathbf{q}$.

Moreover, the LRT is composite if r_{kl} is unknown, which would be the case when detecting multiple steganographic methods, an unknown payload size, or a steganographic

method with unknown or partially known selection channel, e.g. side-informed steganography [25, 23, 19] or methods with synchronized embedding changes [34, 24, 7]. On the other hand, for detecting a known steganography and a known payload size, the selection channel is approximately available — the change rates β_{ij} can be computed from the analyzed stego image — which means that r_{kl} can also be approximately computed. By Lindeberg's extension of the CLT, the normalized LRT

$$\Lambda(\mathcal{B}) = \frac{\mathcal{L}(\mathcal{B}) - \mathbb{E}_{\mathcal{H}_0}[\mathcal{L}(\mathcal{B})]}{\sqrt{\mathrm{Var}_{\mathcal{H}_0}[\mathcal{L}(\mathcal{B})]}} \tag{2.2.5}$$

follows the distribution $\mathcal{N}(0,1)$ under $\mathcal{H}_0$, which allows setting a decision threshold for the normalized LRT that achieves the largest detection power for a fixed false-alarm probability. Figure 2.2.1 shows the distribution of $\Lambda(\mathcal{B})$ under $\mathcal{H}_0$ across images from the training and validation sets when ε_{kl} are sampled from their distributions (2.2.1). Details on computing the moments of $\mathcal{L}(\mathcal{B})$ can be found in Appendix A.2.

2.2.2 Block elimination

In practice, blocks should be eliminated from hypothesis testing if they do not adhere to at least one of the assumptions above; there is no guarantee that the conclusions apply to such blocks. To this end, we formulate rules for rejecting a block $\mathbf{x}$ from $\mathcal{B}$ based on the following common phenomena.

1. Block saturation: A block $\mathbf{x}$ with pixel values x_{ij} is *saturated* if there exists a phase (i,j) such that $x_{ij} = 0, 1, 254,$ or 255.
2. Block sparsity: A block $\mathbf{x}$ is *sparse* if the number of zero DCT coefficients in $\mathbf{y}$ is larger than or equal to 8. To account for floating-point error in the DCT, a coefficient y_{kl} is considered "zero" if $|y_{kl}| < 10^{-5}$.

Saturated blocks potentially violate Assumption 2.2 due to clipping. We include pixel values 1 and 254 to account for the possibility of embedding into pixels at the boundary of the dynamic range. As for sparse blocks, having 8 or more zero DCTs concentrate around zero is highly unlikely since the y_{kl} are Gaussian random variables.[2] Hence, we conclude that the CLT fails for sparse blocks. Therefore, if a block is deemed *saturated* or *sparse* (or both), then the block is rejected. Throughout the chapter, all experiments with block elimination abide by this criteria.

We note that content-adaptive schemes tend to embed in non-saturated and non-sparse blocks. Thus, block elimination may artificially increase the image's overall change-rate which is to the steganalyst's benefit. On the other hand, we do not foresee steganographers intentionally embedding in rejected blocks since doing so would be highly detectable by methods outside the JCA and methods we introduce later in Section 2.3.

The BOSSBOWS2 dataset contains a small number of images (depending on JPEG quality) whose blocks were all eliminated due to lack of content. In our experiments, we eliminated these singular images entirely since they are known to be bad covers.

[2]The authors observed that zero DCTs typically occur in entire rows or columns of modes which is why the sparse block threshold was chosen to be 8.

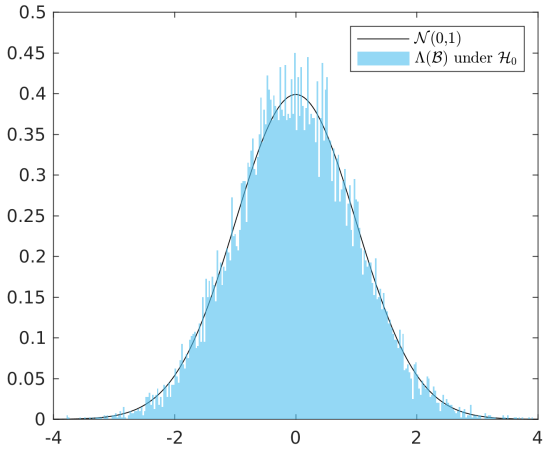

Figure 2.2.1: Distribution of LRT $\Lambda(\mathcal{B})$ under $\mathcal{H}_0$ for Monte-Carlo sampled ε_{kl} with s_{kl} and r_{kl} computed from images in the union of TRN and VAL. One sample of ε_{kl} was taken per DCT mode per block.

2.2.3 Asymptotic performance

In this section, we derive a closed form approximation for the specific case of non-adaptive LSB matching and assuming $q_{kl} > 1$ for all DCT modes. We can approximate $\mathcal{L}(\mathcal{B})$ using the $n = 0$ terms of the wrapped Gaussians, which leads to the energy detector:

$$\ell(\mathcal{B}) = \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \frac{r_{kl}}{s_{kl}(s_{kl}+r_{kl})} \varepsilon_{kl}^2 \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma'. \tag{2.2.6}$$

Assuming the number of pixels $M = N_1 \times N_2 \gg 0$ is large, $\ell(\mathcal{B})$ is approximately Gaussian by the CLT. In particular, the non-adaptivity condition implies that $\mathrm{Var}[\eta_{ij}] = \beta > 0$ is constant for all i, j, and so $r_{kl} = \beta$. Additionally, $q_{kl} > 1$ for all k, l implies $s_{kl} \approx 1/12$. Thus, we have the test statistic

$$\frac{1}{M} \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \varepsilon_{kl}^2 \sim \begin{cases} \mathcal{N}\left(1/12, 2(1/12)^2/M\right) & \text{under } \mathcal{H}_0 \\ \mathcal{N}\left(1/12+\beta, 2(1/12+\beta)^2/M\right) & \text{under } \mathcal{H}_1 \end{cases} \tag{2.2.7}$$

which has the same detection power as $\ell(\mathcal{B})$ for a fixed probability of false alarm. Hence, a closed form approximation of the receiver operating characteristic (ROC) curve is given by

$$P_D = Q\left(\frac{Q^{-1}(P_{FA}) - 12\beta\sqrt{M/2}}{1+12\beta}\right), \tag{2.2.8}$$

where Q is the tail probability of the standard normal random variable, the Q-function [27].

2.2.4 Other considerations

We also experimented with modeling the marginals of the uncompressed DCT coefficients $\mathbf{y}^{(0)}$ as generalized Gaussian [38] random variables. It follows that the quantization error

$\boldsymbol{\varepsilon}^{(0)}$ would be wrapped generalized Gaussian (WGG) distributed, and the phase-dependent variances, s_{kl}, would be computed by numerically integrating the WGG. In practice, though, the shape and width parameters would need to be estimated from $\mathbf{y}$ / $\mathbf{y}^{(s)}$, the unrounded DCTs of the cover / stego image, which complicates matters. However, even when estimating the parameters directly from the uncompressed image, we did not see the LRT benefit. Also, we noticed that the number of terms needed to approximate the WGG becomes unwieldy if the shape and width parameters are too small.

2.3 Machine Learning Detectors

The LRT detector discussed above was derived in the DCT domain under the assumption that the distributions of different 8×8 blocks are independent. The embedding changes are, however, performed in the spatial domain, and the steganalyst can and should make use of dependencies between pixels across the block boundaries, which is ignored by the LRT test. Moreover, the heuristic block rejection rules were adopted based on experiments and are likely an additional source of suboptimality as the modeling assumptions, such as the validity of the CLT, will generally depend on the block content as well as the quality factor. Thus, we anticipate Convolutional Neural Network (CNN) detectors will provide better detection performance especially when supplying the image under investigation as one of the channels on top of the Q / SQ error during training. Such detectors could also potentially be more robust to differences between JPEG compressors simply by enlarging the training set. They can also more easily be made universal in the sense of being able to detect multiple embedding schemes at the same time, covering both round and trunc DCT quantizers, and possibly trained for unknown payloads.

These advantages motivate the study of deep learning based detectors. All previous art made use of the recompression residual $\text{clip}([\widetilde{\mathbf{x}}]) - \mathbf{x}$ as a reference signal, because recompressing the image and then decompressing to the spatial domain essentially erases the embedding changes for lower quality factors. For detecting content-adaptive stego schemes, however, the original image should be used as input so the network can properly learn the selection channel and form better detection statistics from dependencies between neighboring pixels.

Section 2.3.1 and Section 2.3.2 introduce the experimental setup for SRNet and the prior art, respectively.

2.3.1 SRNet

In this thesis, we report the results for three flavors of SRNet [6]: an SRNet trained only on Q errors (Q-SRNet), on SQ errors (SQ-SRNet), and on two channels (SQY-SRNet) — the normalized image $\mathbf{x}/255$ (Y channel) and the SQ error — which provided by far the best overall performance especially for high quality factors. We also investigated an SRNet trained on both the image and its recompression residual but found that it performed worse than the LRT for high QFs. We hypothesize the recompression residual loses information about the embedding after rounding / clipping in the spatial domain.

Training was done for 50 epochs using mini-batches of size 64, the adamax optimizer [29], the one-cycle learning-rate (LR) scheduler with maximum LR 1×10^{-3} [43], and the cross-entropy loss function. All classifiers were trained using a pair-constraint, requiring batches to contain cover-stego pairs.

To augment the training data, a random dihedral group (D4) operation was applied to each cover-stego pair in the batch before extracting Q / SQ errors. Observe that the quantization table must be transposed when images are rotated by 90 or 270 degrees.

In experiments with multiple payloads, we trained networks from scratch on the largest payload with maximum LR 1×10^{-3}. The checkpoint with minimal validation loss was then used as a starting seed for training on smaller payloads with maximum LR 3×10^{-4}. Curriculum training in this manner significantly helped facilitate convergence.

2.3.2 RRH

For comparison against the prior art, we also implemented the RRH method [30] (see Section 1.3.1) trained on the union of the TRN and VAL. The recompression residual was computed using Matlab's `imwrite` and `imread` to match the initial (de)compressor implementation.

2.4 Experiments

In this section, our goal is to determine the best detector from Section 2.2 and 2.3. First, we compare the performance of the LRT and the three SRNets with respect to JPEG quality for a fixed stego scheme and payload. The best detector of these four will then be rigorously tested against the prior art, RRH, for a variety of stego schemes and payloads. Throughout the section, we present the results through the lens of our analysis in Section 2.1.

2.4.1 Methodology

As in Section 2.2, we used the same split 14,000 / 1,000 / 5,000 for TRN / VAL / TST. Images were initially compressed and decompressed using Matlab's `imwrite` and `imread`. In order to compare the LRT to the machine learning detectors, we first choose the decision threshold that minimized P_{E} on the union of TRN and VAL. The measurement P_{E} is the probability of error under equal priors defined by $P_{\text{E}} = (P_{\text{MD}} + P_{\text{FA}})/2$, where P_{MD} and P_{FA} are the probabilities of missed detection and false alarm. The test accuracy of the LRT is then computed on TST using this fixed threshold. Cover-stego pairs were generated using the MiPOD [41] simulator at 0.01 bits per pixel (bpp).

2.4.2 Performance with respect to quality

In Figure 2.4.1, the plot visualizes the trends for the LRT and all versions of SRNet. Since SQY-SRNet outperformed the other detectors especially for high qualities, we continued by testing SQY-SRNet and the prior art on the following four content-adaptive steganographic schemes: S-UNIWARD [23], HILL [33], MiPOD [41], and WOW [21]. These schemes were tested on the following range of payloads: 0.02, 0.01, 0.005, and 0.002 bpp. We refer the reader to Tables A.1 and A.2 in Appendix A.3 for the full results for SQY-SRNet and the prior art. A subset of these results are shown in Figure 2.4.2. The SQY-SRNet significantly outperforms the RRH especially for small payloads for QFs above 93.

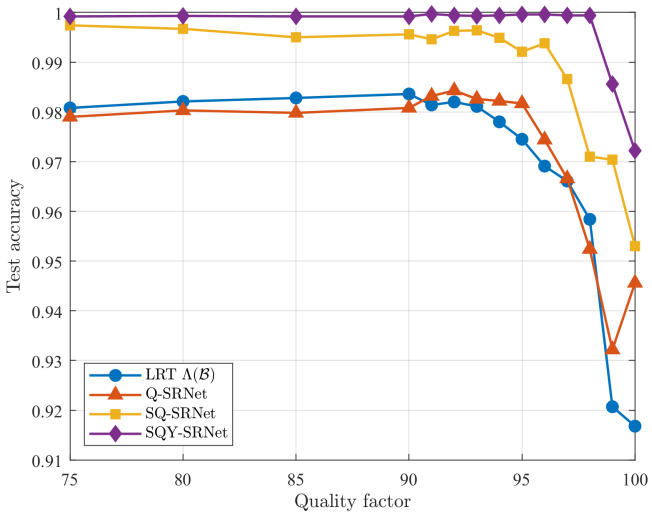

Figure 2.4.1: Testing accuracy as a function of JPEG quality for the LRT (2.2.5) and all flavors of SRNet. Embedded using MiPOD at 0.01 bpp.

Note that the model-based MiPOD is consistently more secure than the other three cost-based stego algorithms. The difference is most pronounced for the smallest payloads and largest qualities. We were able to trace the reason for this difference to the average number of pixels modified by these four schemes. For QF100 and payload 0.002 bpp, the average number of changed pixels for MiPOD, S-UNIWARD, WOW, and HILL are 9.7, 12.2, 13.8, and 14.1, which matches the trend in increased detectability with SQY-SRNet: 0.689, 0.811, 0.863, and 0.872.

We note that the performance of the LRT matches the performance of Q-SRNet except for QFs 99–100. We interpret this overlap as an indication that our modeling assumptions take into account all relevant information contained in the Q error representation of the image (besides inter-block dependencies). We hypothesize that the deviation for QFs 99–100 occurs due to $\boldsymbol{\delta}$ not being jointly independent since the PLT does not apply for these qualities as per Section 2.1.1. This implies the CLT may not apply to the marginals of $\mathbf{y}$, hence the ε_{kl} is not guaranteed to follow the wrapped Gaussian in Section 2.1.2.

We note that SRNet generally has trouble forming inter-block statistics in DCT domain representations [51] which is likely why we see a jump in performance when the SQ error is used instead.

In [14], QF100 is deemed the hardest quality for the JCA due to search complexity. This hints at the existence of suboptimality in the prior art for which QF97 is empirically the hardest quality. Note that SQY-SRNet closely matches the monotonic behavior we intuitively expect.

2.4.3 Detecting a diversified stego source

In this experiment, we trained a multi-class version of SQY-SRNet on four content-adaptive embedding schemes. Table 2.1 shows the 5×5 confusion matrices for QF 90, 95, 100. Observe that the false-positive rate is comparable to the binary SQY-SRNet. Although the

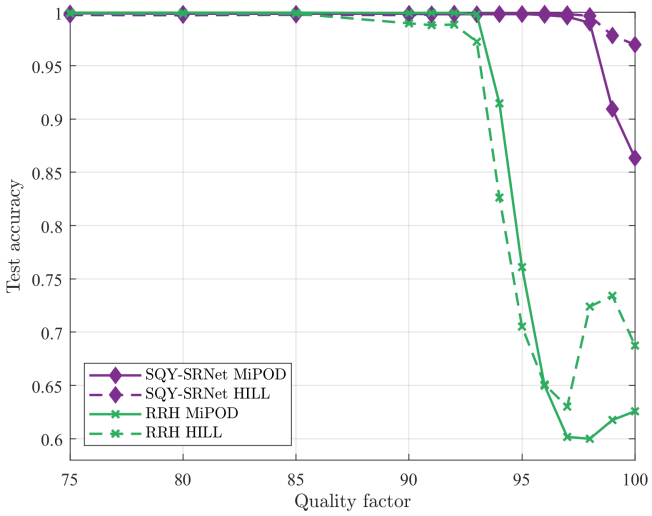

Figure 2.4.2: Testing accuracy as a function of JPEG quality for SQY-SRNet (purple) and RRH (green). Embedded using MiPOD (solid) and HILL (dashed) at 0.005 bpp.

binary SRNets were trained using the pair-constraint, we opted out of any constraint here. The network would not converge from scratch at 0.01 bpp; however, pre-training at 0.02 bpp resolved the issue.

2.5 Robustness to JPEG Compressors

There exist many variants of JPEG compressors, which can differ in the implementation of the DCT, the quantizer, and the internal number representation. If two compressors differ, they may produce different JPEG images from the same raw image. Similarly, if two decompressors differ, they may produce different decompressed images from the same JPEG file. As a result, a cover image can potentially originate from a vast number of JPEG compressor-decompressor combinations. In addition, the steganalyst must use a JPEG variant for recompression and decompression to compute, e.g. the SQ errors. Any mismatch of JPEG combination may complicate the distribution of rounding errors and potentially dramatically decrease the performance of the JCA. Also, machine-learning detectors may perform poorly on a variant not seen during training. In this section, we pinpoint the JPEG compressor variant that should be used for training in order to maximize the robustness of SQY-SRNet.

Since the recompression method for the JCA is the steganalyst's choice, we are free to select the one that works the best overall. Since rounding errors are not easily attainable using off-the-shelf JPEG compressors, we manually recompress via SciPy's `dct` to compute Q / SQ errors for all experiments. To simplify matters, we exclusively use Matlab's `imwrite` to compute the recompression residual for the prior art [30] since this variant was used for benchmarking in Section 2.3.

On the other hand, the steganalyst does not know the compressor-decompressor pair used to obtain the spatial representation of the cover image. The following (de)compressor

Table 2.1: Confusion matrices for the multi-class SQY-SRNet (0.01 bpp). Each row corresponds to the true embedding scheme, and each column corresponds to the predicted scheme.

QF	True scheme	Predicted scheme				
		Cover	MiPOD	HILL	S-UNI	WOW
100	Cover	.9848	.0134	0	0	.0018
	MiPOD	.0234	.9106	.0114	.0014	.0532
	HILL	.0010	.0136	.9552	.0086	.0216
	S-UNI	.0002	.0036	.0088	.8112	.1762
	WOW	.0024	.0484	.0148	.2064	.7280
95	Cover	1	0	0	0	0
	MiPOD	.0010	.9748	.0012	.0004	.0226
	HILL	.0008	.0006	.9974	.0008	.0004
	S-UNI	0	.0002	.0002	.9384	.0612
	WOW	0	.0100	.0004	.0742	.9154
90	Cover	.9996	.0002	0	0	.0002
	MiPOD	.0008	.9820	.0002	.0002	.0168
	HILL	.0010	.0006	.9970	.0004	.0010
	S-UNI	.0004	.0002	.0002	.9522	.0470
	WOW	0	.0136	.0002	.0546	.9316

implementations were considered: Matlab's `imwrite`/`imread`, Python3 library PIL (PIL), ImageMagick's Convert (Convert), Int and Float DCT compressors in libjpeg (version 6b).[3] Fast DCT compression in libjpeg has not been included in our tests because it is not recommended for QFs larger than 97 since the compression is then slower and more lossy than on smaller QFs.[4]

For experimental feasibility, we reduced the number of compressor-decompressor pairs tested by restricting our attention purely to differences between quantizers used for the initial compression. We specifically use Matlab's `imwrite` for its round quantizer and a manually implemented trunc compressor in Python3 using SciPy's `dct`.

2.5.1 Mismatching the decompressor

First, we try to determine the best JPEG decompressor for the steganalyst under the assumption 1) that the original JPEG cover was obtained using a round quantizer for the DCTs and 2) the steganographer was free to choose any of the decompressors. Table 2.2 shows the testing accuracies for SQY-SRNet trained and tested on mismatched decompressors for QFs 95, 99, 100. While a loss can indeed be observed especially in the case when the detector was built with images generated by 'Float' and 'Convert', the detector trained on images from Python's PIL and Matlab's `imread` generalized overall very well when evaluated on images from all five compressors.

We also studied the prior art's robustness to decompressor since no benchmarking exists in [30]. The testing accuracies for the RRH are shown in Table 2.3. We observed that QF99

[3] `http://libjpeg.sourceforge.net/`

[4] Taken from libjpeg documentation `https://manpages.ubuntu.com/manpages/artful/man1/cjpeg.1.html`.

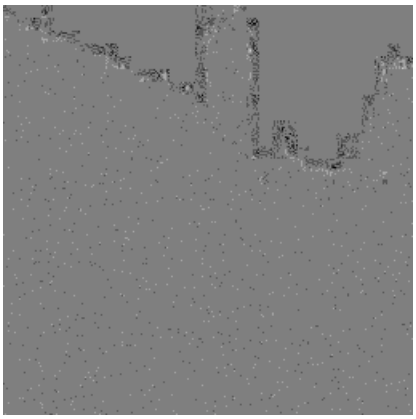

Figure 2.5.1: The recompression residual for the cover image in Figure 1.1.1. The cover was initially compressed-decompressed with SciPy's DCT. However, it was recompressed with Matlab's `imwrite` which introduces noise artifacts that resemble embedding changes.

and 100 had the same pattern in the results (with accuracies in the range $[.7486, .7616]$ for QF100), so we report the results for QFs 90, 95, 99.

The recompression residual will typically contain blocks with no pixel changes or blocks with large patterns of changes; residual blocks will rarely contain single pixel changes especially for QFs with no 1's in the quantization table [30]. Thus, for QF92 and below, embedding is highly detectable since single pixel changes will appear in the recompression residual. We observed, however, that having mismatched JPEG variants in the JCA pipeline commonly creates salt-and-pepper noise artifacts in the recompression residual, which the RRH misinterprets as steganography. An example of this phenomenon is shown in Figure 2.5.1. For example, the accuracy of RRH for QF90 trained and tested on the float decompressor only has an accuracy of .6349 because the compressor is Matlab's `imwrite`. For QFs above 92, mismatching is less problematic since the RRH classifier gets trained on covers that more commonly produce salt-and-pepper noise.

2.5.2 Mismatching the quantizer

Having seen that training on MATLAB's `imread` or PIL generalize the best for decompressor robustness, we turn to investigating robustness to a compressor's quantizer. Table 2.4 shows that training on either the `imread` or PIL decompressor gives similar accuracies when images are initially compressed with a trunc quantizer. Overall, the accuracies are somewhat lower compared to when quantized with round (see Table 2.2) with the largest difference for QF100. This is related to the differences between both quantizers, namely the way they affect the distribution of δ_{ij}. Except for QFs 99–100, $\boldsymbol{\delta}$ is well-approximated by the uniform distribution for both quantizers (see Section 2.1.1). Therefore, the SQ errors for both quantizers approximately follow the same distribution under assumptions of Section 2.1 which explains the matching accuracies for QF95. For QFs 99–100, however, the DCT quantization errors for the trunc quantizer $\varepsilon_{kl}^{(0)} \in [0, q_{kl})$ for positive DCTs and $\varepsilon_{kl}^{(0)} \in (-q_{kl}, 0]$ for negative DCTs. Thus, any asymmetry in the distribution of the DCT coefficients in the cover image transfers to an asymmetry of the quantization errors, giving them a non-zero mean. In contrast, the distribution of quantization errors for the round

Table 2.2: Testing accuracy for SQY-SRNet trained and tested on combinations of decompressors. Each row / column corresponds to the decompressor used for training / testing, respectively. Initially compressed with Matlab's `imwrite`. Embedded using MiPOD at 0.01 bpp.

QF	TRN decomp.	TST decompressor imread	float	int	convert	PIL
100	imread	.9721	.9556	.9716	.9568	.9729
	float	.9500	.9742	.9491	.9739	.9491
	int	.9695	.9587	.9682	.9570	.9685
	convert	.9461	.9732	.9455	.9742	.9456
	PIL	.9721	.9633	.9706	.9635	.9708
99	imread	.9856	.9846	.9870	.9849	.9859
	float	.9781	.9875	.9784	.9899	.9777
	int	.9845	.9833	.9856	.9832	.9838
	convert	.9760	.9878	.9770	.9885	.9771
	PIL	.9843	.9864	.9849	.9860	.9844
95	imread	.9996	.9997	.9993	.9994	.9996
	float	.9991	.9992	.9991	.9992	.9992
	int	.9996	.9996	.9993	.9992	.9995
	convert	.9996	.9994	.9993	.9993	.9996
	PIL	.9994	.9995	.9994	.9992	.9995

Table 2.3: Testing accuracy for RRH trained and tested on combinations of decompressors. Matlab's `imwrite` is used for the initial compressor and used to compute the recompression residual. Embedded using MiPOD at 0.01 bpp.

QF	TRN decomp.	TST decompressor imread	float	int	convert	PIL
99	imread	.7538	.7315	.7523	.7341	.7522
	float	.7481	.7453	.7451	.7485	.7437
	int	.7552	.7323	.7518	.7342	.7512
	convert	.7486	.7446	.7460	.7480	.7448
	PIL	.7540	.7339	.7518	.7363	.7517
95	imread	.9042	.5000	.9031	.5000	.9041
	float	.5288	.6813	.5281	.6828	.5281
	int	.9035	.5000	.9032	.5000	.9041
	convert	.5166	.6834	.5159	.6834	.5172
	PIL	.9022	.5000	.9019	.5000	.9029
90	imread	.9993	.5000	.9993	.5000	.9993
	float	.4819	.6349	.4816	.6359	.4817
	int	.9993	.5000	.9992	.5000	.9993
	convert	.4831	.6350	.4828	.6350	.4823
	PIL	.9993	.5000	.9994	.5000	.9993

Table 2.4: Testing accuracy for SQY-SRNet trained and tested on the trunc quantizer and combinations of decompressors. Embedded using MiPOD at 0.01 bpp.

QF	TRN decomp.	TST decompressor				
		imread	float	int	convert	PIL
100	imread	.8894	.8641	.8918	.8664	.8897
	PIL	.8906	.8608	.8940	.8642	.8923
99	imread	.9830	.9775	.9830	.9770	.9834
	PIL	.9842	.9777	.9824	.9767	.9833
95	imread	.9993	.9993	.9992	.9993	.9996
	PIL	.9994	.9994	.9996	.9994	.9996

Table 2.5: Testing accuracy for SQY-SRNet trained and tested on mismatched quantizers and combinations of decompressors. Embedded using MiPOD at 0.01 bpp.

		Train quantizer: round Test quantizer: trunc					Train quantizer: trunc Test quantizer: round				
QF	Train decomp.	Test decompressor					Test decompressor				
		imread	float	int	convert	PIL	imread	float	int	convert	PIL
100	imread	.5004	.5007	.5004	.5007	.5004	.5049	.5040	.5049	.5045	.5047
	PIL	.5004	.5005	.5004	.5005	.5004	.5050	.5044	.5049	.5052	.5052
99	imread	.8095	.8969	.8093	.8962	.8098	.9335	.9141	.9362	.9133	.9351
	PIL	.7595	.8546	.7591	.8541	.7586	.9204	.8979	.9247	.8984	.9245
95	imread	.9992	.9995	.9995	.9993	.9995	.9993	.9991	.9993	.9993	.9994
	PIL	.9991	.9993	.9994	.9994	.9995	.9996	.9996	.9993	.9994	.9994

quantizer is much less affected by such asymmetries.[5] Consequently, the rounding errors δ_{ij} in the spatial domain for the trunc quantizer are wrapped Gaussians with non-zero means, which has an effect on the accuracy of the LRT (not shown in this thesis) and, apparently, also on the CNN detectors.

Next, we investigate what happens when there is a mismatch between the quantizer used to obtain the original cover JPEG and the quantizer used by the steganalyst for training their detectors. In Table 2.5, SQY-SRNet exhibits no loss of accuracy for mismatched quantizers at QF95, a noticeable loss for QF99, and a catastrophic loss for QF100. As explained in the paragraph above, this demonstrates the utility of the PLT when steganalyzing (lower quality) images compressed with quantizers not seen during training. For QFs 99–100, however, the distribution of $\boldsymbol{\delta}$ is quantizer-dependent, which implies the SQ errors are quantizer-dependent.

Since the JPEG quantizers can be distinguished quite accurately with machine-learning tools, we decided to address the performance loss simply by training on images obtained using both quantizers. As Table 2.6 portrays, training in this fashion resolves the problem with an unknown quantizer; the detection accuracies are now comparable to those of the detectors trained and tested on images obtained with matching quantizers (as shown in Tables 2.2 and 2.4). Overall, training on the `imread` decompressor generalizes slightly better than training on PIL.

[5] Also note that this effect of non-zero mean for $\varepsilon_{kl}^{(0)}$ is mitigated for lower qualities – the increased variance of $\varepsilon_{kl}^{(0)}$ makes the wrapped Gaussian uniform.

Table 2.6: Testing accuracy for SQY-SRNet trained on both round and trunc at QF 100. SQY-SRNet is tested on two sets: TST only quantized with round and TST only quantized with trunc.

Test quant.	Train decomp.	Test set decompressor				
		imread	float	int	convert	PIL
round	imread	.9719	.9545	.9690	.9548	.9735
	PIL	.9676	.9487	.9674	.9503	.9695
trunc	imread	.8829	.8590	.8857	.8634	.8818
	PIL	.8738	.8452	.8762	.8478	.8705

2.6 Estimating the Quantization Table

As mentioned earlier, the steganalyst must estimate the true quantization table $\mathbf{q}$ directly from the image under investigation since it is not provided in the decompressed JPEG. Ideally, and for the most general case, each quantization step should be estimated separately for each DCT mode k, l since JPEG images can have non-standard quantization tables. This problem belongs to the field of image forensics and is well studied [45, 14]. However, the exact quantization steps are not needed to apply the JCA because estimating the Q errors is an *easier* task compared to estimating the exact quantization steps. Instead, we need only find a table $\widehat{\mathbf{q}}$ such that the estimated Q errors $\widehat{\boldsymbol{\varepsilon}} \triangleq \mathrm{err}_{\widehat{\mathbf{q}}}(\mathbf{y})$ are close in distribution to the true Q errors $\boldsymbol{\varepsilon}$. In particular, it is enough to estimate a divisor of the true quantization step — the so-called "sufficient" steps defined in Appendix A.1. Additionally, indeterminable steps [45] that may occur for high frequencies k, l do not pose a problem for the JCA either. Both cases are explained and discussed in more detail in Appendix A.1.

A comprehensive study of the effects of incorrectly estimated quantization steps on the accuracy of machine learning based JCAs is well beyond the scope of this thesis mostly due to the enormous diversity of custom quantization tables in use today. Due to time limitations, we postpone such study to future work and limit ourselves to standard tables so we may estimate the QF instead of individual quantization steps. We propose a simple maximum likelihood estimator (MLE) and show that its estimation accuracy is high enough so the effects of estimating incorrect tables / steps on steganalysis can be ignored. The reader is referred to [14, 45] for further discussion on incorrectly estimated steps and for estimation techniques more powerful than the MLE proposed.

Given a collection of observed blocks $\mathcal{B}$ (after block elimination), we can estimate the standard quantization table by maximizing the log-likelihood over all qualities $QF \in \{1, \dots, 100\}$:

$$\widehat{\mathbf{q}} = \operatorname*{argmax}_{QF} \sum_{\mathbf{x} \in \mathcal{B}} \sum_{k,l=0}^{7} \log f_{y_{kl}} \left((\mathbf{D}\mathbf{x})_{kl} \right), \tag{2.6.1}$$

where $(\mathbf{D}\mathbf{x})_{kl} = y_{kl}$ denotes the kl^{th} non-rounded recompressed DCT coefficient for a block. From Eq. (2.1.7), the PDF of y_{kl} can be expressed as

$$f_{y_{kl}}(u) = \sum_{n \in \mathbb{Z}} \frac{\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})}{\sqrt{2\pi s_{kl}}} \exp\left(-\frac{(u - nq_{kl})^2}{2s_{kl}} \right), \tag{2.6.2}$$

where $\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})$ is the prior probability that $y_{kl}^{(0)}$ had quantized to nq_{kl}. Each step q_{kl} is computed as per the JPEG standard for every quality factor. In practice, for each

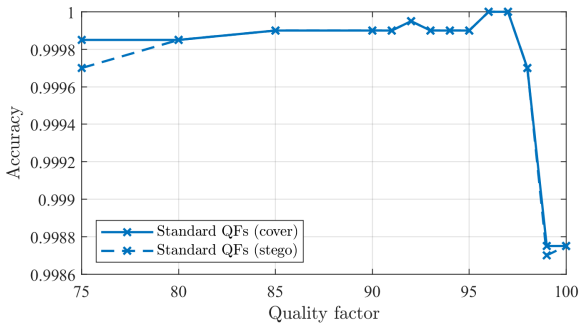

Figure 2.6.1: The ratio of images in BOSSBOWS2 whose quality factors were correctly estimated from covers (solid) and stegos (dashed) embedded using MiPOD at 0.01 bpp.

mode (k, l) we must estimate $\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})$ using a quantity $\widehat{P}_{kl}(nq_{kl})$ derived from the decompressed JPEG itself. For simplicity, if $|nq_{kl}| \leq M_{kl}$, we set

$$\widehat{P}_{kl}(nq_{kl}) = 1/(2M_{kl} + 1) \tag{2.6.3}$$

and $\widehat{P}_{kl}(nq_{kl}) = 0$ otherwise where $M_{kl} = \max_{\mathbf{x}\in\mathcal{B}} |\widetilde{y}_{kl}|$ is the maximum realization of $|\widetilde{y}_{kl}|$ attained across all blocks. Figure 2.6.1 shows the accuracy of estimating the correct QF from cover (solid line) and stego (dashed line) images. The authors deem this accuracy to be high enough to have a minimal effect on steganography detection in practice.

2.7 Conclusions

This thesis revisits the JPEG Compatibility Attack in light of the most recent advancements in steganalysis as well as steganography. The focus is on detection of modern content-adaptive embedding schemes and high quality factors when previous state-of-the-art methods experience computational complexity issues and loss of accuracy. Close attention is paid to the robustness of the proposed detectors to JPEG compressors and DCT coefficient quantizers. To better understand the observed trends in accuracy of various implementations of the JCA with respect to the quality factor and the effects of different JPEG quantizers, the authors derived a likelihood ratio test under mild modeling assumptions.

To summarize, the best detector was SQY-SRNet, a two-channel SRNet trained on the image and its SQ error. It exhibited a markedly better accuracy than previous art especially for high JPEG qualities and small payloads. Since the DCT quantizer used for the cover JPEG image and the decompressor are not available to the steganalyst to build the training datasets, this thesis includes a comprehensive study of the robustness of the SQY-SRNet with respect to these unknowns. We found that training SQY-SRNet on images obtained using both DCT quantizers and using Matlab's `imread` for decompression gave the best generalized results. This detector enjoys a similar level of accuracy as the clairvoyant detectors informed by and trained on the right combination of cover JPEG quantizer and decompressor.

Our future effort will be directed towards extending the JCA to color images and to make it robust to errors when estimating custom quantization tables.

Chapter 3

Alternative Formulations

In this chapter, we introduce several formulations of the JCA derived from the original approach discovered in [14]. The purpose of this chapter is to bring insight into different perspectives of the JCA, not to yield state-of-the-art performance. In fact, most methods in this chapter are computationally infeasible in practice. Hence, we maintain a high-level point-of-view in this chapter, refraining from implementation details and extensive experiments. The reader is referred to Chapter 2 for the development of feasible, statistically-based methods.

The original version of the JCA [14] was inspired by the fact that most blocks of pixels cannot possibly result from JPEG compression and decompression. This is inherently due to JPEG compression being a many-to-one mapping. The approach is formulated as a tree search over the space of quantized DCT blocks to verify if the block of pixels $\mathbf{x}$ could have resulted from a decompression. The search begins with $[\mathbf{y}]_{\mathbf{q}}$, the closest point to $\mathbf{y}$ of the form $\mathbf{q} \odot \mathbf{z}$, and branches out from there in the sense of increasing L^2 norm. Originally, a fixed upper bound on the L^2 norm is used in order to prune the search.

In Section 3.1, we recall the search algorithm in more detail. The tree search has a natural geometric interpretation, so we will formulate the search using tools different from those used in the original version. Then, we introduce new pruning strategies that further cut the computational expense. In Section 3.2, we formulate the JCA as a "softened" lattice point decision problem. We conclude this chapter in Section 3.3 with a discussion about the advantages and disadvantages of the approaches in terms of performance and practicality.

3.1 Brute-Force Search

3.1.1 Relevant geometric objects

Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be an invertible matrix. The lattice generated by a basis $\mathbf{B}$ is the set of all integer combinations of the columns of $\mathbf{B}$ written as

$$\mathscr{L}(\mathbf{B}) = \{\mathbf{B}\mathbf{z} : \mathbf{z} \in \mathbb{Z}^n\}. \tag{3.1.1}$$

Let $\mathbf{Q} \in \mathbb{R}^{64 \times 64}$ be the diagonal matrix formed by first rearranging the quantization table $\mathbf{q}$ as a column vector of length 64 (in the same manner depicted in Section 1.5 for blocks

of pixels) and then setting $Q_{jj} = q_j$ for $j = 1, \ldots, 64$. Note that the lattice $\mathscr{L}(\mathbf{Q})$ is isomorphic to the set of all dequantized DCT blocks $\{\mathbf{q} \odot \mathbf{z} : \mathbf{z} \in \mathbb{Z}^n\}$ (ignoring bounds on DCT coefficients for simplicity).

Denote the L^2 ball in $\mathbb{R}^n$ with radius R and centered at $\mathbf{x} \in \mathbb{R}^n$ by

$$S_n(\mathbf{x}, R) = \{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v} - \mathbf{x}\|_2 \leq R\}, \tag{3.1.2}$$

where $\|\cdot\|_2$ is the L^2 norm. If $\mathbf{x}$ is a block of pixels, we abuse notation slightly by implicitly converting it to a column vector before computing the norm.

Let $N(\mathbf{x}) = \{\mathbf{v} \in \mathbb{R}^{64} : \|\mathbf{v} - \mathbf{x}\|_\infty \leq 1/2\}$ denote the 1/2-neighborhood of $\mathbf{x}$ with respect to the sup norm $\|\cdot\|_\infty$. Again, if $\mathbf{x}$ is a block of pixels, treat it as a column vector for computations. Equivalently, $N(\mathbf{x})$ is the set of all points $\mathbf{v}$ for which $[\mathbf{v}] = \mathbf{x}$. Applying the DCT to (the points of) $N(\mathbf{x})$ can be viewed as rotating $N(\mathbf{x})$ about the origin $\mathbf{0} \in \mathbb{R}^{64}$. The resulting neighborhood is given by

$$\mathbf{D}N(\mathbf{x}) = \{\mathbf{D}\mathbf{v} : \|\mathbf{v} - \mathbf{x}\|_\infty \leq 1/2\} = \{\mathbf{v} : \|\mathbf{D}^{-1}\mathbf{v} - \mathbf{x}\|_\infty \leq 1/2\}. \tag{3.1.3}$$

3.1.2 Tree-pruning via L^2 norm

Consider the decompressed block of pixels $\mathbf{x}$ in the spatial domain. Let us further assume that $\mathbf{x}$ has no pixels saturated at 0 or 255 so we may ignore the effects of clipping. We call the point $\mathbf{z} \in \mathscr{L}(\mathbf{Q})$ a candidate for $\widetilde{\mathbf{y}}^{(0)}$ if it satisfies

$$[\mathbf{D}^{-1}\mathbf{z}] = \mathbf{x}, \tag{3.1.4}$$

meaning that $\mathbf{x}$ could have originated from the dequantized DCT coefficients $\mathbf{z}$. If there is at least one candidate for $\widetilde{\mathbf{y}}^{(0)}$, then the block $\mathbf{x}$ is JPEG compatible. Otherwise, $\mathbf{x}$ is incompatible.

We can search for candidates as follows. Due to rounding, we have the bound $|\delta_{ij}| = |x_{ij}^{(0)} - x_{ij}| \leq 1/2$ for all $0 \leq i, j, \leq 7$. We can deduce an upper bound in terms of the L^2 norm $\|\boldsymbol{\delta}\|_2^2 \leq 16$. By the orthonormality of the DCT (or Parseval's theorem), the DCT is an isometry with respect to L^2 norm which implies

$$\|\boldsymbol{\delta}\|_2^2 = \|\mathbf{D}\boldsymbol{\delta}\|_2^2 = \|\mathbf{D}\widetilde{\mathbf{x}}^{(0)} - \mathbf{D}\mathbf{x}\|_2^2 = \|\widetilde{\mathbf{y}}^{(0)} - \mathbf{y}\|_2^2 \leq 16. \tag{3.1.5}$$

We can limit our search space to points that satisfy the necessary condition given by Eq. (3.1.5). Hence, we must check if there is at least one $\mathbf{z} \in S_{64}(\mathbf{y}, 4) \cap \mathscr{L}(\mathbf{Q})$ that satisfies Eq. (3.1.4) to determine the JPEG compatibility of $\mathbf{x}$. Implementation details for searching through $S_{64}(\mathbf{y}, 4) \cap \mathscr{L}(\mathbf{Q})$ (e.g. ordering candidates by their distance from $\mathbf{y}$) can be found in [14].

Assuming the image is indeed a decompressed JPEG, the brute-force search guarantees a zero false alarm probability, $P_{\text{FA}} = 0$. Of course, there is a non-zero probability of missed detection since pixels could be modified so that the block remains JPEG compatible. As a pathological example, Alice could synchronize embedding changes so that pixels in an 8×8 block all simultaneously change in the same direction or do not change at all. This embedding scheme would completely circumvent the JCA (but of course would be detectable by other means).

3.1.3 Bounding box strategy

We can also limit the search space using a bounding box around $\mathbf{y}$. The bounding box is completely determined by the quantities

$$b_{kl}^{\text{upper}} = \sup\{v_{kl} : \mathbf{v} \in \mathbf{D}N(\mathbf{x})\}, \tag{3.1.6}$$

$$b_{kl}^{\text{lower}} = \inf\{v_{kl} : \mathbf{v} \in \mathbf{D}N(\mathbf{x})\}, \tag{3.1.7}$$

which are the upper and lower bounds obtained when projecting the rotated cube $\mathbf{D}N(\mathbf{x})$ onto the kl^{th} component, respectively. So when performing the search, we can also check if $\mathbf{z} \in S_{64}(\mathbf{y}, 4) \cap \mathscr{L}(\mathbf{Q})$ satisfies $b_{kl}^{\text{lower}} \leq z_{kl} \leq b_{kl}^{\text{upper}}$ for all k, l.

We can explicitly compute b_{kl}^{upper} as follows

$$\begin{aligned} b_{kl}^{\text{upper}} &= \max_{\mathbf{v} \in N(\mathbf{x})} \text{DCT}_{kl}(\mathbf{v}) \\ &= \text{DCT}_{kl}(\mathbf{x}) + \max_{\|\mathbf{u}\|_\infty \leq 1/2} \text{DCT}_{kl}(\mathbf{u}) \\ &= y_{kl} + \max_{\|\mathbf{u}\|_\infty \leq 1/2} \sum_{i,j=0}^{7} f_{kl}^{ij} u_{ij} \\ &= y_{kl} + \frac{1}{2} \sum_{i,j=0}^{7} |f_{kl}^{ij}|, \end{aligned} \tag{3.1.8}$$

since we get a maximum when $u_{kl} = 1/2$ for $f_{kl}^{ij} \geq 0$ and $u_{kl} = -1/2$ otherwise. By a similar computation, $b_{kl}^{\text{lower}} = y_{kl} - \frac{1}{2}\sum_{i,j=0}^{7} |f_{kl}^{ij}|$. It also follows that each face of the bounding box intersects a corner of the rotated cube $\mathbf{D}N(\mathbf{x})$.

3.1.4 Concentration of measure

Searching over the entire ball $S_{64}(\mathbf{D}\mathbf{x}, 4)$ turns out to be incredibly inefficient since most of its volume is concentrated near its boundary and outside of $\mathbf{D}N(\mathbf{x})$. We can amend our search for candidate points by adopting a probabilistic mindset. In particular, we can cut the search complexity by reducing the radius of $S_{64}(\mathbf{y}, 4)$ at the cost of permitting a non-zero P_{FA}.

We recall a result from high-dimensional probability theory [18, 11]. Consider a random vector $\mathbf{x} \in \mathbb{R}^n$ following the uniform distribution on the n-cube $[-1/2, 1/2)^n$; that is, the coordinates are i.i.d. and $x_j \sim \mathcal{U}[-1/2, 1/2)$ for $j = 1, \ldots, n$. Let $\mu = \mathbb{E}[x_j^2] = 1/12$ and $\sigma^2 = \mathbb{E}[x_j^4] - \mu^2 = 1/180$. By the weak law of large numbers, $\frac{1}{n}\|\mathbf{x}\|_2^2 \to \mu$ in probability as $n \to \infty$. Loosely speaking, the length of $\mathbf{x}$ should be very close to $\sqrt{n\mu}$ with high probability. In particular, consider the shell of radius $\sqrt{n\mu}$ and thickness parameter $a > 0$:

$$\begin{aligned} A_n(a) &= \left\{\mathbf{v} \in \mathbb{R}^n : n\mu - \sqrt{n\sigma^2}a < \|\mathbf{v}\|_2^2 < n\mu + \sqrt{n\sigma^2}a\right\} \\ &= \left\{\mathbf{v} \in \mathbb{R}^n : -a < \frac{\|\mathbf{v}\|_2^2 - n\mu}{\sqrt{n\sigma^2}} < a\right\}. \end{aligned} \tag{3.1.9}$$

The CLT implies

$$\text{Vol}(A_n(a) \cap [-1/2, 1/2)^n) = \mathbb{P}(\mathbf{x} \in A_n(a)) \to \int_{-a}^{a} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} \, \mathrm{d}t \tag{3.1.10}$$

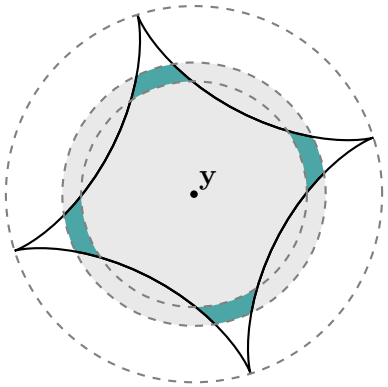

Figure 3.1.1: Concentration of measure in $\mathbf{D}N(\mathbf{x})$. The largest circle represents the L^2 ball $S_{64}(\mathbf{y}, 4)$ of radius 4. The region between the smaller circles represents the shell $A_{64}(a)$. The majority of mass in $\mathbf{D}N(\mathbf{x})$ is located within the intersection $A_{64}(a) \cap \mathbf{D}N(\mathbf{x})$, shaded in green. The search space is limited to $S_{64}(\mathbf{y}, 4\sqrt{1/3} + a)$, shaded in gray.

as $n \to \infty$, where Vol is the Lebesgue measure. Indeed, this result says that most of the mass of the n-cube concentrates within a thin shell of radius $\sqrt{n\mu}$, where the parameter a can be set large enough so that the probability in Eq. (3.1.10) is arbitrarily close to 1.

Applying Eq. (3.1.10) to the JCA, we can reduce the L^2 bound so that we search for candidates within $S_{64}(\mathbf{y}, 4\sqrt{1/3} + a)$ where a is chosen to satisfy a desired $P_{\text{FA}} > 0$. Figure 3.1.1 portrays the concentration phenomenon. We represent $\mathbf{D}N(\mathbf{x})$ as star-shaped because in high dimensions, the distance from the center $\mathbf{y}$ to any corner is much larger than the distance from $\mathbf{y}$ to the center of a face.

3.2 Softened Decision Problem

To determine the compatibility of $\mathbf{x}$, we need to check if $\mathbf{D}N(\mathbf{x}) \cap \mathscr{L}(\mathbf{Q})$ is non-empty. Asking about the existence of a lattice point in $\mathbf{D}N(\mathbf{x})$ is a discrete binary question. In fact, this decision problem about lattice points in a convex body is well-studied in the fields of the geometry of numbers and combinatorial geometry [17]. In this section, however, we introduce a setup that yields a "softened" or continuous answer to the binary question.

Consider the function $h : \mathbb{R}^{64} \to [0, \infty)$ defined by

$$h(\mathbf{v}) = \prod_{k,l=0}^{7} \cos^{2n}(\pi v_{kl}/q_{kl}). \tag{3.2.1}$$

We can interpret $h(\mathbf{v})$ as an electric field potential supposing the points of $\mathscr{L}(\mathbf{Q})$ are charged particles or alternatively, a function that quantifies an amount of heat that radiates from the lattice points. For large $n \gg 0$, the heat is highly concentrated near the lattice points; that is, if $\mathbf{v}$ is near a lattice point $h(\mathbf{v}) \approx 1$ otherwise $h(\mathbf{v}) \approx 0$. Consider the integral expression

$$L(\mathbf{x}) = \int_{\mathbf{D}N(\mathbf{x})} h(\mathbf{v})\mathrm{d}\mathbf{v} = \int_{N(\mathbf{x})} h(\mathbf{D}\mathbf{v})|\det(\mathbf{D})|\mathrm{d}\mathbf{v} = \int_{N(\mathbf{x})} h(\mathbf{D}\mathbf{v})\mathrm{d}\mathbf{v}, \tag{3.2.2}$$

where a change of variables is applies since the bounds of integration for the region $N(\mathbf{x})$ are much easier to work with. Note that the determinant of the Jacobian satisfies $|\det(\mathbf{D})| = 1$ due to the linearity and orthonormality of $\mathbf{D}$. Observe that $L(\mathbf{x})$ essentially counts the number of lattice points contained in $\mathbf{D}N(\mathbf{x})$ when $n \gg 0$. For the JCA, $L(\mathbf{x})$ can be used to detect whether or not the region contains a lattice point. Note that $L(\mathbf{x})$ can be integrated by using Euler's formula for complex exponentials, multiplying out the product, and applying the multinomial theorem. Ultimately, the integral resolves to a combinatorial expression. However, the explicit form is omitted since the result does not yield much insight in terms of algorithmic efficiency. Another possibility for the integrand of Eq. (3.2.2) is the function known as the Dirac comb function

$$C(\mathbf{v}) = \prod_{k,l=0}^{7} \sum_{n\in\mathbb{N}} \delta(v_{kl} - nq_{kl}), \tag{3.2.3}$$

where δ is the Dirac delta function.

3.3 Conclusions

It is clear that the search can catch steganography even if only one pixel change was made in the entire image. However, this method comes with several disadvantages. If all blocks are deemed incompatible, then it is likely that the image was not a decompressed JPEG to begin with, the DCT blocks are misaligned, e.g., due to cropping, or there is a mismatch between the JPEG compressors of the steganographer and steganalyst. Along with performing an expensive search (for high qualities) for each block, the steganalyst would need to check all 64 alignments and exhaustively check different JPEG compressors which is infeasible at scale.

Appendix A

Additional Results

A.1 Details on Estimating Quantization Steps

In this appendix, we explain why the steganalyst only needs to estimate sufficient steps for the JCA to apply as they provide approximately the same Q errors. We also touch upon indeterminable steps.

Suppose q_{kl} is the true quantization step, and let $f_{\widehat{\varepsilon}_{kl}}$ and $f_{\varepsilon_{kl}}$ denote the PDFs of the estimated Q error $\widehat{\varepsilon}_{kl}$ and true Q error ε_{kl}, respectively. We say an estimated quantization step $\widehat{q}_{kl}$ is *sufficient* if 1) $\widehat{q}_{kl} = q_{kl}$, or 2) $q_{kl} > \widehat{q}_{kl} \geq 2$ and $\widehat{q}_{kl}$ divides q_{kl}.

Proposition A.1. *If $\widehat{q}_{kl}$ is sufficient, then $|f_{\widehat{\varepsilon}_{kl}}(u) - f_{\varepsilon_{kl}}(u)| \leq C \doteq 3.43 \times 10^{-3}$ for all $u \in \mathbb{R}$.*

Informally, Proposition A.1 gives a sufficient condition under which $f_{\widehat{\varepsilon}_{kl}}(u) \approx f_{\varepsilon_{kl}}(u)$ (meaning "approximately equal") within some negligibly small uniform error C. The proposition is trivial to prove under the condition $\widehat{q}_{kl} = q_{kl}$, so we turn to the case $q_{kl} > \widehat{q}_{kl} \geq 2$ and $\widehat{q}_{kl}$ divides q_{kl}. The density $f_{\widehat{\varepsilon}_{kl}}$ is obtained by wrapping $f_{y_{kl}}$ (2.6.2) onto a circle of circumference $\widehat{q}_{kl}$: $f_{\widehat{\varepsilon}_{kl}}(u) = \sum_{m\in\mathbb{Z}} f_{y_{kl}}(u + m\widehat{q}_{kl})$ for $u \in [-\widehat{q}_{kl}/2, \widehat{q}_{kl}/2)$ and $f_{\widehat{\varepsilon}_{kl}}(u) = 0$ otherwise.

When $|u| \geq \widehat{q}_{kl}/2$, observe that $f_{\varepsilon_{kl}}(u) \approx 0 = f_{\widehat{\varepsilon}_{kl}}(u)$.[1] In particular, $|f_{\widehat{\varepsilon}_{kl}}(u) - f_{\varepsilon_{kl}}(u)| = f_{\varepsilon_{kl}}(u) \leq C$ by direct evaluation of the maximum.[2]

For $u \in [-\widehat{q}_{kl}/2, \widehat{q}_{kl}/2)$, observe that the Gaussian terms in $f_{\widehat{\varepsilon}_{kl}}$ are offset by integer multiples of $\widehat{q}_{kl}$ because

$$m\widehat{q}_{kl} - nq_{k\ell} = m\widehat{q}_{kl} - nj\widehat{q}_{kl} = (m - nj)\widehat{q}_{kl}, \tag{A.1.1}$$

for some $j \in \mathbb{Z}_{>0}$. By swapping the sums in $f_{\widehat{\varepsilon}_{kl}}$, we can re-index the sum over m according to Eq. (A.1.1) to produce

$$\begin{aligned} f_{\widehat{\varepsilon}_{kl}}(u) &= \sum_{n\in\mathbb{Z}} \frac{\mathbb{P}(\widetilde{y}^{(0)}_{kl} = nq_{kl})}{\sqrt{2\pi s_{kl}}} \sum_{m\in\mathbb{Z}} \exp\left(-\frac{(u + m\widehat{q}_{kl})^2}{2s_{kl}}\right) \\ &= \frac{1}{\sqrt{2\pi s_{kl}}} \sum_{m\in\mathbb{Z}} \exp\left(-\frac{(u + m\widehat{q}_{kl})^2}{2s_{kl}}\right), \end{aligned} \tag{A.1.2}$$

[1] This is due to the fact that $s_{kl} \leq 1/12$ and $q_{kl} > \widehat{q}_{kl} \geq 2$.

[2] $f_{\varepsilon_{kl}}(u)$ is maximized when $|u| = 1$, $\widehat{q}_{kl} = 2$, $q_{kl} = 4$, $s_{kl} = 1/12$.

for $u \in [-\widehat{q}_{kl}/2, \widehat{q}_{kl}/2)$. The last line in Eq. (A.1.2) follows from $\sum_{n\in\mathbb{Z}} \mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl}) = 1$. Observe that $|f_{\widehat{\varepsilon}_{kl}}(u) - f_{\varepsilon_{kl}}(u)|$ is upper bounded by $g(u; 0, s_{kl}, \widehat{q}_{kl})$ without the $n = 0$ term which has a maximum of C when $u \in [-\widehat{q}_{kl}/2, \widehat{q}_{kl}/2)$. Thus, we get $f_{\widehat{\varepsilon}_{kl}}(u) \approx f_{\varepsilon_{kl}}(u)$, proving Proposition A.1 as desired.[3]

In practice, there is another (content-dependent) sufficient condition that commonly holds for lower qualities: $\widehat{q}_{kl}, q_{kl} \geq 2$ and the DCTs y_{kl} are contained within the interval $[-1, 1)$ across all sampled blocks. This condition is known as the "indeterminable" case in [45] and is a point of failure for many quantization step estimation methods. However, this case benefits the steganalyst since $\mathrm{err}_{\widehat{q}_{kl}}(y_{kl}) = y_{kl} = \mathrm{err}_{q_{kl}}(y_{kl})$ for any chosen step $\widehat{q}_{kl} \geq 2$.

Observe that Proposition A.1 holds when either the round or the trunc quantizer is used for the initial JPEG compression; the differences in quantization bins only affect the values of $\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})$ and s_{kl}. Also note that the proposition considered only cover images. When estimating the steps from stego images, the variance s_{kl} is replaced with $s_{kl} + r_{kl}$, which has a negligible effect on the accuracy of the Q error for the most relevant case of small payloads $r_{kl} \ll 1$. Finally, quantization step estimation methods such as the one proposed in [45] will often select a divisor of the true step when wrong, which tells us that steps are commonly sufficient in practice.

A.2 Computing Moments of the LRT

In Section 2.2.1, we derived the normalized LRT $\Lambda(\mathcal{B})$ given in Eq. (2.2.5). Computing $\Lambda(\mathcal{B})$ requires knowledge of the mean and variance of $\mathcal{L}(\mathcal{B})$ under $\mathcal{H}_0$ which, due to Assumption 2.4 (joint independence of ε_{kl}), can be written as

$$\begin{aligned}
\mathbb{E}[\mathcal{L}(\mathcal{B})] &= \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \mathbb{E}\left[\log \frac{g(\varepsilon_{kl}; 0, s_{kl} + r_{kl}, \widehat{q}_{kl})}{g(\varepsilon_{kl}; 0, s_{kl}, \widehat{q}_{kl})}\right], \\
\mathrm{Var}[\mathcal{L}(\mathcal{B})] &= \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \mathrm{Var}\left[\log \frac{g(\varepsilon_{kl}; 0, s_{kl} + r_{kl}, \widehat{q}_{kl})}{g(\varepsilon_{kl}; 0, s_{kl}, \widehat{q}_{kl})}\right].
\end{aligned} \tag{A.2.1}$$

The mean and variance could be computed via numerical integration alone, however the number of numerical integrations needed for the experiments in Section 2.4 scales with the size of the images, dataset, and number of quality factors.

In this section, we derive an analytic expression that approximates the mean to arbitrary precision and substantially reduces the computation burden. We assume the $r_{kl} \ll 1$ are small for the relevant case of small payloads and the quantization steps have been correctly estimated $\widehat{q}_{kl} = q_{kl}$. To simplify notation, the following discussion considers a fixed JPEG mode so that we may remove the subscript kl.

[3] $g(u; 0, s_{kl}, \widehat{q}_{kl})$ without the $n = 0$ term is maximized at $|u| = 1$, $\widehat{q}_{kl} = 2$, $s_{kl} = 1/12$.

A.2.1 Analytic form

In this section, we can make use of the following representations of the (mean zero) wrapped Gaussian

$$\begin{aligned} g(x;0,s,q) &= \frac{1}{\sqrt{2\pi s}} \sum_{n\in\mathbb{Z}} \exp\left(-\frac{(x+qn)^2}{2s}\right) && \text{(Gauss)} \\ &= \frac{1}{q} \sum_{n\in\mathbb{Z}} p_s^{n^2/2} e^{i\phi n x} && \text{(Fourier)} \\ &= \frac{1}{q} \prod_{n=1}^{\infty} (1-p_s^n)(1+p_s^{n-\frac{1}{2}} e^{i\phi x})(1+p_s^{n-\frac{1}{2}} e^{-i\phi x}) && \text{(Jacobi)} \end{aligned} \tag{A.2.2}$$

where $p_s = \exp\left(-s\phi^2\right)$, $\phi = 2\pi/q$, $x \in [-q/2, q/2)$, and i is the imaginary unit. Here, we specifically refer to the original expression given in Eq. (1.4.1) as the Gauss representation. The Fourier representation expresses the wrapped Gaussian in terms of its characteristic function (via the Fourier transform) [37]. The Jacobi representation is derived from applying the so-called Jacobi triple product identity [20] to the Fourier representation.

First, we discuss some preliminary computations for better readability. Let $m, n \in \mathbb{Z}$ be integers. Using the Taylor expansion and properties of log, we have the following identity for $a > 0$

$$\log[(1+ae^{i\phi x})(1+ae^{-i\phi x})] = \sum_{m=1}^{\infty} \frac{1}{m}(-1)^{m+1} a^m (e^{im\phi x} + e^{-im\phi x}). \tag{A.2.3}$$

The complex exponentials $e^{im\phi x}$ are orthogonal on the interval $x \in [-q/2, q/2)$. That is to say,

$$\int_{-q/2}^{q/2} e^{in\phi x} e^{-im\phi x} \mathrm{d}x = \int_{-q/2}^{q/2} e^{i(n-m)\phi x} \mathrm{d}x = q\delta_{n,m} = \begin{cases} q & n = m \\ 0 & n \neq m \end{cases}, \tag{A.2.4}$$

where $\delta_{n,m}$ is the Kronecker delta function. Hence, it follows that

$$\sum_{n\in\mathbb{Z}} p_s^{n^2/2} \int_{-q/2}^{q/2} e^{i\phi n x} \mathrm{d}x = \sum_{n\in\mathbb{Z}} p_s^{n^2/2} q\delta_{n,0} = 1, \tag{A.2.5}$$

where only the $n = 0$ term survives. Similarly, we have

$$\begin{aligned} \sum_{n\in\mathbb{Z}} p_s^{n^2/2} \int_{-q/2}^{q/2} e^{i\phi n x}(e^{im\phi x} + e^{-im\phi x}) \mathrm{d}x &= \sum_{n\in\mathbb{Z}} p_s^{n^2/2} q(\delta_{n,-m} + \delta_{n,m}) \\ &= q p_s^{(-m)^2/2} + q p_s^{m^2/2} \\ &= 2q p_s^{m^2/2}, \end{aligned} \tag{A.2.6}$$

where only the $n = \pm m$ terms survive.

Now, consider the random variable $X_t \sim \mathcal{N}_{\mathcal{W}}(0, t, q)$ parametrized by $t > 0$. Utilizing the Fourier and Jacobi representations along with the previous results, we compute the

(negative) cross-entropy between X_s and X_t as follows

$$\begin{aligned}
H(s,t) &= \int_{-q/2}^{q/2} \log\left(g(x;0,t,q)\right) g(x;0,s,q)\mathrm{d}x \\
&= \int_{-q/2}^{q/2} \log\left(\frac{1}{q}\prod_{n=1}^{\infty}(1-p_t^n)(1+p_t^{n-\frac{1}{2}}e^{i\phi x})(1+p_t^{n-\frac{1}{2}}e^{-i\phi x})\right)\frac{1}{q}\sum_{n\in\mathbb{Z}} p_s^{n^2/2}e^{i\phi n x}\mathrm{d}x \\
&= \int_{-q/2}^{q/2}\left[\log\left(\frac{1}{q}\prod_{n=1}^{\infty}(1-p_t^n)\right)+\sum_{n=1}^{\infty}\log[(1+p_t^{n-\frac{1}{2}}e^{i\phi x})(1+p_t^{n-\frac{1}{2}}e^{-i\phi x}))]\right] \\
&\qquad \times \frac{1}{q}\sum_{\ell\in\mathbb{Z}} p_s^{\ell^2/2}e^{i\phi\ell x}\mathrm{d}x \\
&= \frac{1}{q}\log\left(\frac{1}{q}\prod_{n=1}^{\infty}(1-p_t^n)\right)\sum_{\ell\in\mathbb{Z}} p_s^{\ell^2/2}\int_{-q/2}^{q/2} e^{i\phi\ell x}\mathrm{d}x \\
&\qquad + \frac{1}{q}\sum_{n=1}^{\infty}\sum_{m=1}^{\infty}\frac{1}{m}(-1)^{m+1}p_t^{(n-\frac{1}{2})m}\sum_{\ell\in\mathbb{Z}} p_s^{\ell^2/2}\int_{-q/2}^{q/2} e^{i\phi\ell x}(e^{im\phi x}+e^{-im\phi x})\mathrm{d}x \\
&= \sum_{n=1}^{\infty}\log(1-p_t^n) - \log q + 2\sum_{n=1}^{\infty}\sum_{m=1}^{\infty}\frac{1}{m}(-1)^{m+1}p_t^{(n-\frac{1}{2})m}p_s^{m^2/2}. \qquad \text{(A.2.7)}
\end{aligned}$$

To reduce computation complexity, we further simplify the result by making use of the geometric series formula

$$\begin{aligned}
H(s,t) &= \sum_{n=1}^{\infty}\sum_{m=1}^{\infty}\frac{-1}{m}p_t^{mn} - \log q + 2\sum_{m=1}^{\infty}\frac{1}{m}(-1)^{m+1}p_s^{m^2/2}\sum_{n=1}^{\infty}p_t^{(n-\frac{1}{2})m} \\
&= \sum_{m=1}^{\infty}\frac{-1}{m}\left(\frac{p_t^m}{1-p_t^m}\right) - \log q + 2\sum_{m=1}^{\infty}\frac{1}{m}(-1)^{m+1}p_s^{m^2/2}\left(\frac{p_t^{m/2}}{1-p_t^m}\right) \\
&= \sum_{m=1}^{\infty}\frac{1}{m}\left[(-1)^{m+1}2p_s^{m^2/2}p_t^{-m/2}-1\right]\left(\frac{p_t^m}{1-p_t^m}\right) - \log q. \qquad \text{(A.2.8)}
\end{aligned}$$

Therefore, the expectation has the form

$$\mathbb{E}_{\mathcal{H}_0}\left[\log\frac{g(\varepsilon;0,s+r,q)}{g(\varepsilon;0,s,q)}\right] = H(s,s+r) - H(s,s). \qquad \text{(A.2.9)}$$

A.2.2 The case of $q \geq 2$ and $s \approx 1/12$

In the case $q \geq 2$ and $s \approx 1/12$, we can use the $n = 0$ approximation of the wrapped Gaussian to simplify the integrands in Eq. (A.2.1) which gives us

$$\log\frac{g(x;0,s+r,q)}{g(x;0,s,q)} \approx \frac{1}{2}\log\frac{s}{s+r} + \frac{r}{2s(s+r)}x^2. \qquad \text{(A.2.10)}$$

The expectation (under $\mathcal{H}_0$) is thus

$$\mathbb{E}\left[\log\frac{g(\varepsilon;0,s+r,q)}{g(\varepsilon;0,s,q)}\right] \approx \frac{1}{2}\log\frac{s}{s+r} + \frac{r}{2(s+r)} \qquad \text{(A.2.11)}$$

since $\mathbb{E}[\varepsilon^2] \approx s$ for $q \geq 2$. The variance is

$$\begin{aligned}
\operatorname{Var}\left[\log \frac{g(\varepsilon; 0, s+r, q)}{g(\varepsilon; 0, s, q)}\right] &\approx \mathbb{E}\left[\left(\frac{r}{2s(s+r)}\varepsilon^2 - \frac{r}{2(s+r)}\right)^2\right] \\
&= \frac{r^2}{4(s+r)^2}\mathbb{E}\left[\left(\frac{1}{s}\varepsilon^2 - 1\right)^2\right] \\
&= \frac{r^2}{4(s+r)^2}\mathbb{E}\left[\frac{1}{s^2}\varepsilon^4 - \frac{2}{s}\varepsilon^2 + 1\right] \\
&= \frac{r^2}{2(s+r)^2}. \qquad \text{(A.2.12)}
\end{aligned}$$

since $\mathbb{E}[\varepsilon^4] \approx 3s^2$ for $q > 1$.

A.3 Trends for SQY-SRNet and RRH

Tables A.1 and A.2 show the complete performance trends for SQY-SRNet and the prior art discussed in Section 2.4.2.

Table A.1: Testing accuracy for SQY-SRNet. (De)compressed with Matlab's imwrite.

	Payload	QF										
	(bpp)	90	91	92	93	94	95	96	97	98	99	100
MiPOD	0.02	.9996	.9996	.9996	.9995	.9994	.9996	.9997	.9996	.9999	.9924	.9899
	0.01	.9992	.9997	.9994	.9993	.9994	.9996	.9995	.9994	.9994	.9856	.9721
	0.005	.9988	.9983	.9983	.9984	.9983	.9986	.9975	.9960	.9903	.9094	.8634
	0.002	.9918	.9905	.9916	.9874	.9856	.9791	.9747	.9587	.9231	.7512	.6891
HILL	0.02	.9998	.9997	.9997	.9997	.9996	.9997	.9998	.9998	.9998	.9981	.9982
	0.01	.9994	.9993	.9995	.9994	.9995	.9996	.9996	.9997	.9996	.9964	.9950
	0.005	.9975	.9981	.9990	.9981	.9986	.9983	.9989	.9985	.9968	.9783	.9698
	0.002	.9926	.9948	.9926	.9927	.9906	.9885	.9885	.9841	.9683	.8958	.8717
S-UNI	0.02	.9995	.9995	.9997	.9996	.9996	.9997	.9998	.9999	.9999	.9970	.9959
	0.01	.9994	.9995	.9997	.9994	.9996	.9997	.9990	.9997	.9996	.9934	.9918
	0.005	.9991	.9988	.9985	.9985	.9989	.9994	.9983	.9976	.9968	.9650	.9498
	0.002	.9934	.9923	.9910	.9921	.9908	.9895	.9831	.9721	.9603	.8511	.8109
WOW	0.02	.9997	.9997	.9997	.9998	.9997	.9996	.9998	.9995	.9998	.9990	.9981
	0.01	.9996	.9996	.9996	.9996	.9997	.9991	.9999	.9993	.9995	.9965	.9959
	0.005	.9987	.9991	.9985	.9988	.9983	.9990	.9986	.9987	.9972	.9804	.9725
	0.002	.9920	.9942	.9917	.9929	.9912	.9888	.9872	.9813	.9726	.8978	.8630

Table A.2: Testing accuracy for RRH. (De)compressed with Matlab's imwrite.

	Payload	QF										
	(bpp)	90	91	92	93	94	95	96	97	98	99	100
MiPOD	0.02	.9995	.9994	.9987	.9970	.9924	.9778	.9189	.8722	.9052	.9239	.9290
	0.01	.9993	.9998	.9991	.9984	.9826	.9035	.7778	.7114	.7172	.7543	.7577
	0.005	.9994	.9996	.9996	.9983	.9147	.7610	.6497	.6016	.5999	.6175	.6257
	0.002	.9939	.9942	.9982	.9519	.7247	.6157	.5597	.5370	.5342	.5390	.5438
HILL	0.02	.9906	.9903	.9891	.9763	.9440	.9087	.8974	.9242	.9676	.9796	.9651
	0.01	.9926	.9902	.9881	.9807	.9044	.8165	.7613	.7551	.8745	.8736	.8312
	0.005	.9898	.9881	.9886	.9725	.8263	.7052	.6508	.6301	.7240	.7341	.6874
	0.002	.9817	.9830	.9837	.9219	.6971	.5987	.5623	.5505	.5721	.5872	.5705
S-UNI	0.02	.9980	.9982	.9977	.9924	.9819	.9562	.9078	.8816	.9368	.9536	.9501
	0.01	.9984	.9979	.9961	.9939	.9598	.8744	.7776	.7303	.7930	.8142	.7964
	0.005	.9978	.9974	.9967	.9939	.8884	.7503	.6497	.6216	.6363	.6649	.6572
	0.002	.9930	.9931	.9965	.9540	.7306	.6137	.5680	.5472	.5488	.5614	.5575
WOW	0.02	.9932	.9929	.9911	.9754	.9448	.9127	.8919	.9210	.9665	.9787	.9677
	0.01	.9931	.9911	.9893	.9766	.9136	.8248	.7709	.7547	.8608	.8756	.8353
	0.005	.9905	.9898	.9899	.9766	.8440	.7208	.6526	.6268	.7070	.7260	.6873
	0.002	.9840	.9881	.9868	.9354	.7112	.6104	.5668	.5511	.5621	.5852	.5699

Bibliography

[1] S. Agarwal and H. Farid. Photo forensics from rounding artifacts. In C. Riess and F. Schirrmacher, editors, *The 8th ACM Workshop on Information Hiding and Multimedia Security*, Denver, Colorado, June 22–25, 2020. ACM Press.

[2] P. Bas, T. Filler, and T. Pevný. Break our steganographic system – the ins and outs of organizing BOSS. In T. Filler, T. Pevný, A. Ker, and S. Craver, editors, *Information Hiding, 13th International Conference*, volume 6958 of Lecture Notes in Computer Science, pages 59–70, Prague, Czech Republic, May 18–20, 2011.

[3] P. Bas and T. Furon. BOWS-2. http://bows2.ec-lille.fr, July 2007.

[4] R. Böhme. Weighted stego-image steganalysis for JPEG covers. In K. Solanki, K. Sullivan, and U. Madhow, editors, *Information Hiding, 10th International Workshop*, volume 5284 of Lecture Notes in Computer Science, pages 178–194, Santa Barbara, CA, June 19–21, 2007. Springer-Verlag, New York.

[5] R. Böhme. *Advanced Statistical Steganalysis*. Springer-Verlag, Berlin Heidelberg, 2010.

[6] M. Boroumand, M. Chen, and J. Fridrich. Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 14(5):1181–1193, May 2019.

[7] M. Boroumand and J. Fridrich. Synchronizing embedding changes in side-informed steganography. In *Proceedings IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics 2020*, San Francisco, CA, January 26–30 2020.

[8] J. Butora and J. Fridrich. Reverse JPEG compatibility attack. *IEEE Transactions on Information Forensics and Security*, 15:1444–1454, 2020.

[9] J. Butora and J. Fridrich. Steganography and its detection in JPEG images obtained with the "trunc" quantizer. In *Proceedings IEEE, International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, May 4–8, 2020.

[10] R. Cogranne. Selection-channel-aware reverse JPEG compatibility for highly reliable steganalysis of JPEG images. In *Proceedings IEEE, International Conference on Acoustics, Speech, and Signal Processing*, pages 2772–2776, Barcelona, Spain, May 4–8, 2020.

[11] Richard Durrett. *Probability: theory and examples, 5th edition*. Cambridge University Press, 2020.

[12] T. Filler, J. Judas, and J. Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Transactions on Information Forensics and Security*, 6(3):920–935, September 2011.

[13] J. Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.

[14] J. Fridrich, M. Goljan, and R. Du. Steganalysis based on JPEG compatibility. In A. G. Tescher, editor, *Special Session on Theoretical and Practical Issues in Digital Watermarking and Data Hiding, SPIE Multimedia Systems and Applications IV*, volume 4518, pages 275–280, Denver, CO, August 20–24, 2001.

[15] J. Fridrich and J. Kodovský. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, June 2011.

[16] M. Goljan and J. Fridrich. Cfa-aware features for steganalysis of color images. In A. Alattar and N. D. Memon, editors, *Proceedings SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics 2015*, volume 9409, San Francisco, CA, February 8–12, 2015.

[17] Jacob E. Goodman, Joseph ORourke, and Csaba D. Toth. *Handbook of discrete and computational geometry, third edition*. CRC Press, 2018.

[18] Olivier Guedon. Concentration phenomena in high dimensional geomrety. *ESAIM Proceedings*, 44:47–60, 2014.

[19] L. Guo, J. Ni, and Y. Q. Shi. Uniform embedding for efficient JPEG steganography. *IEEE Transactions on Information Forensics and Security*, 9(5):814–825, May 2014.

[20] Godfrey H. Hardy and Edward M. Wright. *An introduction to the theory of numbers (4. ed.)*. Oxford, at the Clarendon Press, 1960.

[21] V. Holub and J. Fridrich. Designing steganographic distortion using directional filters. In *Fourth IEEE International Workshop on Information Forensics and Security*, Tenerife, Spain, December 2–5, 2012.

[22] V. Holub and J. Fridrich. Low-complexity features for JPEG steganalysis using undecimated DCT. *IEEE Transactions on Information Forensics and Security*, 10(2):219–228, February 2015.

[23] V. Holub, J. Fridrich, and T. Denemark. Universal distortion design for steganography in an arbitrary domain. *EURASIP Journal on Information Security, Special Issue on Revised Selected Papers of the 1st ACM IH and MMS Workshop*, 2014:1, 2014.

[24] X. Hu, J. Ni, W. Su, and J. Huang. Model-based image steganography using asymmetric embedding scheme. *Journal of Electronic Imaging*, 27(4):1 – 7, 2018.

[25] F. Huang, W. Luo, J. Huang, and Y.-Q. Shi. Distortion function designing for JPEG steganography with uncompressed side-image. In W. Puech, M. Chaumont, J. Dittmann, and P. Campisi, editors, *1st ACM IH&MMSec. Workshop*, Montpellier, France, June 17–19, 2013.

[26] P. E. Jupp. A Poincaré limit theorem for wrapped probability distributions on compact symmetric spaces. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 95, pages 329–334. Cambridge University Press, 1984.

[27] S. M. Kay. *Fundamentals of Statistical Signal Processing, Volume II: Detection Theory*, volume II. Upper Saddle River, NJ: Prentice Hall, 1998.

[28] A. D. Ker and R. Böhme. Revisiting weighted stego-image steganalysis. In E. J. Delp, P. W. Wong, J. Dittmann, and N. D. Memon, editors, *Proceedings SPIE, Electronic Imaging, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, pages 5 1–17, San Jose, CA, January 27–31, 2008.

[29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, 2014. http://arxiv.org/abs/1412.6980.

[30] J. Kodovský and J. Fridrich. JPEG-compatibility steganalysis using block-histogram of recompression artifacts. In M. Kirchner and D. Ghosal, editors, *Information Hiding, 14th International Conference*, volume 7692 of Lecture Notes in Computer Science, pages 78–93, Berkeley, California, May 15–18, 2012.

[31] J. Kodovský, J. Fridrich, and V. Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, April 2012.

[32] G. Kurz, I. Gilitschenski, and U. D. Hanebeck. Efficient evaluation of the probability density function of a wrapped normal distribution. In *2014 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–5. IEEE, 2014.

[33] B. Li, M. Wang, and J. Huang. A new cost function for spatial image steganography. In *Proceedings IEEE, International Conference on Image Processing, ICIP*, Paris, France, October 27–30, 2014.

[34] B. Li, M. Wang, X. Li, S. Tan, and J. Huang. A strategy of clustering modification directions in spatial image steganography. *IEEE Transactions on Information Forensics and Security*, 10(9):1905–1917, September 2015.

[35] Bin Li, Tian-Tsong Ng, Xiaolong Li, Shunquan Tan, and Jiwu Huang. Revealing the trace of high-quality jpeg compression through quantization noise analysis. *IEEE Transactions on Information Forensics and Security*, 10(3):558–573, 2015.

[36] W. Luo, Y. Wang, and J. Huang. Security analysis on spatial ±1 steganography for JPEG decompressed images. *IEEE Signal Processing Letters*, 18(1):39–42, 2011.

[37] K. V. Mardia and P. E. Jupp. *Directional Statistics*. Wiley Series in Probability and Statistic. John Wiley & Sons, Inc., 1999.

[38] V. K. Nath, , and D. Hazarika. Comparison of generalized Gaussian and Cauchy distributions in modeling of dyadic rearranged 2D DCT coefficients. In *3rd National Conference on Emerging Trends and Applications in Computer Science (NCETACS)*, pages 89–92, March 2012.

[39] C. Pasquini and R. Böhme. Towards a theory of JPEG block convergence. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 550–554. IEEE, 2018.

[40] W. Pennebaker and J. Mitchell. *JPEG: Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1993.

[41] V. Sedighi, R. Cogranne, and J. Fridrich. Content-adaptive steganography by minimizing statistical detectability. *IEEE Transactions on Information Forensics and Security*, 11(2):221–234, 2016.

[42] G. J. Simmons. The prisoner's problem and the subliminal channel. In D. Chaum, editor, *Advances in Cryptology, CRYPTO '83*, pages 51–67, Santa Barbara, CA, August 22–24, 1983. New York: Plenum Press.

[43] L. N. Smith and N. Topin. Super-convergence: Very fast training of neural networks using large learning rates, 2018.

[44] A. Sripad and D. Snyder. A necessary and sufficient condition for quantization errors to be uniform and white. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(5):442–448, 1977.

[45] Thanh H. Thai, R. Cogranne, F. Retraint, and Thi-Ngoc-Canh Doan. JPEG quantization step estimation and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security*, 12(1):123–133, 2017.

[46] G. Xu. Deep convolutional neural network to detect J-UNIWARD. In M. Stamm, M. Kirchner, and S. Voloshynovskiy, editors, *The 5th ACM Workshop on Information Hiding and Multimedia Security*, Philadelphia, PA, June 20–22, 2017.

[47] G. Xu, H. Z. Wu, and Y. Q. Shi. Structural design of convolutional neural networks for steganalysis. *IEEE Signal Processing Letters*, 23(5):708–712, May 2016.

[48] J. Ye, J. Ni, and Y. Yi. Deep learning hierarchical representations for image steganalysis. *IEEE Transactions on Information Forensics and Security*, 12(11):2545–2557, November 2017.

[49] M. Yedroudj, M. Chaumont, and F. Comby. How to augment a small learning set for improving the performances of a CNN-based steganalyzer? In A. Alattar and N. D. Memon, editors, *Proceedings IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics 2018*, San Francisco, CA, January 29–February 1, 2018.

[50] M. Yedroudj, F. Comby, and M. Chaumont. Yedroudj-net: An efficient CNN for spatial steganalysis. In *IEEE ICASSP*, pages 2092–2096, Alberta, Canada, April 15–20, 2018.

[51] Y. Yousfi and J. Fridrich. An intriguing struggle of CNNs in JPEG steganalysis and the one-hot solution. *IEEE Signal Processing Letters*, 27:830–834, 2020.

[52] J. Zeng, S. Tan, B. Li, and J. Huang. Large-scale JPEG image steganalysis using hybrid deep-learning framework. *IEEE Transactions on Information Forensics and Security*, 13(5):1200–1214, May 2018.