Advancing the JPEG Compatibility Attack: Theory, Performance, Robustness, and Practice

Eli Dworetzky, Edgar Kaziakhmedov, and Jessica Fridrich
Binghamton University
Department of Electrical and Computer Engineering
Binghamton, NY 13850
{edworet1,ekaziak1,fridrich}@binghamton.edu

ABSTRACT

The JPEG compatibility attack is a steganalysis method for detecting messages embedded in the spatial representation of an image under the assumption that the cover image was a decompressed JPEG. This paper addresses a number of open problems in previous art, namely the lack of theoretical insight into how and why the attack works, low detection accuracy for high JPEG qualities, robustness to the JPEG compressor and DCT coefficient quantizer, and real-life performance evaluation. To explain the main mechanism responsible for detection and to understand the trends exhibited by heuristic detectors, we adopt a model of quantization errors of DCT coefficients in the recompressed image, and within a simplified setup, we analyze the behavior of the most powerful detector. Empowered by our analysis, we resolve the performance deficiencies using an SRNet trained on a two-channel input consisting of the image and its SQ error. This detector is compared with previous state of the art on four content-adaptive stego methods and for a wide range of payloads and quality factors. The last sections of this paper are devoted to studying robustness of this detector with respect to JPEG compressors, quantizers, and errors in estimating the JPEG quantization table. Finally, to demonstrate practical usability of this attack, we test our detector on stego images outputted by real steganographic tools available on the Internet.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Image manipulation**; **Neural networks**;

KEYWORDS

Steganography, steganalysis, JPEG, compatibility, robustness, rounding errors, deep learning, wrapped distributions

ACM Reference Format:
Eli Dworetzky, Edgar Kaziakhmedov, and Jessica Fridrich. 2023. Advancing the JPEG Compatibility Attack: Theory, Performance, Robustness, and Practice. In *Proceedings of the 2023 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '23), June 28–30, 2023, Chicago, IL.*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/XXXXXXX

1 INTRODUCTION

The JPEG Compatibility Attack (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.

The assumption that the cover was originally stored as JPEG is feasible as the vast majority of images are stored in the JPEG format. Two scenarios can make the JCA applicable, either due to action of the user or the stego program itself. A steganographer might supply a decompressed JPEG to the stego tool perhaps because it offers a larger embedding capacity than hiding directly in its JPEG form or because the data hiding tool cannot handle the JPEG format. In the second scenario, a stego tool accepts JPEG images on their input but hides in their decompressed form. To show how wide spread the second scenario is and to underline the relevance of our work, we downloaded from the Internet 45 stego tools available from various repositories. Out of these 45 tools, 42 accepted a JPEG cover image and 15 (or 36%) of these tools produced stego images in raster format that were all perfectly detected as stego with our implementation of the JCA (Section 10).

The attack was originally conceived in [12] 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 Warden, in practice the incompatibility of a block would also need to be verified w.r.t. 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 [22]. The detection accuracy of this attack is fairly robust w.r.t. 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 [27], 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 [23] counts the number of different pixels between the image and the recompressed-decompressed version in each 8×8 block. 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.

All forms of the JCA generally 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 goals of this paper.

In Section 2, we introduce the notation used throughout the paper and relevant background material from the field of directional statistics. Section 3 describes the compression pipeline within which the JCA operates, which includes the original compression, decompression and embedding, and recompression by the Warden. This pipeline is analyzed in Section 4 by imposing a model on the DCT coefficient quantization errors during the initial compression. In Section 5, this model allows us to derive the most powerful detector in the form of the likelihood ratio (LRT), which is subsequently used to obtain insight into the inner workings of the JCA and also explain the trends in detection accuracy observed for heuristic detectors in Section 6. Section 7 contrasts the detection of the LRT, our newly proposed CNN-based detectors, and previous art — for a wide range of JPEG quality factors, payloads, and embedding schemes. The remaining sections are devoted to studying important practical aspects of the JCA. Section 8 focuses on the JCA's robustness to various JPEG compressors and DCT quantizers. Since the JCA needs to estimate the quantization table of the original JPEG compression, in Section 9 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. In Section 10, we report how our JCA performed in real life conditions on existing stego tools. The paper is concluded in Section 11.

2 PRELIMINARIES

2.1 Notation

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 $\mathbf{x} \in \mathbb{R}^{m \times n}$ w.r.t. a matrix $\mathbf{q} \in \mathbb{R}^{m \times n}$ 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 $\mathrm{err}_{\mathbf{q}}(\mathbf{x}) \triangleq \mathbf{x} - [\mathbf{x}]_{\mathbf{q}}$.

2.2 Directional statistics

For any real-valued random variable X and positive integer q, the distribution of the quantization error $\mathrm{err}_q(X)$ is obtained by partitioning $\mathbb{R}$ into intervals of length q and applying the law of total probability. From the perspective of directional statistics [28] we can also obtain the distribution of $\mathrm{err}_q(X)$ 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 interval $[-q/2, q/2)$.[1] Seeing this connection between quantization error and wrapped distributions will allow us to use the following results during our study of the JCA.

When $X \sim \mathcal{N}(\mu, \sigma^2)$, the quantization error $\mathrm{err}_q(X)$ follows a wrapped Gaussian distribution $\mathcal{N}_{\mathcal{W}}(\mu, \sigma^2, q)$ whose PDF is

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

when $-q/2 \leq x < q/2$ 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 [9]. However, since the class of wrapped distributions is well studied and is the standard nomenclature in directional statistics [28], 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 [28]. For example, $g(x; 0, 1/12, q)$ is well-approximated by one term ($n = 0$) for $q \geq 2$ and three terms ($n = -1, 0, 1$) for $q = 1$. General bounds for the approximation error are found in [8, 24, 28].

Finally, we recall a fundamental asymptotic result known as Poincaré's Limit Theorem (PLT) [28] applied to quantization error. 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 $c \to \infty$. A generalized PLT is developed in [20] for wrapping a joint distribution onto a compact symmetric space. Here, we state a simplified version for wrapping distributions over $\mathbb{R}^m$ onto a m-torus via the map $\mathrm{err}_{\mathbf{q}}$ where $\mathbf{q} \in \mathbb{R}^m$.[2] If X is an absolutely continuous random vector on $\mathbb{R}^m$, then the distribution of $\mathrm{err}_{\mathbf{q}}(cX)$ tends to the uniform distribution on the set $\prod_{i=1}^{m} [-q_i/2, q_i/2)$ as $c \to \infty$.

[1] One can think of the interval $[-q/2, q/2)$ as having its end points glued together to form the circle (or a 1-torus).

[2] Loosely speaking, a m-torus is the hypercube $\prod_{i=1}^{m} [-q_i/2, q_i/2)$ with its opposing faces glued together (topologically) where $\prod$ denotes the cartesian product.

3 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 zig-zag fashion, losslessly compressed, and written into the JPEG file together with a header. In this paper, we constrain ourselves to grayscale images.

The original uncompressed 8-bit grayscale image with $N_1 \times N_2$ pixels is an element of $\{0, 1, \dots, 255\}^{N_1 \times N_2}$. Throughout this paper, $\mathbf{x}^{(0)} = (x_{ij}^{(0)})$ 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 $y_{kl}^{(0)} = \mathrm{DCT}_{kl}(\mathbf{x}^{(0)}) \triangleq \sum_{i,j=0}^{7} f_{kl}^{ij} x_{ij}^{(0)}$, $0 \le k, l \le 7$, where

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

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 $\mathbf{c}^{(0)} = [\mathbf{y}^{(0)} \oslash \mathbf{q}]$, $c_{kl}^{(0)} \in \{-1024, \dots, 1023\}$ 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 $\widetilde{x}_{ij}^{(0)} = \mathrm{DCT}_{ij}^{-1}(\widetilde{\mathbf{y}}^{(0)}) \triangleq \sum_{k,l=0}^{7} f_{kl}^{ij} \widetilde{y}_{kl}^{(0)}$, where $\widetilde{x}_{ij}^{(0)} \in \mathbb{R}$, or in the matrix form $\widetilde{\mathbf{x}}^{(0)} = \mathbf{D}^{\top} \widetilde{\mathbf{y}}^{(0)}$. The pair (i, j) used to index $\widetilde{x}_{ij}^{(0)}$ is called the ij^{th} JPEG phase [15]. 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 by the Warden to obtain a reference image. To simplify the analysis and improve the clarity of our results, we postpone discussion of potentially mismatching the steganographer and Warden's JPEG compressors to Section 8. Until then, we assume the two parties share and have full knowledge of the same JPEG compressor (besides the initial compression possibly involving trunc). Since $\mathbf{q}$ is not available in a decompressed JPEG's file format, recompression in practice is performed using a quantization matrix, $\widehat{\mathbf{q}}$, estimated directly from $\mathbf{x}$ or $\mathbf{x}^{(s)}$. To simplify matters further, until Section 9 we assume the Warden knows the exact quantization table $\widehat{\mathbf{q}} = \mathbf{q}$.

$$\begin{tikzcd}
 & \boldsymbol{\delta} = \widetilde{\mathbf{x}}^{(0)} - \mathbf{x} & & \\
\mathbf{x}^{(0)} \arrow[d, "\mathbf{D}"] & \widetilde{\mathbf{x}}^{(0)} \arrow[r, "{[\cdot]}"] & \mathbf{x} \arrow[d, "\mathbf{D}"] & \widetilde{\mathbf{x}} \\
\mathbf{y}^{(0)} \arrow[r, "{[\cdot]_{\mathbf{q}}}"] & \widetilde{\mathbf{y}}^{(0)} \arrow[u, "\mathbf{D}^{-1}"] & \mathbf{y} \arrow[r, "{[\cdot]_{\widehat{\mathbf{q}}}}"] & \widetilde{\mathbf{y}} \arrow[u, "\mathbf{D}^{-1}"] \\
 & \boldsymbol{\varepsilon}^{(0)} = \mathbf{y}^{(0)} - \widetilde{\mathbf{y}}^{(0)} & & \boldsymbol{\varepsilon} = \mathbf{y} - \widetilde{\mathbf{y}}
\end{tikzcd}$$

Figure 1: JPEG compression - decompression - recompression pipeline. Adjusting pixels to $[-128, 127]$ **and clipping to** $[0, 255]$ **are ignored.**

Figure 1 visually conveys the JCA pipeline considered in this paper. 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 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.

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 [23]. Ignoring clipping, the (negative) SQ error can be seen as the unrounded recompression residual since $\mathbf{x}$ is a block of integers:

$$[-\mathbf{D}^{-1}\boldsymbol{\varepsilon}] = [\widetilde{\mathbf{x}} - \mathbf{x}] = [\widetilde{\mathbf{x}}] - \mathbf{x}. \tag{3}$$

4 PIPELINE ANALYSIS

Equipped with the tools introduced in Section 2.2, we can now study the objects in Figure 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) Cover / stego images are stored using integers which allows us to analytically isolate the rounding errors in each domain.
(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 $\boldsymbol{\delta}$ is approximately uniform with jointly independent components for lower JPEG quality factors.

4.1 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

$$\begin{aligned}\widetilde{\mathbf{x}}^{(0)} &= \mathbf{D}^{-1}\widetilde{\mathbf{y}}^{(0)} \\ &= \mathbf{D}^{-1}\mathbf{y}^{(0)} - \mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)} \\ &= \mathbf{x}^{(0)} - \mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)}. \qquad (4)\end{aligned}$$

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

Assumption 1. *For all modes (k,l), the DCT quantization errors $\varepsilon_{kl}^{(0)}$ are jointly independent and satisfy*

$$\varepsilon_{kl}^{(0)} \sim \mathcal{U}[-q_{kl}/2, q_{kl}/2). \qquad (5)$$

Assumption 1 has been studied in [32], used in [8, 9, 29], 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

$$\widetilde{x}_{ij}^{(0)} \sim \mathcal{N}(x_{ij}^{(0)}, s_{ij}^{(0)}), \qquad (6)$$

with variance

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

The rounding error in the spatial domain has the form

$$\boldsymbol{\delta} = \widetilde{\mathbf{x}}^{(0)} - [\widetilde{\mathbf{x}}^{(0)}] = \mathrm{err}_1(-\mathbf{D}^{-1}\boldsymbol{\varepsilon}^{(0)}), \qquad (8)$$

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 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. (7), 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.

4.2 Cover images

By reasoning similar to that of Eq. (4), the linearity of the DCT implies

$$\mathbf{y} = \widetilde{\mathbf{y}}^{(0)} - \mathbf{D}\boldsymbol{\delta}. \qquad (9)$$

Assumption 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. As discussed at the end of Section 4.1, the PLT only guarantees joint independence for QFs 98 and below for round and trunc. We assume joint independence for QF99 and 100 as well in Assumption 2 for modeling simplicity. Using Assumption 2, we may ignore the effects of clipping and approximate the marginals of $\mathbf{y}$ using the CLT:

$$y_{kl} \sim \mathcal{N}(\widetilde{y}_{kl}^{(0)}, s_{kl}), \qquad (10)$$

$$s_{kl} = \sum_{i,j=0}^{7}(f_{kl}^{ij})^2\mathrm{Var}[\delta_{ij}]. \qquad (11)$$

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

$$\boldsymbol{\varepsilon} = \mathbf{y} - [\mathbf{y}]_{\mathbf{q}} = \mathrm{err}_{\mathbf{q}}(-\mathbf{D}\boldsymbol{\delta}), \qquad (12)$$

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

4.3 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

$$\mathbf{y}^{(s)} = \widetilde{\mathbf{y}}^{(0)} - \mathbf{D}\boldsymbol{\delta} + \mathbf{D}\boldsymbol{\eta}. \qquad (13)$$

Assumption 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

$$y_{kl}^{(s)} \sim \mathcal{N}(\widetilde{y}_{kl}^{(0)}, s_{kl} + r_{kl}), \qquad (14)$$

$$r_{kl} = \sum_{i,j=0}^{7}(f_{kl}^{ij})^2\mathrm{Var}[\eta_{ij}]. \qquad (15)$$

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

$$\boldsymbol{\varepsilon}^{(s)} = \mathrm{err}_{\mathbf{q}}(-\mathbf{D}\boldsymbol{\delta} + \mathbf{D}\boldsymbol{\eta}), \qquad (16)$$

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.

5 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 6.

All experiments in this section, and in this paper in general, were conducted on the union of the BOSSbase 1.01 [1] and BOWS2 [2] 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 [6, 34–36, 38]. 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.

5.1 Likelihood ratio test

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

$$\mathcal{H}_0 : \varepsilon_{kl} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl}, q_{kl}) \tag{17}$$

$$\mathcal{H}_1 : \varepsilon_{kl} \sim \mathcal{N}_{\mathcal{W}}(0, s_{kl} + r_{kl}, q_{kl}),\ r_{kl} > 0. \tag{18}$$

Assumption 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 [9, 12]. Thus, the log-likelihood ratio test for an image is

$$\mathcal{L}(\mathcal{B}) = \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \mathcal{L}_{kl}(\mathbf{x}) \tag{19}$$

$$= \sum_{\mathbf{x}\in\mathcal{B}} \sum_{k,l=0}^{7} \log \frac{g(\varepsilon_{kl}; 0, s_{kl} + r_{kl}, \widehat{q}_{kl})}{g(\varepsilon_{kl}; 0, s_{kl}, \widehat{q}_{kl})} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma. \tag{20}$$

Assuming the change rates (and thus r_{kl}) are known, the Warden is faced with a simple hypothesis, for which the LRT is uniformly most powerful in the clairvoyant case according to the NP-lemma [21]. As a remark, we remind the reader that the quantization matrix must be estimated from the image first — a preanalytical step discussed in Section 9. 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 [13, 16, 18] or methods with synchronized embedding changes [7, 17, 26]. 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 the Lindeberg's extension of the CLT, the normalized LRT

$$\Lambda(\mathcal{B}) = \frac{\mathcal{L}(\mathcal{B}) - \mathbb{E}_{\mathcal{H}_0}[\mathcal{L}(\mathcal{B})]}{\sqrt{\mathrm{Var}_{\mathcal{H}_0}[\mathcal{L}(\mathcal{B})]}} \tag{21}$$

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.

5.2 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 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.[3] 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 paper, 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 Warden'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 6.

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.

6 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 Warden 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, the authors 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, DCT quantizers,

[3]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.

and possibly trained for unknown payloads simply by enlarging the training set.

These advantages motivated the authors to study 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 6.1 and Section 6.2 introduce the experimental setup for SRNet and the prior art, respectively.

6.1 SRNet

In this paper, 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, the one-cycle learning-rate (LR) scheduler with maximum LR 1×10^{-3} [31], 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.

6.2 RRH

For comparison against the prior art, we also implemented the RRH method [23] (see Section 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.

7 EXPERIMENTS

The goal of this section is to determine the best detector from Section 5 and 6. First, we compare the performance of the LRT and the three SRNets w.r.t. JPEG quality for a fixed stego scheme and payload. The best detector of these four is then 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 4. We assume the JPEG compressor is fixed and the Warden knows the true quantization table to mitigate the effects of confounding variables while studying detectability trends.

7.1 Methodology

As in Section 5, 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 fairly compare the LRT to the machine learning detectors (as outlined in [11]), we use the non-normalized version $\mathcal{L}(\mathcal{B})$ (19) and 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. Owing to Doddington's rule of 30 and for visual clarity, in all experiments in this paper we report test accuracies above 0.9970 as "≈ 1".

7.2 Performance w.r.t quality

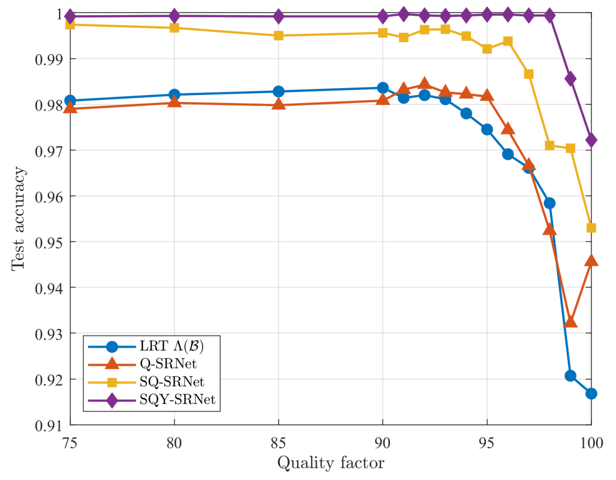
In Figure 2, the left plot visualizes the trends for the LRT and all versions of SRNet. Since SQY-SRNet outperformed the other detectors especially for high qualities, we continued by testing SQY-SRNet and the prior art on the following four content-adaptive steganographic schemes: S-UNIWARD [16], HILL [25], MiPOD [30], and WOW [14]. These schemes were tested on the following range of payloads so an informative comparison between SQY-SRNet and RRH can be made: 0.02, 0.01, 0.005, and 0.002 bpp. We refer the reader to Tables 5 and 6 in the appendix for the full results for SQY-SRNet and the prior art. A subset of these results are shown in the right plot of Figure 2. The SQY-SRNet significantly outperforms the RRH especially for small payloads for QFs above 93.

Note that the model-based MiPOD is consistently more secure that 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 to the average number of modified pixels 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 modelling 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 4.1. 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 4.2.

We note that SRNet generally has trouble forming inter-block statistics in DCT domain representations [37] which is likely why we see a jump in performance when the SQ error is used instead.

In [12], 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.



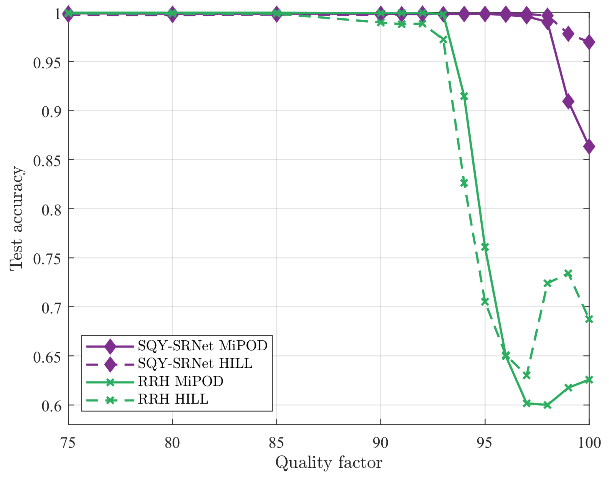


Figure 2: Left: Testing accuracy as a function of JPEG quality for LRT (21) and all flavors of SRNet. Embedded using MiPOD at 0.01 bpp. Right: Testing accuracy as a function of JPEG quality for SQY-SRNet (purple) and RRH (green). Embedded using MiPOD (solid) and HILL (dashed) at 0.005 bpp.

Note that SQY-SRNet closely matches the monotonic behavior we intuitively expect.

8 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 uncompressed (or 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 [3]. In this section, we study three types of mismatches involved with the steganographer's and Warden's choices of compressors / decompressors. We will show experimentally that SQY-SRNet can be trained to be substantially more robust to implementation differences compared to the prior art RRH. Additionally, we will employ the analysis from Section 4 to explain quality factor trends and performance differences between quantizers.

8.1 Types of mismatch

To make our discussion of compressor mismatch and robustness precise, we introduce the following notation based on the JPEG pipeline discussed in Section 3 and shown in Figure 1. The reader is advised to follow Figure 3 for easier understanding. In general, the initial JPEG compression comp_S (S for "steganographer") satisfies $\widetilde{\mathbf{y}}^{(0)} = \text{comp}_\text{S}(\mathbf{x}^{(0)})$, mapping the uncompressed image to the dequantized DCT coefficients.[4] In order to embed a secret message, the steganographer (or the embedding tool) decompresses the image using dec_S to obtain the cover image $\mathbf{x} = \text{dec}_\text{S}(\widetilde{\mathbf{y}}^{(0)})$. The Warden typically does not have access to comp_S and dec_S and must use their own comp_W and dec_W (W for "warden") to generate JPEG covers for their training dataset. The subsequent recompression and decompression is denoted $\widetilde{\mathbf{y}} = \text{comp2}_\text{W}(\mathbf{x})$ and $[\widetilde{\mathbf{x}}] = \text{dec2}_\text{W}(\widetilde{\mathbf{y}})$. We remind the reader that the steganographer typically does not have access to comp_S either. Figure 3 visualizes the relationship between these compressors and decompressors based on the pipeline in Figure 1. Additionally, Figure 3 delineates how the Warden generates decompressed JPEGs for training their detector (top row) and how the steganographer generates the decompressed JPEG for embedding (bottom row).

Since the recompression method for the JCA is the Warden's choice, they are generally free to select the one that works the best overall. To simplify matters, we exlusively use Matlab's `imwrite` for comp2_W and `imread` for dec2_W to compute the recompression residual for the prior art RRH [23] since this variant was used for benchmarking in Section 6. To compute Q / SQ errors, we manually recompress via SciPy's `dct` for all experiments since rounding errors are not easily attainable using off-the-shelf JPEG compressors.

Fixing comp2_W and dec2_W, we focus on three types of mismatches depicted in Figure 3. First, the "decompressor mismatch" refers to mismatching implementations of dec_S and dec_W. Second, the "quantizer mismatch" considers differences between the quantizers of comp_S and comp_W each possibly being round or trunc. For experiment feasibility, we do not consider differences between DCT implementations of comp_S and comp_W. Both the decompressor mismatch and quantizer mismatch contribute to the overall cover source mismatch between the Warden's training set and the covers the steganographer uses in practice. The third type of mismatch, called "steganographer's mismatch", occurs when the steganographer uses an implementation of dec_S different than the one used for comp_S.[5] This mismatch may complicate the distribution of rounding errors and potentially decrease the performance of the JCA. We

[4] In practice, compressors compute quantized DCTs $\mathbf{c}^{(0)}$, but note that it is equivalent to use dequantized DCTs $\widetilde{\mathbf{y}}^{(0)}$.

[5] In general, since the steganographer and Warden do not have access to the full compression history, there could be mismatches between any pair of comp_S, dec_S,

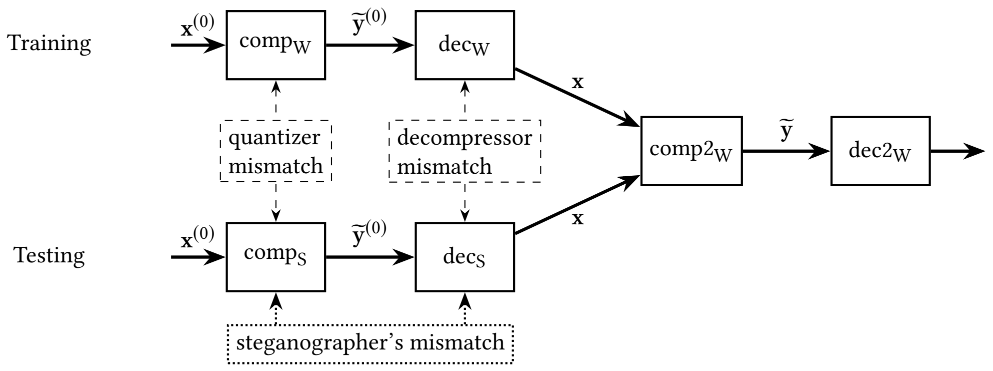

Training
$\mathbf{x}^{(0)}$
comp_W
$\widetilde{\mathbf{y}}^{(0)}$
dec_W
$\mathbf{x}$
quantizer mismatch
decompressor mismatch
$\mathrm{comp2}_\mathrm{W}$
$\widetilde{\mathbf{y}}$
$\mathrm{dec2}_\mathrm{W}$
Testing
$\mathbf{x}^{(0)}$
comp_S
$\widetilde{\mathbf{y}}^{(0)}$
dec_S
$\mathbf{x}$
steganographer's mismatch

Figure 3: Compression pipeline for images in training set (top row) and pipeline for images created by the steganographer (bottom row). The quantizer and decompressor mismatch are drawn using the same line style (dashed) to signify they are types of cover source mismatch. See Sec. 8.1 for more detail.

remind the reader that there is no need to study differences in quantizer for $\mathrm{comp2}_\mathrm{W}$ since Q error is always computed using round (it does not make sense to use trunc per our analysis). Additionally, the "quantizer" for all decompressors dec_S, dec_W, $\mathrm{dec2}_\mathrm{W}$ always involves rounding to nearest integer and clipping to dynamic range as per JPEG standard.

The following implementations are considered in our experiments: Matlab's `imwrite`/`imread`, Python3 library PIL (PIL), ImageMagick's Convert (Convert), Int and Float DCT compressors in libjpeg (version 6b).[6] 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.[7] When studying differences between quantizers, we compare Matlab's `imwrite` to a manually implemented trunc compressor written in Python3 using SciPy's `dct`.

8.2 Mismatching the decompressor

First, we solely focus on the combined effect of the decompressor mismatch and steganographer's mismatch. In particular, we determine the best dec_W for training assuming 1) comp_S and comp_W are fixed to Matlab's `imwrite` and 2) the steganographer was free to choose any of the decompressors. Table 1 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' decompresser, the detector trained on images decompressed with Python's PIL and Matlab's `imread` generalized overall very well when evaluated on images from all five decompressors dec_S.

We also studied RRH's robustness to decompressor since no benchmarking exists in [23]. The testing accuracies for the RRH are shown in Table 2. We observed that QF99 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.

and $\mathrm{comp2}_\mathrm{W}$. Note the Warden can guarantee no mismatch occurs between $\mathrm{comp2}_\mathrm{W}$ and $\mathrm{dec2}_\mathrm{W}$.

[6] http://libjpeg.sourceforge.net/

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

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 [23]. Thus, for QF92 and below, embedding is highly detectable since single pixel changes will appear in the recompression residual. We observed, however, that the steganographer's mismatch commonly creates salt-and-pepper noise artifacts in the recompression residual, which the RRH misinterprets as steganography. For example, QF90 RRH has an accuracy of .6349 (see Table 2) when the float decompressor is used for both dec_W and dec_S and the compressor comp_S is Matlab's `imwrite`; just having steganographer's mismatch can greatly impact RRH's performance. When dec_W is `imread` instead, QF90 RRH is a random guesser now due to the combined effect of decompressor mismatch and steganographer's mismatch. For QFs above 92, steganographer's mismatch is less problematic since the RRH classifier gets trained on covers that more commonly produce salt-and-pepper noise.

8.3 Mismatching the quantizer

Having seen that training on MATLAB's `imread` or PIL decompressors generalize the best for decompressor mismatch, we turn to investigating the effect of using a trunc quantizer for both for comp_S and comp_W while considering effects of steganographer's mismatch. Table 3 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 (c.f. Table 1) with the largest difference for QF100. This is related to the differences between both quantizers, namely the way they affect the distribution of the rounding errors δ_{ij} in the spatial domain (8). Except for QFs 99–100, $\boldsymbol{\delta}$ is well-approximated by the uniform distribution for both quantizers (see Section 4.1 and the PLT). Therefore, the SQ errors for both quantizers approximately follow the same distribution under assumptions of Section 4 which explains the matching accuracies for QF95. For QFs 99–100, however, the DCT quantization errors for the trunc quantizer $\varepsilon^{(0)}_{kl} \in [0, q_{kl})$ for positive DCTs and $\varepsilon^{(0)}_{kl} \in (-q_{kl}, 0]$ for negative DCTs. Thus, any

Table 1: Testing accuracy for SQY-SRNet trained on dec_W and tested on dec_S. Each row / column corresponds to the decompressor used for training / testing, respectively. Matlab's `imwrite` is used for comp_S and comp_W. Q / SQ errors are computed with SciPy's `dct`. Embedded using MiPOD at 0.01 bpp.

QF	dec_W	dec_S				
		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	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	float	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	int	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	convert	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	PIL	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1

Table 2: Testing accuracy for RRH trained on dec_W and tested on dec_S. Matlab's `imwrite` is used for comp_S and comp_W and used to compute the recompression residual. Embedded using MiPOD at 0.01 bpp.

QF	dec_W	dec_S				
		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	≈ 1	.5000	≈ 1	.5000	≈ 1
	float	.4819	.6349	.4816	.6359	.4817
	int	≈ 1	.5000	≈ 1	.5000	≈ 1
	convert	.4831	.6350	.4828	.6350	.4823
	PIL	≈ 1	.5000	≈ 1	.5000	≈ 1

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 quantizer is much less affected by such asymmetries.[8] 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 paper) and, apparently, also on the CNN detectors.

[8] 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 3: Testing accuracy for SQY-SRNet when both comp_S and comp_W use the trunc quantizer. Embedded using MiPOD at 0.01 bpp.

QF	dec_W	dec_S				
		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	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	PIL	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1

Table 4: Testing accuracy for SQY-SRNet trained on the union of images initially compressed, comp_W, with round (Matlab's `imwrite`) and trunc at QF 100. SQY-SRNet is tested on round and trunc images (comp_S) separately. Embedded using MiPOD at 0.01 bpp.

comp_S	dec_W	dec_S				
		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

Next, we investigated what happens when one of comp_S and comp_W is round and the other is trunc. We found that SQY-SRNet exhibits no loss of accuracy for mismatched quantizers at QF95 (≈ 1 test accuracy for all dec_S) but a significant loss for QF99 and QF100 (near random guessing for all dec_S). 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 4 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 1 and 3). Overall, training on the `imread` decompressor generalizes slightly better than training on PIL.

9 ESTIMATING THE QUANTIZATION TABLE

Since the quantization table $\mathbf{q}$ is not provided in a decompressed JPEG image, whether it is cover or stego, we must first first determine that the image is a decompressed JPEG and also estimate the quantization table so that the image can be analyzed with the correct JCA detector. These two preliminary steps are well studied by the forensics community [12, 33]. In particular, we refer the reader to [33] for its intuitive, model-based approach and for its detailed discussion on two unavoidable issues most quantization table estimators face: 1) incorrectly estimating a divisor of the quantization step and 2) the so-called "indeterminable steps." In Section 9.1 we will, however, leverage our analysis from Section 4 to argue that these issues, in fact, vanish when estimating $\mathbf{q}$ for the JCA. Additionally, most model-based quantization table estimators are unable to distinguish between QF100 and uncompressed images, and are typically not tested on stego images. In Section 9.2, we will discuss the practical issues of distinguishing between QF100 and uncompressed images, and we will also show that embedding changes have a negligible effect on the estimation.

9.1 Estimating Q errors is easier

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, what we call "sufficient" steps. As it happens, quantization step estimation methods such as the one proposed in [33] will often select a divisor of the true step when wrong, which tells us that steps are commonly sufficient in practice.

Formally, 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}.

Theorem 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, Theorem 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. A proof of Theorem 1 can be found in the Appendix. Additionally, the proof will show that Theorem 1 does not depend on the type of quantizer used for the initial JPEG compression and will show that embedding changes have a negligible effect on the estimation.

Next, we can overcome an issue that commonly occurs when the true quantization step is large (which is usually the case for high frequency modes / low quality factors). For large enough quantization steps, the quantized DCTs $\widetilde{y}_{kl}^{(0)}$ of the uncompressed image will all be 0. This will cause the non-quantized DCTs y_{kl} of the recompressed image to have a single cluster around 0. Specifically, for a fixed mode k, l, if $q_{kl} \geq 2$ and if the non-quantized DCTs y_{kl} of the recompressed image are contained within the interval $[-1, 1)$ across all blocks, we say that the quantization step at mode k, l is *indeterminable*. This indeterminable case is a point of failure for many quantization step estimation methods including [33] since they rely on finding the most likely integer spacing of clusters of y_{kl} of the recompressed image, which cannot be done if there is only one cluster at 0. However, indeterminable steps do not pose a problem for the JCA; for any chosen step $\widehat{q}_{kl} \geq 2$ we can correctly compute the Q error since $\mathrm{err}_{\widehat{q}_{kl}}(y_{kl}) = y_{kl} = \mathrm{err}_{q_{kl}}(y_{kl})$ for $y_{kl} \in [-1, 1)$.

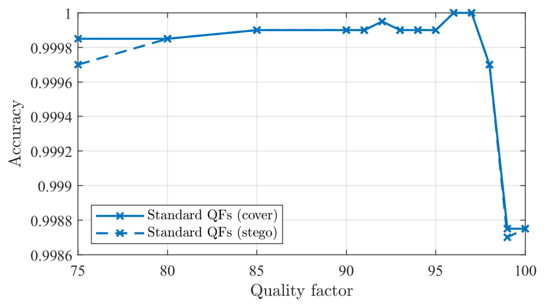


Figure 4: The ratio of images in BOSSBOWS2 whose quality factors were correctly estimated from covers (solid) and stegos (dashed) embedded using MiPOD at 0.01 bpp.

9.2 Practical Considerations

In this paper, we make use of a simplified version of the estimator proposed in [33]. In short, our version is a maximum likelihood estimator restricted to estimating standard quantization tables[9] using a uniform prior for the quantized DCTs $\widetilde{y}_{kl}^{(0)}$. We refer the reader to [10] for implementation details of our simplified estimator. If the estimated quality is not QF100, then we declare the image to be JPEG compatible and feed it to the SQY-SRNet trained on the estimated QF. However, many model-based estimators, including ours, output tables with all ones if given either a QF100 decompressed JPEGs or uncompressed image. Thus, if the estimated quality is QF100, we run into an issue not addressed by [12, 33]. To solve this issue for our estimator, we trained a version of SQY-SRNet to disinguish between QF100 decompressed JPEG (cover) images and uncompressed images from BOSSBOWS2 using the same hyperparameters from previous sections; this detector achieved a test accuracy of ≈ 1.

Finally, we experimentally verify that embedding changes have a negligible effect on the estimation accuracy. Figure 4 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.

10 PERFORMANCE IN REAL-LIFE CONDITIONS

To demonstrate the usefulness and relevance of this work, we evaluated SQY-SRNet's performance in real-life conditions (using real stego tools) for one "easy" quality factor, QF95, and for QF99 and 100. We looked up stego tools from two sources: a github repository search and a Google search. Github contains open-source tools with available source code predominantly written in python. These tools have typically been created and are maintained by amateurs or communities with interest in the field. The Google search returned proprietary commercial tools which are expected to be more secure. As mentioned in Section 1, the user could also introduce a vulnerability by decompressing the image themselves

[9] 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.

before feeding the image to the tool. We do not consider this case since it is infeasible to assess how likely this is going to happen. A tool is considered only if it does the decompression. The list of all stego tools together with additional details can be found at http://dde.binghamton.edu/download/jca/.

To make our detectors applicable to a wider range of stego tools, we opted to train SQY-SRNet, for each QF, on a rather diverse stego source with a fixed payload 0.02 bpp using embedding simulators. To create covers, we compress and decompress using Matlab's `imwrite`. To create a stego image, we first flip an unbiased three-sided coin to determine whether we use embedding simulators for optimally coded MiPOD, HILL, or non-adaptive LSB matching. If the coin tells us to use LSBM, we flip an unbiased two-sided coin to determine whether we embed in pseudo-randomly selected pixels (with optimal ternary codes) or by rows / columns without coding. By inspection, some tools used sequential embedding on the top of the image or the bottom of the image, so we applied a random D4 operation to the selection channel to account for these variations. All hyper-parameters for training are the same as the ones in previous sections, except we do not use pair contraint, and the net was seeded with weights from training on MiPOD.

10.1 Results

Out of 42 tools, we discovered that 19 of them output a decompressed JPEG stego image when given a regular JPEG image as input. In general, in order for a tool to be susceptible to JCA, the tool must output the stego image in a raster format. That is, if the output is a JPEG we automatically assume the tool is not susceptible to JCA. We point out that all cover images supplied to the tools were grayscale (they are from BOSSBOWS2). Out of the 19 tools, there were 15 that considered grayscale cover images as color and distributed the payload across three copies of the Y channel, outputting thus a color stego image. For these tools, we adjusted the size of the embedded payload to be three times larger so that the relative payload is still 0.02 bpp in each channel, and we steganalyzed them by simply supplying the detector with the R channel. Of course, this embedding will create suspicious ± 1 differences between color channels, so the Warden, in practice, could simply check for these artifacts [19] before using the JCA.

For each of the 19 tools, we did the following for each quality QF $\in \{95, 99, 100\}$. Since the tools have to be used manually through either a terminal or GUI, which is labor intensive, we selected at random five uncompressed images from TST and JPEG compressed them using Matlab's `imwrite` with quality QF.[10] Then, we embedded a random message of relative length 0.02 bpp in each cover presented to the tool as JPEG. Using our quantization table estimator from Section 9, we determined each stego image's quality (from the R channel if the tool outputted a color image). We computed the SQ error using the estimated table $\widehat{\mathbf{q}}$. For images estimated to be QF100, we additionally sent these images to our QF100 vs. uncompressed detector (Section 9) to verify they are QF100. Finally, the images and their SQ error were sent to the SQY-SRNet trained on the *estimated* quality. Each SQY-SRNet's decision threshold was set to achieve a 10^{-3} false alarm rate on TST.

[10]Different images were selected for different qualities and tools.

For 15 of the 19 susceptible tools (different 15 than above), the quantization table estimator and SQY-SRNets correctly identified the quality factor and perfectly classified the images as stego (meaning all $15 \times 5 \times 3$ images were detected). For the tools with source code available (a total of 12 tools), we double-checked the code to see that JPEGs are decompressed, embedded, and saved to disk in raster format without any processing in the spatial domain that breaks JPEG compatibility.

The remaining four tools produced stego images whose quality factors were difficult to determine—most of the images were classified as uncompressed. One of these tools can be easily detected, since the embedding is done simply to the three least significant bits, which introduces visible artifacts. The other tools either have no source code available, or source code that is not well documented and difficult to analyze.

11 CONCLUSIONS

This paper 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 w.r.t. 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 a 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 Warden to build the training datasets, this paper includes a comprehensive study of the robustness of the SQY-SRNet w.r.t. 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 similiar level of accuracy as the clairvoyant detectors informed by and trained on the right combination of cover JPEG quantizer and decompressor.

The paper is closed with a test of the JCA under realistic conditions. A total of 15 steganographic tools available from public repositories (out of 42 inspected) were found to be vulnerable to the attack and were reliably detected with our detectors.

Our future effort will be directed towards extending the JCA to color images, to make it robust to errors when estimating custom quantization tables, and to study the JCA on a very diverse image source such as Flickr where the development pipeline is unknown.

12 ACKNOWLEDGEMENTS

The work on this paper was supported by NSF grant No. 2028119. Special thanks go to Jan Butora for helping start this work.

APPENDIX

In this section, we prove Theorem 1. The theorem is trivial to prove under the condition $\widehat{q}_{kl} = q_{kl}$, so we assume $q_{kl} > \widehat{q}_{kl} \geq 2$ and

$\widehat{q}_{kl}$ divides q_{kl}. From Eq. (10), the PDF of y_{kl} can be expressed as

$$f_{y_{kl}}(u) = \sum_{n\in\mathbb{Z}} \frac{\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})}{\sqrt{2\pi s_{kl}}} \exp\left(-\frac{(u - nq_{kl})^2}{2s_{kl}}\right), \tag{22}$$

where $\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})$ is the prior probability[11] that $y_{kl}^{(0)}$ had quantized to nq_{kl}. The density $f_{\widehat{\varepsilon}_{kl}}$ is obtained by wrapping $f_{y_{kl}}$ (22) 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 $f_{\varepsilon_{kl}}(u) \approx 0 = f_{\widehat{\varepsilon}_{kl}}(u)$.[12] In particular, $|f_{\widehat{\varepsilon}_{kl}}(u) - f_{\varepsilon_{kl}}(u)| = f_{\varepsilon_{kl}}(u) \leq C$ by direct evaluation of the maximum.[13]

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

$$m\widehat{q}_{kl} - nq_{k\ell} = m\widehat{q}_{kl} - nj\widehat{q}_{kl} = (m - nj)\widehat{q}_{kl}, \tag{23}$$

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. (23) to produce

$$\begin{aligned} f_{\widehat{\varepsilon}_{kl}}(u) &= \sum_{n\in\mathbb{Z}} \frac{\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})}{\sqrt{2\pi s_{kl}}} \sum_{m\in\mathbb{Z}} \exp\left(-\frac{(u + m\widehat{q}_{kl})^2}{2s_{kl}}\right) \\ &= \frac{1}{\sqrt{2\pi s_{kl}}} \sum_{m\in\mathbb{Z}} \exp\left(-\frac{(u + m\widehat{q}_{kl})^2}{2s_{kl}}\right), \end{aligned} \tag{24}$$

for $u \in [-\widehat{q}_{kl}/2, \widehat{q}_{kl}/2)$. The last line in Eq. (24) 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 $g(u; 0, s_{kl}, \widehat{q}_{kl})$ (1) 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 Theorem 1 as desired.[14] Observe that Theorem 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 theorem 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 $r_{kl} \ll 1$.

REFERENCES

[1] 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.

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

[3] M. Beneš, N. Hofer, and R. Böhme. Know your library: How the libjpeg version influences compression and decompression results. In J. Butora, B. Tondi, and C. Veilhauer, editors, *The 10th ACM Workshop on Information Hiding and Multimedia Security*, Santa Barbara, CA, 2022. ACM Press.

[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.

[11] In practice, for each mode (k, l) we must estimate $\mathbb{P}(\widetilde{y}_{kl}^{(0)} = nq_{kl})$ from the decompressed JPEG itself.

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

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

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

[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] 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.

[10] E. Dworetzky. Explaining and improving the JPEG compatibility attack with statistical hypothesis testing and deep learning. Master's thesis, Binghamton University, Binghamton, NY, 2021.

[11] E. Dworetzky, E. Kaziakhmedov, and J. Fridrich. On comparing ad hoc detectors with statistical hypothesis tests. In Y. Yousfi, C. Pasquini, and A. Bharati, editors, *The 11th ACM Workshop on Information Hiding and Multimedia Security*, Chicago, IL, June 28–30, 2023. ACM Press.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] N. F. Johnson and S. Jajodia. Steganalysis of images created using current steganography software. In D. Aucsmith, editor, *Information Hiding, 2nd International Workshop*, volume 1525 of Lecture Notes in Computer Science, pages 273–289, Portland, OR, April 14–17, 1998. Springer-Verlag, New York.

[20] 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.

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

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

[29] 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.

[30] V. Sedighi, R. Cogranne, and J. Fridrich. Content-adaptive steganography by minimizing statistical detectability. *IEEE Transactions on Information Forensics*

Table 5: 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	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9924	.9899
	0.01	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9856	.9721
	0.005	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9960	.9903	.9094	.8634
	0.002	.9918	.9905	.9916	.9874	.9856	.9791	.9747	.9587	.9231	.7512	.6891
HILL	0.02	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	0.01	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9964	.9950
	0.005	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9968	.9783	.9698
	0.002	.9926	.9948	.9926	.9927	.9906	.9885	.9885	.9841	.9683	.8958	.8717
S-UNI	0.02	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9970	.9959
	0.01	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9934	.9918
	0.005	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9968	.9650	.9498
	0.002	.9934	.9923	.9910	.9921	.9908	.9895	.9831	.9721	.9603	.8511	.8109
WOW	0.02	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1
	0.01	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9965	.9959
	0.005	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	≈ 1	.9804	.9725
	0.002	.9920	.9942	.9917	.9929	.9912	.9888	.9872	.9813	.9726	.8978	.8630

Table 6: 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	≈ 1	≈ 1	≈ 1	.9970	.9924	.9778	.9189	.8722	.9052	.9239	.9290
	0.01	≈ 1	≈ 1	≈ 1	≈ 1	.9826	.9035	.7778	.7114	.7172	.7543	.7577
	0.005	≈ 1	≈ 1	≈ 1	≈ 1	.9147	.7610	.6497	.6016	.5999	.6175	.6257
	0.002	.9939	.9942	≈ 1	.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	≈ 1	≈ 1	≈ 1	.9924	.9819	.9562	.9078	.8816	.9368	.9536	.9501
	0.01	≈ 1	≈ 1	.9961	.9939	.9598	.8744	.7776	.7303	.7930	.8142	.7964
	0.005	≈ 1	≈ 1	.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

and Security, 11(2):221–234, 2016.

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

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.