INCREASING TRUST IN IMAGE ANALYSIS BY DETECTING TRELLIS QUANTIZATION IN JPEG IMAGES

Nora Hofer

Security and Privacy Lab, University of Innsbruck, Austria

ABSTRACT

JPEG image forensics investigates the authenticity and origin of compressed images. Many established methods rely on assumptions about the statistical distribution of quantized discrete cosine transform coefficients. However, JPEG implementations that use trellis quantization, such as *mozjpeg*, produce images that challenge these assumptions. In this study, we demonstrate that artifacts resulting from trellis quantization can compromise the reliability of established forensic methods and cause false alarms for innocuous images. We address this issue by presenting methods to detect trellis artifacts and validating their robustness in scenarios commonly encountered in forensic analyses.

***Index Terms*—** trustworthy image forensics, steganalysis, trellis quantization, *mozjpeg*

1. INTRODUCTION

Trellis quantization [1] addresses the rate-distortion problem in data compression by finding the path through a trellis structure that minimizes a cost function. This cost function balances the size in bits needed to encode a coefficient value against the distortion introduced by quantization. By evaluating the cumulative cost of different paths, trellis quantization identifies the sequence of quantization steps that results in the most efficient compression with minimal loss in quality. Trellis quantization is particularly effective in the compression of transform coefficients, such as those obtained from the Discrete Cosine Transform (DCT) in video [2] and image compression [3].

Mozjpeg [4] is a popular JPEG compression library that implements a variant of trellis quantization by default to achieve reduced file sizes. Specifically, it employs a perceptual model that accounts for the additional distortion introduced by changing coefficients to values with shorter variable-length encodings.

These modifications have been found to cause characteristic artifacts in the DCT coefficient distribution of compressed images [5]. While such artifacts can be exploited in image forensics to fingerprint the JPEG implementation [6], they might pose challenges to other forensic applications if they rely on assumptions about the distribution of quantized DCT coefficients.

In recent years, statistical learning has gained popularity in multimedia forensics [7]. However, while machine learning detectors achieve high performances, they are known to be sensitive to training-test mismatches.

Table 1 demonstrates the detrimental effects of unaddressed trellis artifacts on the detection of image steganography. Similar to related work [8], we train three EfficientNet-B0 detectors [9] on cover and stego images compressed with *libjpeg–turbo*, a widely used JPEG compression library that does not implement trellis quantization. The left and center column in Table 1 show the baseline accuracy and the false positive rate (FPR). Next, we evaluate the detectors' sensitivity to images compressed with *mozjpeg*. Up to 99% of all innocuous cover images are now falsely classified as stego images, as shown in the rightmost column. Both test sets use the same images, DCT method, subsampling, and quantization table (QT). The differences in the FPR can, therefore, be attributed to the characteristics of *mozjpeg*'s trellis quantization.

Table 1: State-of-the-art steganalysis models misclassify innocent cover images if they are unaware of trellis artifacts.

Detection performance			
Embedding	*libjpeg–turbo*		*mozjpeg*
	Baseline acc.	FPR	FPR
nsF5 [10]	99%	1%	**99%**
UERD [11]	93%	4%	**43%**
J-UNIWARD [12]	91%	8%	**94%**

ImageNet-pretrained, 32 batch size, 0.25 dropout rate, 0.0001 learning rate, Adam, QF 75, 0.4 bits per non-zero AC coefficients (bpnzAC), ALASKA2.

This work is funded by the European Union's Horizon 2020 research and innovation program under grant agreement No. 1010216. The computational results presented here have been achieved (in part) using the LEO HPC infrastructure of the University of Innsbruck.

In this paper, we analyze and quantify trellis artifacts and determine characteristics in the frequency distribution of quantized coefficient values. Leveraging these characteristics, we build detectors for trellis artifacts based on analytic modelling and statistical learning. The detectors are intended to serve as forensic preprocessors and can help practitioners that apply forensic tools, to make informed interpretations of their results. The remainder of the paper is organized as follows: Section 2 describes processing steps specific to *mozjpeg* and quantifies their effect on the image signal. Section 3 describes the proposed detectors, and Section 4 evaluates their performance for in- and out-of-distribution scenarios. Section 5 discusses our findings and their implications for the research community before Section 6 concludes our paper.

2. MOZJPEG

Mozjpeg has recently attracted attention within the multimedia security community due to its changes in output images (*e.g.*, [13, 14]). The library implements several compression optimizations to reduce file size and improve the perceptual image quality, namely overshoot deringing, adapted QTs, trellis quantization, and default progressive encoding with optimized scan scripts and Huffman tables. The first three alter the DCT coefficients, whereas the latter optimize the stream encoding without altering coefficients. This section reviews the background of the signal-based optimizations and investigates their effects on coefficients in an isolated manner.

To quantify these effects, we use the *image change rate*, *i.e.*, the share of images with at least one changed DCT coefficient, and the average *coefficient change rate*, *i.e.*, the number of changed DCT coefficients normalized by the number of non-zero DCT coefficients. We use 10 000 never-compressed images of size 512×512 randomly sampled from ALASKA2 [15], the benchmark dataset in steganography. As our reference, we compress these images using *mozjpeg* v4.0.3 with all optimizations disabled. We then selectively enable individual optimizations and measure the image and coefficient change rates compared to our reference. We do this for the quality factors (QFs) 50, 75, 80, 85, 90, 95, and 100.

Overshoot deringing During JPEG compression, the DCT converts blocks of 8×8 pixels into a frequency domain representation. When blocks contain combinations of pixels that cannot be exactly represented by the discretized cosine functions, the DCT causes ringing at the upper (overshoot) and lower (undershoot) value range, also known as the *Gibbs phenomenon* [16]. During decompression, the JPEG decoder clips the positive overshooting values to 255 and negative undershooting values to 0. This results in visual artifacts known as ringing artifacts. They typically appear around sharp edges or text and are visible as alternating light and dark signals.

Starting in version 3, *mozjpeg* implements an overshoot deringing algorithm that tackles the ringing of positively overshooting pixels during compression [17, 18]. It enlarges the upper bound of pixel values and deliberately moves overshoots outside the 8-bit range, hiding ringing waves from the decoder. The allowed overshoot is based on the sharpness of edges. The algorithm extrapolates the pixel in a block with the highest value using *Catmull-Rom* splines.

Effect: 18% of all images from our dataset are changed by the overshoot deringing algorithm. Less than 1% of the DC and AC coefficients in those images are changed. The change rates are largely independent of the QF.

The low change rates can be attributed to the dataset, which contains photographs of natural scenes with few instances of ringing. To highlight the effect of the image content, we repeat this evaluation on images that mainly consist of text. Specifically, we collect 1 000 PDF documents [1] and convert them to the TIFF format using the Python package *pdf2image* with 300 dpi resolution. We center crop them to 512×512 and compress with *mozjpeg*.

Effect: The deringing optimization now changes now changes more than 92% of the images. The coefficient change rate increases to 40%. Again, the change rates are largely independent of the QF.

Quantization tables *Libjpeg* and *libjpeg–turbo* (with one exception [19]) use the QTs for luminance and chrominance components as defined in Annex K of the JPEG standard [20]. While supporting several QTs, including the standard tables, *mozjpeg* implements stronger quantization by default and uses specific QTs [21]. To measure the change rate in images using *mozjpeg*'s specific QTs, we compare them to images compressed using the tables defined in the standard.

Effect: We observe changes in all images except for QF 100, where all entries of the QT are 1 and no images are changed. The AC change rate is between 49% and 60% for the tested QFs. We observe no changes in DC coefficients.

Trellis quantization *Mozjpeg*'s trellis quantization aims to improve the rate–distortion tradeoff after an initial quantization step for each 8×8 coefficient block. It uses a perceptual model to calculate the distortion implied by reducing non-zero DCT coefficients to values of shorter bit sizes. Following [5], we denote y_j^* and y_j^{**} as the coefficient value at subband j before and after trellis quantization, respectively. We define the set

$$\mathcal{C} = \{\pm(2^k - 1) : k = 1, \ldots, 15\}, \tag{1}$$

which leads us to the **candidate values** for a given y_j^*,

$$\mathcal{C}_j = \{c \in \mathcal{C} : |c| < |y_j^*|\} \cup \{y_j^*\}. \tag{2}$$

[1] https://github.com/tpn/pdfs

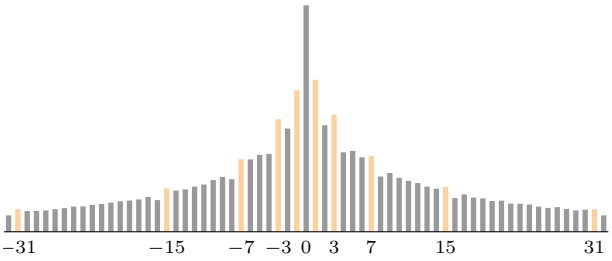

Fig. 1: Candidate values (orange) are amplified in the coefficient histogram of images compressed with trellis quantization. (Data for AC subband 1, compressed with QF 90.)

Moreover, we define the set of **outer neighbors**

$$\mathcal{C}^{++} = \{\pm 2^k : k = 1, \ldots, 15\}. \tag{3}$$

For a given y_j^*, the algorithms evaluates the cost implied by replacing y_j^* with any $c \in \mathcal{C}_j$, weighs the additional distortion against the bits saved by shorter encoding, and sets y_j^{**} to the c with the lowest cost.

Effect: For all QFs, more than 99% of all images contain changes. The average change rate over all AC coefficients is between 10% and 18% for the measured QFs, with a decreasing trend for higher QFs. The opposite trend is observable for the change rate of DC coefficients, which is constant below 5% for QFs up to 90 and exceeds 10% at QF 100.

Figure 1 shows the distribution of quantized AC coefficients after trellis quantization. For natural images, the coefficient distribution can be approximated by Laplacian distributions [22]. Observe, that this is not the case for images compressed with trellis quantization. Here, the probability mass increases for bins of candidate values and decreases for their outer neighbors, the pair of which we call **candidate pairs**.

Note that the effect of trellis quantization is limited when recompressing previously compressed images. We demonstrate this in a simplified example: Let $y_j = 540$ be an unquantized coefficient value and $q_j = 72$ the quantization factor. Quantization divides y_j by q_j to 7.5 and rounds to the nearest integer $y_j^* = 8$. During decompression y_j^* gets dequantized by $y_j^* \times q_j$, resulting in $y_j' = 576$, which is now evenly divisible by q_j. This prevents trellis quantization from modifying the rounding in a direction of fewer bits. In reality, multiple rounding operations during de- and recompression influence the effectiveness.

In [23], the authors uncover the quantization factor by searching for two local minima in the quantization error of recompressed images. It seems intuitive to follow their approach for the detection of trellis quantization and recompress an image with and without trellis quantization before comparing the magnitude of artifacts in the recompressed images. However, as the effectiveness of trellis quantization is limited in previously compressed images, this approach is unsuitable for our means.

3. DETECTORS

In this section, we propose methods based on analytic modelling and statistical learning for detecting trellis artifacts in the distribution of quantized DCT coefficients. We use coefficients of the first eight AC DCT subbands (in zigzag order) with values $i \in \mathcal{I} = \{-32, \ldots, 32\}$. This ensures that our methods generalize to low QFs, where bins with higher absolute values are often unpopulated. Without loss of generality, we consider i as an absolute value and denote the outer neighboring coefficient value as $i + 1$.

To construct the dataset for the detection of trellis artifacts, we use the same sample of 10 000 never-compressed images from the ALASKA2 dataset and compress with *mozjpeg* v4.0.3 with default settings (4:2:0 subsampling, DCT *ISLOW*, *mozjpeg*'s QTs, progressive encoding). We generate two datasets: The negative class are images where all optimizations are disabled during compression. The positive class are images compressed with trellis quantization. We use a $50 : 50$ train–test split.

3.1. Analytic modelling

Our modelling based detection aims to analytically describe the distribution of DCT coefficients of images compressed with trellis quantization. In our measurements in Section 2 we find that for images from the ALASKA2 dataset and coefficient values greater than 2, there are no changes further than from $c + 1$ to c. As changes for the absolute candidate values 2, 1, and 0, are more complicated, we exclude them from the analytical models. We propose two approaches.

Calibration Trellis artifacts resemble in part those of popular steganographic embedding functions. For example, F5 [24] decrements the absolute value of DCT coefficients and inflates the number of zeros. Previous work on the detection of F5 uses *calibration*, which exploits the regularity of the JPEG 8×8 grid. Calibration estimates the histogram of the cover image by cropping a decompressed stego image by 4 pixels on each side and recompresses it using the QT of the stego image [25]. The authors calculate the embedding rate by comparing the histogram of the stego image and the estimated cover histogram. We build on their approach and use calibration to estimate the histogram of an image before trellis quantization. Let H_i be the histogram bin of an image compressed with trellis quantization holding the number of AC coefficients with value equal to i, and $\hat{H}_i$ the respective bin before trellis quantization, as estimated by calibration. We define $H_i \coloneqq \hat{H}_i + \hat{H}_{i+1} \times \alpha_i$, where α_i is the relative frequency of coefficients with value $i + 1$ being changed to i. This results in

$$\alpha_i = \frac{H_i - \hat{H}_i}{\hat{H}_{i+1}}. \tag{4}$$

α_i of 0 refers to the same number of coefficients at bin i before and after calibration, suggesting no changes caused

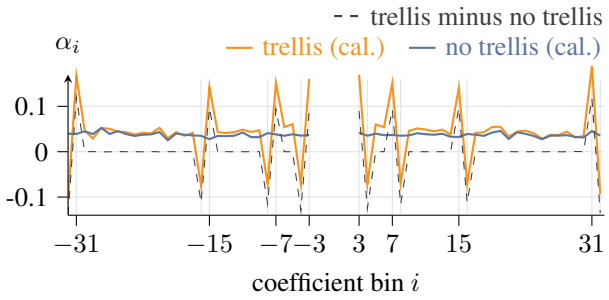

Fig. 2: The relative frequency of coefficients being changed to their inner neighbor. α_i increases at candidate values for images compressed with trellis quantization.

by trellis quantization. A positive α_i and a negative α_{i+1}, at bins where $i \in \mathcal{C}$, indicates the presence of trellis artifacts. As shorthands we write α_c for $\alpha_{i\in\mathcal{C}}$ and α_{c+1} for $\alpha_{i\in\mathcal{C}^{++}}$. (See Eqs. 1 and 3 for the set definitions.)

Figure 2 visualizes α_i as the average over the training set. For images compressed with trellis quantization, we observe high values for α_c and low values for α_{c+1}. For images compressed without trellis quantization, we observe a nearly constant α_i with no deviations at c or $c+1$. The dashed line in Figure 2 plots α_i measured by comparing the same images, compressed with and without trellis quantization, *i.e.*, without applying calibration. The difference between the dashed line and α_i measured for images compressed with trellis quantization is the calibration estimation error. Note that we do not have access to this value when detecting trellis artifacts.

For our purposes we aggregate α_i to α_c and define

$$\alpha_{\mathcal{C}} = \sum_{i\in\mathcal{C}} (\alpha_i - \alpha_{i+1}) \tag{5}$$

as score to detect the presence of trellis artifacts. We use Youden's $\tilde{\mathcal{J}}$ statistic [26] to select the optimal threshold based on the classification performance on the training set.

Vampire neighborhoods In a second approach, we assume a monotonous histogram and measure deviations at candidate pairs with regard to their inner and outer neighbors. We call this set **candidate neighborhoods** $(c_{i-1}, \ldots, c_{i+2})$. We measure the deviation using a *vampire score* β,[2] and define

$$\beta_i = H_i - \frac{H_{i-1} + H_{i+2}}{2} + H_{i+1} - \frac{H_{i-1} + H_{i+2}}{2}. \tag{6}$$

Figure 3 visualizes β_i as the average over the training set. For images compressed without trellis quantization we observe a smooth β_i. For images compressed with trellis quantization we can see spikes at bins where $i \in \mathcal{C}$. This indicates an increased frequency of candidate values and a decreased

[2]The name originates from trellis artifacts in the plotted histogram, which reminded the authors of inverted vampire teeth.

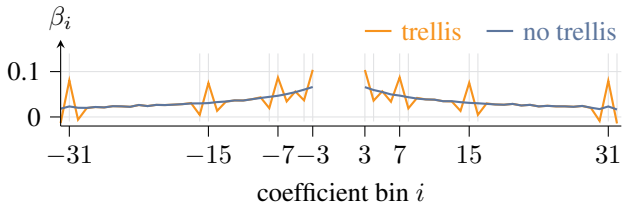

Fig. 3: The average deviation of H_i from a monotonous pattern within neighborhoods. β_i increases at candidate values for images compressed with trellis quantization.

frequency of their outer neighbors with regard to the candidate neighborhood. We aggregate β_i to β_c, simplify Eq. 6, and calculate

$$\beta_{\mathcal{C}} = \sum_{i\in\mathcal{C}} (H_i - H_{i-1} + H_{i+1} - H_{i+2}) \tag{7}$$

as score to detect the presence of trellis artifacts. Again, we find the empirically optimal threshold using Youden's $\tilde{\mathcal{J}}$.

3.2. Statistical learning

Statistical learning refers to the use of machine learning to infer patterns from training images. We evaluate three different types of features extracted from the DCT coefficients. The features are then classified using an ensemble of Fisher linear discriminant base learners [27], which is commonly used in steganalysis [28].

Cartesian calibration Like before, we use calibration to estimate the histogram before trellis quantization. Next, we extract candidate neighborhood features of calibrated images and use them together with the same set of histogram features from the original images to train the classifier on the cartesian product. There are ten candidate values in $\mathcal{I}$. The resulting features have dimensions of $4 \times 8 \times 10 \times 2 = 640$.

Vampire neighborhoods As before, we train the classifier on candidate neighborhood features, this time without cartesian calibration. The resulting features have dimensions of $4 \times 8 \times 10 = 320$.

JRM features As a third approach, we use features extracted from an ensemble of JPEG Rich Models [29] (JRM) with 11 255 dimensions to train a classifier. JRMs model different types of dependencies between adjacent coefficient subbands.

4. RESULTS

Table 2 reports the accuracies on the test set. Both analytic detectors perform well for high QFs but degrade for low QFs. The decline in accuracy for low QFs is also visible, albeit less prevalent, for the detectors based on statistical learning. They

Table 2: Detection accuracies of proposed trellis detectors.

	Analytic detectors		Learning detectors		
QF	cal.	vamp.	cal.	vamp.	JRM
100	95%	99%	100%	100%	100%
95	84%	92%	99%	99%	100%
90	80%	88%	99%	98%	100%
85	75%	82%	99%	97%	100%
80	72%	79%	98%	96%	100%
75	71%	76%	98%	96%	99%
50	72%	75%	97%	93%	98%

all achieve high accuracies. The detector trained on JRM features reaches near-perfect test accuracy, also for low QFs.

4.1. Robustness

Out-of-distribution scenarios occur when the distribution of coefficients in test images differs from that in the training images. In this section, we report the robustness in the following scenarios: detectors tested on images of a different QF than the training images, detectors tested on stego images while trained on covers, detectors tested on double-compressed images while trained on single-compressed, and images that were compressed with the deringing optimization.

Unseen QFs We find that the detectors generalize well to higher QFs. For lower QFs, the performance decreases slightly due to missed detections. The detector based on JRM features is an exception. It does not generalize well to lower QFs, especially when trained on QFs above 90. For all following out-of-distribution experiments, we focus on our detectors based on vampire neighborhoods (without calibration). The positive class $\oplus$ contains images compressed **with** trellis quantization, and the negative class $\ominus$ contains images compressed **without** trellis quantization. We apply processing operations to either the positive or the negative class and report the effect on the performance in Table 3. The reference is the in-distribution performance on images of QF 90.

Steganography In Section 1, we show that a state-of-the-art steganalysis model fails when facing images containing trellis artifacts. Now we investigate the opposite scenario, namely whether our trellis artifact detectors are robust to steganography. We evaluate them for three prominent embedding methods, nsF5, UERD, and J-UNIWARD, with an embedding rate of 0.4 bpnzAC. We assume that a high embedding rate increases the difficulty of identifying trellis artifacts.

Exp. 1: $\oplus$ **trellis** $\ominus$ **no trellis, stego**
Both detectors differentiate between the positive and the negative class with the same performance as before. They are robust against stego embeddings in images without trellis artifacts.

Table 3: Robustness of two detectors based on candidate neighborhoods to deviations in distributions. The effect is measured as the performance difference in %-pts. Reference in-distribution performance for QF 90 is given at the top.

	Analytic detector			Learning detector		
	Acc.	FPR	FNR	Acc.	FPR	FNR
Ref.	88.48	12.40	10.60	98.18	1.62	2.02
Exp. 1: Steganography in $\ominus$ no effect						
Exp. 2: Steganography in $\oplus$						
nsF5	− 3		+ 5	− 0		+ 4
UERD	− 7		+14	− 4		+ 9
J-UNI.	− 9		+18	− 9		+20
Exp. 3: Double compression in $\ominus$ (QF_1: 90)						
QF_2: 93	−37	+75		−49	+98	− 0
QF_2: 90	−44	+88		+ 1	− 1	− 0
QF_2: 87	+ 5	−11	−2	+ 1	− 2	
QF_2: 75	+ 2	− 5		−36	+72	+ 0
Exp. 4: Double compression in $\oplus$ (QF_1: 90)						
QF_2: 93	+ 4		− 8	+ 0	+ 2	− 1
QF_2: 90	+ 0		+ 0	+ 1	− 1	− 1
QF_2: 87	−14		+29	−48		+97
QF_2: 75	− 2		+ 4	+ 2	− 0	− 3

$\oplus$ positive class (trellis) $\ominus$ negative class (no trellis)

Exp. 2: $\oplus$ **trellis, stego** $\ominus$ **no trellis**
The performance of our detectors drops slightly due to an increase in missed detections. While the embedding with nsF5 has little effect on our detectors, the embeddings with UERD and J-UNIWARD seem to wash out trellis artifacts. However, at least 70% of all images from $\oplus$ are still correctly classified by the analytic detector and 78% by the learning-based detector. Note that this is a hypothetical experiment. No practical steganographic tool we know of uses trellis quantization during compression.

Double compression artifacts Double compression causes periodic artifacts and discontinuities in the coefficient distribution. To evaluate our detectors, we use images compressed with QF_1=90 and recompress them with QF_2. We evaluate the detectors trained on single compressed images of QF_2.

Exp. 3: $\oplus$ **trellis** $\ominus$ **no trellis, double compression**
When $QF_2 > QF_1$, the performance of both detectors drops. As for the analytic detector, $\beta_{\mathcal{C}}$ of the negative class now roughly resembles the pattern of trellis artifacts at some candidate values, causing the performance to decrease to 55%. We observe the same for QF_2=90 with a drop to 45%. The learning-based detector fails for QF 93 (acc. = 50%) but is robust against double compression with $QF_2 = QF_1$. When $QF_2 < QF_1$, $\beta_{\mathcal{C}}$ follows a different pattern. This amplifies

the differences between the classes and slightly increases the performance. Interestingly, the performance of the learning based detector decreases for QF_2=75.

Exp. 4: $\oplus$ **trellis, double compression** $\ominus$ **no trellis** Again, double compression with $QF_2 > QF_1$ causes $\beta_{\mathcal{C}}$ to resemble the pattern of trellis artifacts. However, in this case, it happens in the positive class, which amplifies trellis artifacts. The performance of both detectors increases slightly. Respectively, $\beta_{\mathcal{C}}$ for $QF_2 < QF_1$ follows a different pattern than trellis artifacts; now, concealing them. This leads to missed detections of both detectors for QF_2=87. Interestingly, double compression with QF_2=75 has close to no effect.

To investigate if the results of Exp. 3 and 4 impair the reliability of double compression detection, we apply the pre-trained double compression detection model DJPEG-torch [30] on images compressed with trellis quantization. DJPEG-torch uses histogram features and extracted quantization tables as input to a convolution neural network. We find that it is robust, also to images where double compression amplifies trellis artifacts.

Overshoot deringing artifacts To ensure the reliability on images from *mozjpeg*, we measure the effect of overshoot deringing artifacts on our detector. We use the ALASKA2 dataset.

Exp. 5: $\oplus$ **trellis** $\ominus$ **no trellis, deringing** ,
Exp. 6: $\oplus$ **trellis, deringing** $\ominus$ **no trellis**
Overshoot deringing does not affect on the performance of our detectors for the tested QFs. For the sake of space, we do not include this result in Table 3.

5. DISCUSSION

In this paper, we find that state-of-the-art steganalysis models misclassify innocuous cover images when they are unaware of trellis artifacts. To address this, we propose methods based on analytic modelling and statistical learning to detect trellis artifacts in compressed JPEG images. The detectors are intended to help practitioners applying forensic tools to make informed interpretations of their results and avoid unexpected behavior of tools tailored for different libraries when analyzing images compressed with *mozjpeg*.

Our detectors are robust against steganographic embeddings of three popular embedding methods and artifacts from *mozjpeg*'s overshoot deringing algorithm. We find that double compression operations can diffuse trellis artifacts, causing our detectors to fail.

The characteristic of double compression artifacts in an image can reveal information about the history of an image and potential manipulations. We find that the effectiveness of trellis quantization is limited in previously compressed images. Future research should analyze whether this can be exploited during the detection of manipulations in images compressed with trellis quantization.

Mozjpeg's overshoot deringing algorithm introduces changes in approximately 18% of images capturing natural scenes; however, it changes only 1% of the coefficients. In a dataset of JPEG compressed computer graphics and text, where there are more instances of ringing, it changes up to 40% of the coefficients in 90% of the images. This can have implications for other fields, *e.g.*, the detection of sharpening, where the absence [31] or characteristics [32] of ringing artifacts are used as a telltale for image manipulation.

Our detectors complement previous efforts to fingerprint JPEG libraries. Existing approaches investigate implementation differences in common processing steps, such as DCT [33], chroma subsampling [34], and rounding operations during quantization [35]. Furthermore, [36] leverage the statistical features of recompressed images and [37] use rounding errors of decompressed images. Apparent traces to fingerprint *mozjpeg* are library-specific QTs, and image-specific scan scripts and Huffman tables in progressive images. We concentrated our focus on the image signal, as these parameters can be configured by the user during compression, making them unreliable for the detection of *mozjpeg*.

6. CONCLUSION

It is important to understand optimizations of popular JPEG implementations as many methods in multimedia security rely on subtle traces in the signal originating from compression and decompression operations. Researchers proposing learning-based methods for steganography, steganalysis, or image forensics should include images compressed with *mozjpeg* in their evaluation protocol, and revisit known methods in the light of trellis quantization.

Finally, practitioners should be careful when carrying out forensic tests on images of unknown sources using tools tailored to specific libraries.

7. ACKNOWLEDGEMENTS

We thank Benedikt Lorch and Rainer Böhme for their support and valuable comments on the draft, and Martin Beneš for incorporating *mozjpeg* into his `jpeglib` library.

8. REFERENCES

[1] M. W Marcellin and T. Fischer, "Trellis coded quantization of memoryless and gauss-markov sources," *IEEE Transactions on Communications*, pp. 82–93, 1990.

[2] J. Wen, M. Luttrell, and J. Villasenor, "Trellis-based RD optimal quantization in H.263+," *IEEE Transactions on Image Processing*, pp. 1431–1434, 2000.

[3] M. Marcellin, M. Lepley, A. Bilgin, T. Flohr, T. Chinen, and J. Kasner, "An overview of quantization in JPEG 2000," *Signal Processing: Image Communication*, pp. 73–84, 2002.

[4] Mozilla Foundation, "MozJPEG: Improved JPEG encoder," `github.com/mozilla/mozjpeg`, 2014, Accessed on 25 Jan, 2024.

[5] N. Hofer and R. Böhme, "Progressive JPEGs in the wild: Implications for information hiding and forensics," in *IH&MMSec*. 2023, pp. 47–58, ACM.

[6] A. Piva, "An overview on image forensics," *ISRN Signal Processing*, pp. 1–22, 2013.

[7] M. Kharrazi, H. Sencar, and N. Memon, "Blind source camera identification," in *ICIP*. IEEE, 2004, pp. 709–712.

[8] Y. Yousfi, J. Butora, J. Fridrich, and C. Fuji Tsang, "Improving EfficientNet for JPEG steganalysis," in *IH&MMSec*. 2021, pp. 149–157, ACM.

[9] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for CNNs," in *ICML*. PMLR, 2019, pp. 6105–6114.

[10] J. Fridrich, T. Pevný, and J. Kodovský, "Statistically undetectable JPEG steganography: Dead ends challenges, and opportunities," in *MM&Sec*. 2007, pp. 3–14, ACM.

[11] L. Guo, J. Ni, W. Su, C. Tang, and Y. Shi, "Using statistical image model for JPEG steganography: Uniform embedding revisited," *IEEE TIFS*, pp. 2669–2680, 2015.

[12] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP*, pp. 1–13, 2014.

[13] J. Butora, P. Puteaux, and P. Bas, "Errorless robust JPEG steganography using outputs of JPEG coders," *IEEE TDSC*, 2023.

[14] S. McKeown, G. Russell, and P. Leimich, "Fingerprinting JPEGs with optimised huffman tables," *JDFSL*, 2018.

[15] R. Cogranne, Q. Giboulot, and P. Bas, "The ALASKA steganalysis challenge: A first step towards steganalysis," in *IH&MMSec*. 2019, pp. 125–137, ACM.

[16] D. Gottlieb and C. Shu, "On the gibbs phenomenon and its resolution," *SIAM review*, pp. 644–668, 1997.

[17] L. Kornel, "Deringing via overshoot clipping," 2014, https://github.com/mozilla/mozjpeg/pull/101.

[18] T. Richter, "JPEG on steroids: Common optimization techniques for JPEG image compression," in *ICIP*. 2016, pp. 61–65, IEEE.

[19] M. Beneš, N. Hofer, and R. Böhme, "Know your library: How the libjpeg version influences compression and decompression results," in *IH&MMSec*, 2022, pp. 19–25.

[20] G. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, pp. xviii–xxxiv, 1992.

[21] N. Robidoux, "Re: Better JPEG quantization tables?," 2013, Legacy ImageMagick Discussions Archive.

[22] R. Reininger and J. Gibson, "Distributions of the two-dimensional DCT coefficients for images," *IEEE TCOM*, pp. 835–839, 1983.

[23] H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE TIFS*, pp. 154–160, 2009.

[24] A. Westfeld, "F5 – a steganographic algorithm: High capacity despite better steganalysis," in *IH*. Springer, 2001, pp. 289–302.

[25] J. Fridrich, M. Goljan, and D. Hogea, "Steganalysis of JPEG images: Breaking the F5 algorithm," in *IH*. Springer, 2003, pp. 310–323.

[26] W. Youden, "Index for rating diagnostic tests," *Cancer*, pp. 32–35, 1950.

[27] R. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. of Eugenics*, pp. 179–188, 1936.

[28] J. Kodovský, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE TIFS*, pp. 432–444, 2011.

[29] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE TIFS*, pp. 868–882, 2012.

[30] J. Park, D. Cho, W. Ahn, and H. Lee, "Double JPEG detection in mixed JPEG quality factors using deep convolutional neural network," in *ECCV*, 2018, pp. 636–652.

[31] G. Cao, Y. Zhao, and R. Ni, "Detection of image sharpening based on histogram aberration and ringing artifacts," in *ICME*. IEEE, 2009, pp. 1026–1029.

[32] G. Cao, Y. Zhao, R. Ni, and A. Kot, "Unsharp masking sharpening detection via overshoot artifacts analysis," *IEEE Signal Processing Letters*, pp. 603–606, 2011.

[33] S. Agarwal and H. Farid, "Photo forensics from rounding artifacts," in *IH&MMSec*, 2020, pp. 103–114.

[34] B. Lorch and C. Riess, "Image forensics from chroma subsampling of high-quality JPEG images," in *IH&MMSec*, 2019, pp. 101–106.

[35] S. Agarwal and H. Farid, "Photo forensics from JPEG dimples," in *WIFS*. IEEE, 2017, pp. 1–6.

[36] N. Bonettini, L. Bondi, P. Bestagini, and S. Tubaro, "JPEG implementation forensics based on eigen-algorithms," in *WIFS*. IEEE, 2018, pp. 1–7.

[37] J. Butora and P. Bas, "High quality JPEG compressor detection via decompression error," in *GRETSI*, 2022.