Effects and Solutions of Cover-Source Mismatch in Image Steganalysis

Quentin Giboulot, Rémi Cogranne, Dirk Borghys, Patrick Bas

Available online at www.sciencedirect.com
ScienceDirect
Signal Processing: Image Communication 00 (2020) 1–16
Signal Processing: Image Communication

Effects and Solutions of Cover-Source Mismatch in Image Steganalysis

Quentin Giboulot [+*], Rémi Cogranne[+], Dirk Borghys[†], and Patrick Bas [⋆]

[+] Lab. of System Modeling and Dependability - ROSAS Dept. - Troyes University of Technology, Troyes, France
[†] Department of Mathematics, Royal Military Academy, Brussels, Belgium
[⋆] CNRS, CRIStAL Lab., École Centrale de Lille, Lille, France

Abstract

The Cover-Source Mismatch (CSM) has been long recognized as a major problem in modern steganography and steganalysis. Indeed, while a vast majority of works in steganography and steganalysis had been tailored to a specific reference database, namely BOSSbase, recent works show that, because of CSM, the results may greatly differ when changing this dataset. Although the CSM has already been the subject of several publications, these prior works investigated only a few elements in a limited setup. The goal of the current paper is to study the effects of the CSM in a more comprehensive manner and then to examine and compare different strategies for mitigating it. It first defines two different parameters, the source difficulty and the source inconsistency, which are involved in the CSM. Then, using different steganographic schemes and feature sets, it aims at providing a systematic study regarding the various factors that can give birth to CSM for image steganalysis. Finally, two practical ways to mitigate the CSM, using training techniques promoting either diversity of different sources or the specificity of one targeted source which is beforehand identified by training a multi-class classifier, are presented and their performances are compared for different training set sizes.

Keywords: Steganography, Steganalysis, Cover-Source Mismatch, Image processing, Image Heterogeneity

1. Introduction

Over the past decades, steganography and steganalysis have been largely improved. While the former essentially consists in hiding data within innocuous-looking digital media, the latter mostly aims at detecting the very existence of hidden messages among a set of inspected media. Those two fields constitute a cat and a mouse game in which the steganographers try evading the possible detection tools from steganalyzers. On the opposite, the steganalyzers have been working to improve the results of their inspection by designing more and more accurate detection methods.

Besides, over the past two decades the scientific community slowly organized to provide all materials for reproducibility which is fundamental to perform fair benchmarks. This effort culminated with the BOSS competition which was designed to benchmark both a steganographic scheme, namely Hugo [34]; this BOSS steganalysis contest gives birth to well-known steganalysis methods, namely the rich models paradigm with Ensemble classifier [13, 26], which has been used as a reference for several years. The use of the database originating from the contest, aka BOSSBase [2], has been widely adopted by the community for obtaining reproducible and comparable research. However, the use of this database for steganalysis leads to two important drawbacks:

1) BOSSBase is very far from the type of image sets that would be analyzed by a forensic investigator in an operational situation of steganalysis. Indeed, the diversity of the images presented in BOSSBase is extremely poor: let us recall briefly that the BOSS dataset is based on raw images shot by 7 different cameras and developed using exactly the same process. This processing pipeline includes mainly the same demosaicking algorithm (ppg) and downscaling to the same size of 512 pixels for smallest dimension. Therefore, using BOSSBase images, "as is", in a "test tube", brings steganalysts in an environment that hardly reflects the diversity of processing operations, sensors and contents that one would face by analyzing images from the Internet such as the photo-sharing website FlickR.

2) The use of BOSSBase also had a negative effect which

[*] Corresponding author.

[**] This work has been funded in part by the French National Research Agency (ANR-18-ASTR-0009), ALASKA project: https://alaska.utt.fr, by the French ANR DEFALS program (ANR-16-DEFA-0003). Quentin Giboulot PhD Thesis is also funded by DGA project 2018346.

Email addresses: quentin.giboulot@utt.fr (Quentin Giboulot [+]), remi.cogranne@utt.fr (Rémi Cogranne[+]), Dirk.Borghys@rma.ac.be (Dirk Borghys[†]), Patrick.Bas@centralelille.fr (and Patrick Bas [⋆])
URL: http://lm2s.utt.fr/en/members/cogrannr.html (Rémi Cogranne[+])

has not been immediately observed. Indeed, practitioners start slowly and slowly to develop steganographic and steganalysis methods tailored to this dataset. However, designing embedding schemes and detection tools on a very specific dataset does not guarantee their generalization over different images. Even worse, it has been observed in [38] that this may lower the results on different image datasets.

From these two problems we can state that most of the steganalysis works in the literature have been conducted *in Vitro*, i.e. using laboratory conditions. Such setup is too restrictive and very far from real-life scenarios in which the practitioner has to carry out the steganalysis without any information on the image processing tools that have been used and, in the best case, only a very approximative knowledge of both the embedding scheme and payload. When those parameters are partially unknown, the main outcome may be an important impact of the Cover-Source Mismatch (CSM). Even worse, as we will be shown in the next Section 2 through a deep review of current art, prior works offer a very narrow view on CSM. In particular, most of those works study few elements in a limited setup without any quantitative comparison.

Contrasting the present paper with prior works, one can note that, first, many of the parameters that constitute major sources of CSM, namely those related to the image processing pipeline, have never been studied. In addition, all prior works studied only one specific aspect of the CSM, typically, the impact of camera model alone or the effect of resizing. However, all those sources of CSM were never studied together and quantitatively compared to each other. In this context, one would easily understand that the roots of CSM, the impact of the various possibles causes are not assessed in a comprehensive manner and, consequently, the solutions to address this problem can hardly be assessed in a general way.

The overall goal of this paper is to bring steganalysis closer to *in Vivo* setups by providing a rigorous analysis of the CSM. The contributions of this paper can be divided into five parts:

- A comprehensive review of prior works on CSM, highlighting their specificities, originality and limitations (Section 2). This state-of-the-art especially points out the limited evaluation of the different roots of CSM and the lack of general assessment.

- The CSM is defined clearly from a statistical signal processing perspective, and the possible outcomes of the CSM (namely the source difficulty and the source inconsistency) are then identified in Section 3. We propose qualitative definitions and propose possible quantitative evaluations emphasizing the difficulty to measure objectively this phenomenon.

- The most important effects of the CSM are identified throughout a systematic study of all processing steps from image acquisition up to JPEG compression. By performing a wide range of targeted experiments using different parameters related to the acquisition (Section 5), image development parameters (Section 6), and JPEG compression (Section 7) the impact of each processing step on CSM is evaluated separately.

- We also include a brief quantitative study on the semantic content (Section 8) to show the relative impact it may have when compared to the processing pipeline.

- Finally, though present paper does not aim at designing a novel method to cope with CSM, we present, assess and compare two strategies for this purpose (Section 9).

2. Prior works

In order to perform *in Vivo* steganalysis, the first requirement is to alleviate the Cover-Source Mismatch and to design steganalysis schemes that can cope with this problem. We present here the prior investigations and possible solutions related to the CSM. All have their merits, however, we claim that from those different prior contributions, no general settings or global explanation were provided regarding to the CSM.

To the best of our knowledge, CSM has been mentioned for the first time in [16] which shows how the use of images from two different cameras, of the same model, during the training and testing steps may lead to a great drop of steganalysis performance (as compared to the case when images from the very same camera device are used for both training and testing). Similarly, a rather experimental comparison of the so-called “structural” steganalysis methods, which are not based on machine learning, has been proposed in [5] and emphasized how results greatly depend on the image dataset used. The authors especially highlighted the high difficulty to provide a fair comparison between different steganalysis techniques since their ranking, in terms of detection accuracy, changes significantly over various datasets used for assessment.

During the BOSS competition this issue was also raised ; the organizers added, only in the testing set, images from a camera that had not been used for creating the training dataset [2] (those images were actually decompressed JPEG whereas other images came from RAW). Results from all competitors with highest scores were significantly worse on those images. The Cover Source Mismatch has been further identified in the white paper [22] as a fundamental problem for using the academic works “into the real world”.

The works presented in [27] show how much the performance of a steganalysis method depends on the camera model, thus implicitly assumed that this is the most influential factor on CSM. Interestingly, the authors also show that training a classifier over a diversified set containing images from all camera models allows substantially reducing the impact of the CSM.

The effect of the image processing pipeline was also studied in [25] to show the effect of image resizing method. It has been shown experimentally that when downsampling images, the choice of the interpolation kernel as well as the downsizing factor may greatly influence the performance of a steganalyzer. Very similarly, the studies presented in [38] show that when images are downsized by cropping, instead of downsampling, the steganalysis performance may significantly change. Interestingly, the authors ranked several embedding schemes among the state-of-the-art and showed that, depending on the downsizing parameter, this ranking may be flipped.

A notable prior work proposes a different perspective of the CSM. The starting point of the work presented in [23], and later enlarged in [21], is to identify among various actors, that send or share digital images, which are the ones that use steganographic methods. In these works a source is defined as an actor, as we will see in the sequel, this is may be closely related to the definitions we proposed in this paper.

Our previous works [3] and [15] have been the first to study the impact of the processing pipeline, from the acquisition of a raw image up to the JPEG compression. The works presented in [3] focused on the impact of the demosaicking algorithm and used the old but readily available JPHide[1] embedding scheme. On the other hand, the works presented in [15] focused on the image processing tools (denoising and sharpening) and showed that the cover-source mismatch is generally less important when image processing tools give birth to similar statistical properties in the ensuing image noise.

Mitigating the cover source mismatch remains challenging and only a very few previous works studied this problem ; they can be divided into three different strategies:

1. The *atomistic approach* consists in splitting a large image dataset into small subsets with similar properties. The most successful works among this category have been referred to as "forensics-aided steganalysis", see for instance [1] and [19] in which the authors used methods from digital image forensics to create clusters of "homogeneous datasets". This approach seems natural once it is observed that the detection accuracy may greatly change depending upon the source over which the learning is carried out. In fact, the idea of creating "comparable" sources has been applied empirically from the very first works on Cover Source Mismatch [16].
2. The *holistic approach*, on the opposite, consists in blending together as many sources as possible in order to allow the steganalysis to learn the most general steganalysis rules that, hence, will loosely depend on the source. This approach has been explicitly presented in [30] and later used by the authors in [23, 21, 33].
3. Eventually, few prior works have been proposed based on the concept of "*transfer learning*", see [29, 28] for instance. This concept consists in using a dataset from a different domain (source) for training than for testing and to transfer both sets into an invariant domain in order to mitigate the mismatch. Note, however, that one drawback of this method is that the learning of this transfer requires a whole set of images and, hence, cannot be computed for only one test image from an unknown source. This solution is interesting, however, when one does not want to retrain a given classifier.

The present work aims at assessing in a systematic manner the impact of the whole development pipeline on the CSM. More generally, it is aimed at offering a systematic approach for evaluation of the main parameters, from image acquisition to the final JPEG compression, that influence the capacity of images to convey hidden data. We thus study a large range of possible changes in the whole digital imaging pipeline in order to focus, in the present paper, on the steps that have the most important impact on the CSM to show how those steps impact both steganalysis accuracy and CSM.

The paper is organized as follows: Section 3 proposes a formal definition of what a source is, what Cover-Source Mismatch is and how to assess it. Section 4 presents all aspects of the experimental setup used in the present paper. Then, Section 5 assesses the impact on CSM of the first step of image acquisition: the physical imaging device. The second step of image processing and its impact on CSM is studied in Section 6. The last part related to JPEG compression is studied in Section 7. In section 8 the influence of image semantics is examined. Eventually possible ways to mitigate CSM are presented and assessed in Section 9. Section 10 concludes the present paper.

3. Defining the Cover Source Mismatch

3.1. Rationale and motivational example

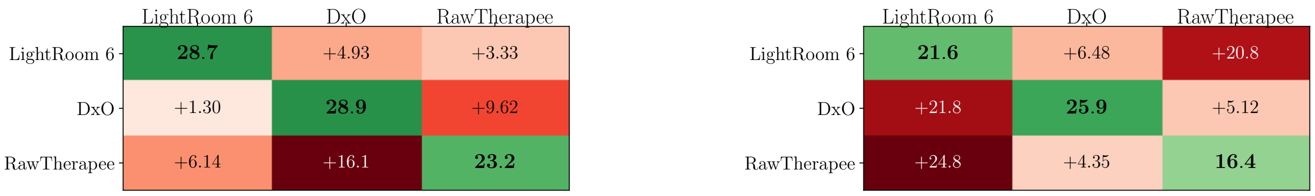
Our rationale is based on the observation that the development pipeline may dramatically impact the statistical distribution of ensuing image elements, pixels or DCT coefficient for JPEG images, [11, 39]. To demonstate visually this phenomenon, we propose a small experiment, using three different image developement and processing softwares (RawTherapee, LightRoom and DxO Photolab) that highlights two fundamental impacts of raw image developement on steganalysis.

On the one hand, Figure 1 shows the impact of the image-processing software on steganalysis for two different embedding algorithms (nsF5 and J-UNIWARD) and two different feature sets (cc-JRM and DCTR). One can observe that the software used for processing raw images has an important impact on the performance of the classifier (the diagonal of matrices) and the mismatch (the off-diagonal error rates).

On the other hand, another experiment emphasizes the high discrepancy of the statistical distributions of DCT coefficients after developement of the very same raw images, made of i.i.d Gaussian random variables, using the three aforementioned softwares: Figure 2 contrasts the joint empirical distribution of the two DCT coefficients $(7, 0)$ and $(0, 7)$ belonging to the same block before quantization. On this Figure 2, one can notice that these distributions are also very different, both marginally and jointly.

One can also note that firstly the rightmost Figure 2(c), associated to *RawTherapee*, does not exhibit any significant correlation between the chosen DCT cofficients and secondly a smaller variance which makes this joint distribution very different from the one obtained with the other softwares. Interestingly, when

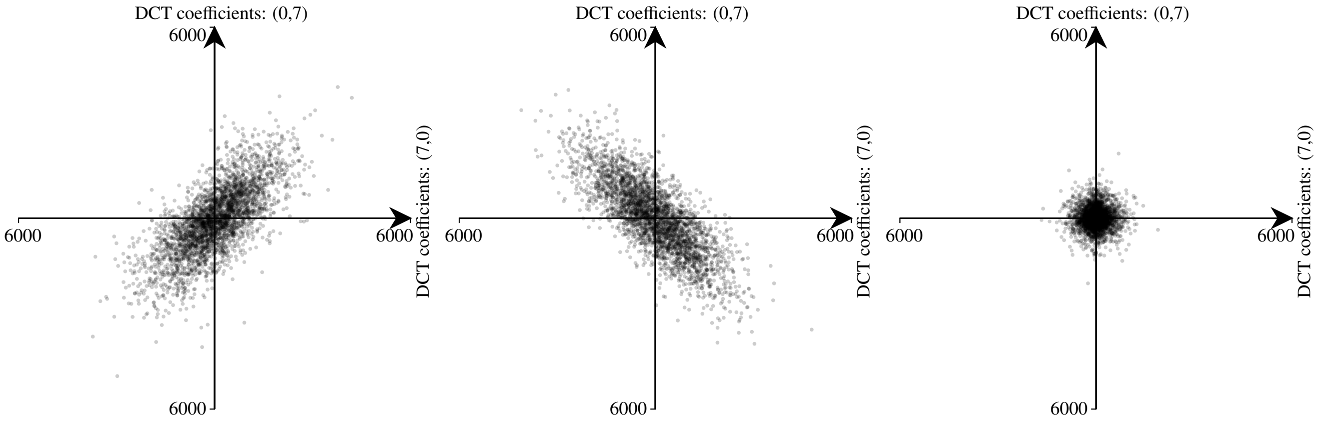
[1] JPHide, which is also referred to as JPHide&Seek and JpegHide, is available at: `https://github.com/h3xx/jphs`.



(a) cc-JRM feature set, nsF5 embedding scheme.

(b) DCTR feature set, J-UNIWARD embedding scheme.

Figure 1: Steganalysis error rate, P_E, as a function of the software used to develop raw files to jpeg images. Each software uses its own set of algorithms, with specific implementation. Each row denotes a constant training set and each column denotes a constant testing set.



(a) Image developed with *DxO PhotoLab*.

(b) Image developed with *Adobe Lightroom*.

(c) Image developed with *RawTherapee*.

Figure 2: Scatter plots that show the empirical joint distribution of unquantized DCT coefficients $(0,7)$ and $(7,0)$ for images corrupted with only i.i.d Gaussian noise and then developed with three different software, namely *DxO PhotoLab* (left) *Adobe Lightroom* (center) and open-source *rawtherapee* (right).

looking at the impact on steganalysis from Figure 1, we notice that this software produces the largest mismatch with the other softwares.

This first motivational example points out to impact on CSM in steganalysis that is due to the development channel, which generates a strong statistical footprint on the image components. We shall see that the results presented in this paper will confirm this rationale.

However, to precisely model the relationship between noise statistical properties and steganalysis, a comprehensive quantitative assessment explaining how each step of the processing pipeline can impact the image statistical distribution and contribute to the CSM is currently missing. This paper aims mainly at providing such a detailed assessment of raw image developement and processing on CSM in steganalysis.

3.2. Definitions

Before analyzing more deeply the CSM, we have to define accurately what a *source* is and what exactly the CSM problem is. In order to state the problem of CSM we propose *qualitative* definitions that try to be as general as possible. Additionally, because we want to provide insights for practionners, we adopt the point of view of the steganalyst and provide *quantitative* metrics that allows to measure in a practical case the CSM between two *sources*. Keeping this in mind, we propose the following definitions:

Definition 1 (of a Source)**.** *A* source *can be defined as a device combined with a set of algorithms that generate cover contents such that for a given semantic content, the succession of acquisitions forms a stationary signal.*

For example, for digital images, a source is a camera with fixed acquisition settings and processing pipeline such that, for a given captured scene, each pixel outputs a stationary signal, following the same statistical distribution.

We will consequently consider that one camera using two different ISO parameters will generate two distinct sources, or that one camera using different JPEG quality factors during the development pipeline will also generate two distinct sources (see Figure 3) since in both cases the underlying distributions of the generated images are modified.

Definition 2 (Cover-Source Mismatch, CSM)**.** *The* Cover-Source Mismatch *is the fact that when using two different sources for training, the learning outcome differs significantly while the set of embedding parameters (same algorithms, same embedding rate, ...) and steganalysis method are the same. The Cover-Source Mismatch is particularly striking when the sources, used to generate training and testing sets, differ.*

As recalled in section 2, the CSM was noticed in multiple practical scenarios such as, for example, during the BOSS contest where the practical accuracy on the test set was very different from the accuracy on the training set due to images coming from a different source in the test set (a camera model not present in the training set and generating JPEG images). Note that CSM can also occur for other learning tasks than classification, for example regression or clustering, but for the following definitions we focus on supervised learning.

As pointed out in the comprehensive state-of-the-art provided in Section 2, most prior works measure the CSM using the following protocol: with two sources A and B, the steganalyzer trains a dedicated classifier for source A on a given training subset. Then the steganalyst measures the detection accuracy for the classifier on source A and on source B and the CSM is thus *quantitatively* defined as the difference between the detection accuracy obtained when training on the target source and when applying the classifier on a different testing set from the source B. As we shall see in the present paper, this procedure is flawed because it mixes several aspect of the same phenomenon and does not allow to distinguish them. First of all, the CSM can be due to two fundamentally different factors which are the source *inconsistency* , w.r.t another source, and the source *difficulty*. Those two factors are fundamental to measure how much the Cover-Source Mismatch between sources is important. These two notions are defined as follows.

Definition 3 (Source Inconsistency)**.** *The inconsistency between sources A and B appears when one trains a function over the source A and obtains a different prediction error as compared to one obtained when training on the source B. Informally this corresponds to a lack of generalization of the prediction rule.*

The *inconsistency between two sources* may be used as a measure of the amount of Cover-Source Mismatch. However *source inconsistency* is, alone, not sufficient to provide such a measure because one must also take into account the fact that a steganalysis method may have different detection performances over two sources, even when trained and tested without CSM. Therefore, for being comprehensive we need to define the *intrinsic difficulty* of a source.

Definition 4 (Source Intrinsic Difficulty)**.** *The* intrinsic *difficulty can be defined practically as the prediction error (for example P_E, defined in Eq.* (1) *for binary classification when both the training set and the testing set comes from the same source. The more important the prediction error, the larger the difficulty.*

To better illustrate these two different concepts and to explicitly define how to measure those quantities w.r.t. to a given steganalysis method, table 1 presents an example when training and testing on two different sources and measuring the classification score in terms of P_E. For this table, as well as for many that will be presented in present paper, each column correspond to the classification error when testing on a given source and as a function, in row, of the source used for training.

The *source intrinsic difficulty* is simply the score in the matched case, which can be read on the diagonal of the table. In our example, source B has a larger difficulty (0.35) than source A (0.2). Consequently, the difference in terms of detection accuracy can be used as a measure of the difference between sources intrinsic difficulty (0.15 in the present example).

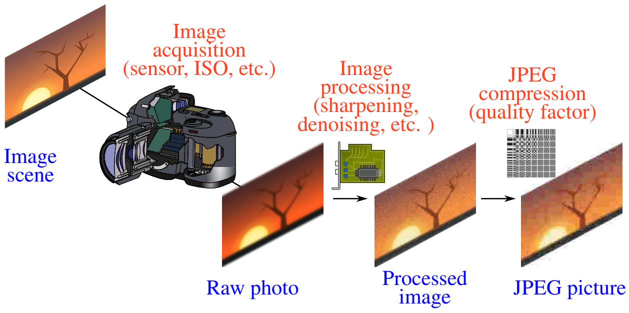


Image acquisition (sensor, ISO, etc.)
Image processing (sharpening, denoising, etc.)
JPEG compression (quality factor)
Image scene
Raw photo
Processed image
JPEG picture

Figure 3: Illustration of the image acquisition from a scene up to JPEG compression. This process is divided into three main steps which are (1) image acquisition (2) image rendering and processing, and (3) JPEG compression. A source is modified for example by changing at least one parameter per step presented in parentheses.

In the present paper, the *inconsistency* is used to measure the impact of the training sources on the detection accuracy. This quantity is measured by simply subtracting every score in the column of interest by the testing source intrinsic difficulty.

In the present example, the *inconsistency* when testing on source A while training on source B, is characterized by an increase of the P_E from 0.2 to 0.33. When the learning process in reversed (training on A, testing on B), the increase in P_E is only from 0.35 to 0.38. One can thus conclude that the inconsistency is larger in the former case as it leads to an increase of the P_E by 11% while it increases by only 3% in the latter case.

Contrary to the previous more general *qualitative* definitions, this example as well as all the results presented in this paper, requires a *quantitatively* defined metric to assess those aspects in practice. Since we adopt a practical point of view, we will have to rely on the detection accuracy of an empirical detector to measure both *inconsistency* and *intrinsic difficulty*. However, using the accuracy of a practical classifier, based upon a specific features set, has a fundamental limitation: it depends on this specific detector and may not always allow to draw general conclusions. For instance, the detector can be very sensitive to a small change in the processing pipeline (e.g. when switch the order of image processing operations). In such a case the quantitative evaluation will show a large inconsistency for similar sources. On the other hand, the very same detector might be robust to a particular parameter, e.g. the quantification matrix in JPEG compression. One who use this detector will measure a small inconsistency while the two sources are quite different. As such, *quantitative* measures of the difference between two sources as performed in this paper are not always good measures of the *qualitative* difference between sources. Nevertheless qualitative ideas about differences between sources are not always relevant to the practical problem of bringing steganalysis into the ”real-world”. Indeed it is the quantitative assessment of the CSM for the used detector that will enable us to understand the practical sources of CSM and to provide indication on how to mitigate it, hence our experimental setup choice.

To ease the reading of the results in this paper, the intrinsic difficulty of the dataset will be displayed in green while the inconsistency, displayed in red, will be directly given as the

Training \Testing	Source A	Source B
Source A	0.2	0.38
Source B	0.33	0.35

Table 1: Illustrative example for defining *qualitatively* defined metric that assesses both "intrinsic difficulty" (difference in column, when using the same dataset for testing) and "inconsistency" (difference between diagonal element that evaluate the detection accuracy without CSM), here the prediction error is measured by P_E.

difference between the intrinsic difficulty and the mismatched score in a given column.

We will see in sections 5–7 that many steps of the processing pipeline can impact, dramatically, both inconsistency and difficulty. Additionally, an important CSM between sources A and B sometimes finds its origins in an important inconsistency between those sources while it may sometimes be due to a higher difficulty of source A or B.

4. Common Core of all Experiments

The present paper focuses on steganalysis (and steganography) into the real world with highly heterogeneous datasets. Therefore, our work is restricted to images compressed using the JPEG standard, as it by far the most widely used image format for decades.

Two different image datasets have been used: the BOSS v1.0 dataset [2], made of $10,000$ images from 7 camera models, as well as the newly released ALASKA dataset[7] made of nearly $50,000$ images from 24 different cameras. In order to master each and every step of the image development, we used exclusively raw images. Except when explicitly mentioned otherwise, see Section 6.1, the conversion from raw to readable image files has been made using RawTherapee v5.4 because this software allows applying a wide range of operations like high-end commercial softwares. Images are compressed in JPEG using the Pillow library for python language. In addition, it is worth noting that only grayscale images have been used i.e. only one single color channel that corresponds to the luminance "Y" channel is encoded. To lower the computational cost of all our experimentations, the images are downsized to 512×512 pixels using a central crop. The downsizing of images through cropping, rather that by downsampling, is justified by the observation reported in [15] that downsampling by an important factor different sources of images tends to greatly homogenize them, hence removing in large part the impact of the CSM.

Regarding steganography and steganalysis we have used three different embedding schemes. We have chosen as a first steganographic method the software JPHide because, despite being very old, it remains quite popular. We also used a more recent embedding software, namely nsF5 [14] (no shrinkage F5) which is not adaptive, yet uses the optimal coding scheme (embedding on Shannon Bounds) for improving efficiency. The choice of nsF5 is due to the fact that, though it is quite outdated as compared to the state-of-the-art, it can be easily implemented using Syndrome Trellis Code (STC) [9]. Eventually, as a state-of-the-art embedding method, we also used J-UNIWARD [18] because it remains the most widely used algorithm in academic research and is ranked as one of the most secure JPEG steganography schemes. Those three embedding schemes provide us with a large range of steganographic methods from the oldest to the current state-of-the-art.

It is very important to note that the script for embedding hidden data has been modified to insert a payload measured in (inserted) bits per AC coefficients, referred to as bpAC. The main reason behind this choice is that when one wants to compare the impact of a parameter (ISO sensitivity or image processing tool for instance) the payload size has to remain constant. For JPEG domain steganography, the capacity is usually measured in bits per non-zero AC coefficients; however because this number of non-zero AC coefficient may greatly change depending on the value of a studied parameter, it may be very difficult to distinguish the intrinsic impact of the change in parameters value from the impact due to the change in number of nonzero AC coefficients. Another reason why we used the bpAC for the present paper, which focuses on a rather practical application, is that in the real life a user may be willing to embed a given message, hence a fixed number of bits; to mimic this situation, it is important to embed a fixed payload that does not depend on individual image properties.

Regarding the steganalysis counterpart, we have used two features sets, namely DCTR (Discrete Cosine Transform Residual) [17] and cc-JRM (Cartesian-Calibrated JPEG Rich Model) [24]. We used those two features sets since those are among the most efficient and because they are complementary: cc-JRM features are extracted directly from Discrete Cosine Transform (DCT) coefficients while DCTR uses the image decompressed into the spatial domain and applies 64 filters representing each DCT mode. Those two features sets are representative examples of the two approaches for JPEG image steganalysis and correspond to a good trade-off between complexity and efficiency.

Eventually the classifier we have used for results reported in this paper is the low-complexity linear classifier from [8]. We have also verified, in most cases, that similar results can be obtained with the well-known ensemble classifiers [26, 6]. However, due to the very high number of experimental cases that have been tested, we have chosen to benefit from the fast linear classifier which allows speeding up the classification time by one order of magnitude, see details in [8].

The steganalysis performance is measured using the usual empirical total probability of error under equal priors, denoted P_E, defined as:

$$P_E = \frac{P_{FA} + P_{MD}}{2}, \tag{1}$$

where P_{FA} and P_{MD} stands for the probability of false-alarm and missed-detection respectively.

It is also worth noting that in the present paper, the classifiers are always trained on pairs of cover and stego images and tested on a disjoint set of pairs of images.

5. Assessment of Sensor Settings on CSM

It is proposed to follow the digital imaging pipeline, as represented in Figure 3, to study the different steps of the processing pipeline, from raw image acquisition to JPEG compression, that impact the difficulty and the inconsistency in steganalysis. As shown in Figure 3, the digital imaging pipeline may be roughly divided into the three following main steps: (1) image acquisition (2) image rendering and processing, and (3) image compression, usually using JPEG standard. This section thus starts focusing on the parameters from image acquisition step and studies how much they may change the statistical properties of images, giving birth to a substantial CSM.

In addition, we have also carried out a study to assess the relative impact of the semantic content. Indeed, one may think that even for a fixed acquisition setup (camera, acquisition parameter and processing pipeline) the work is quite different for the steganalysis if the image has very smooth or a highly textured content. This study is presented in Section 8 and shows that such the semantic content has a much smaller effect on CSM than the processing pipeline.

Though a photographer may tune numerous parameters during image acquisition (aperture, ISO, focal length, exposure time, etc ...) we observed that only two parameters among those may have an important effect on steganalysis difficulty and inconsistency. Those parameters are the Exposure index rating, often referred to as the "ISO" or "ISO sensitivity", and the "quality" of the sensor, or camera model.

5.1. Sensor Model

To start with the most obvious one, we have selected a small set of camera models ranging from high-end cameras with full-frame sensors, to low quality sensors used in smartphones and tablets. The main characteristics of this subset of cameras from the ALASKA dataset are presented in Table 2.

Figure 4 shows the results obtained in terms of steganalysis difficulty, measured with the minimal error probability P_E, Eq. (1), over images from the different camera models. In order to remain as independent as possible from other parameters, we used only raw images with the lowest possible ISO processed using the same development pipeline, see Section 9 for more details.

Figure 4 presents the results obtained with cc-JRM features set for nsF5 with payload 0.04 bpAC and shows a striking correlation between the photo-site area (which is for a given production year a good approximation for the sensor quality), see Table 2 and the intrinsic difficulty, on the diagonal element. For a better readability, the results in those figures are sorted according to the camera sensor size. The three APS-C sensors, with size slightly below 24 × 16 mm have similar P_E and the cameras with smallest sensor size, typically *Samsung Galaxy S8* phone and *Apple Ipad* have a much larger P_E.

This can be explained by the fact that the smaller the pixel area, the smaller number of photons a photo-site counts for a given exposure time; this requires a larger amplification gain, decreasing the Signal-to-Noise Ratio.

The Nikon D610 camera is a notable exception; this can be explained by the fact that this camera is slightly older, since it is (certainly) equipped with the same sensor as its predecessor, the model Nikon D600 released in 2012. Similarly, the oldest camera Panasonic FZ28 (released in 2008) exhibits a P_E similar to that of smartphones and tablet devices while equipped with larger pixels.

One can also note that the different camera models may give birth to a substantial amount of inconsistency; in the most extreme case, when training over images from smartphones or tablets and testing on images from cameras with larger pixel area, the classifier almost acts like a random guess with P_E close to 50%.

Other results, not shown due to space limitations, with different embedding schemes, feature sets and JPEG quality factors, show the same tendencies on the role of the sensor on inconsistency. More precisely DCTR feature set seems quite more robust with respect to inconsistency, as compared to cc-JRM (see also Section 9) while the embedding scheme does not seem to have much impact.

5.2. Exposure Index Rating: ISO Sensitivity

To study the impact of the second acquisition parameter that has an important impact on both "intrinsic" difficulty and inconsistency, namely the ISO sensitivity, it is first proposed to use a similar approach as the one used for sensors. We gathered images, regardless the camera model and the scene content, from ALASKA and BOSS datasets with the same ISO sensitivity. The results of steganalysis on such subsets of constant ISO are presented in Figure 5 using cc-JRM features and nsF5 embedding scheme at payload 0.04 bpAC.

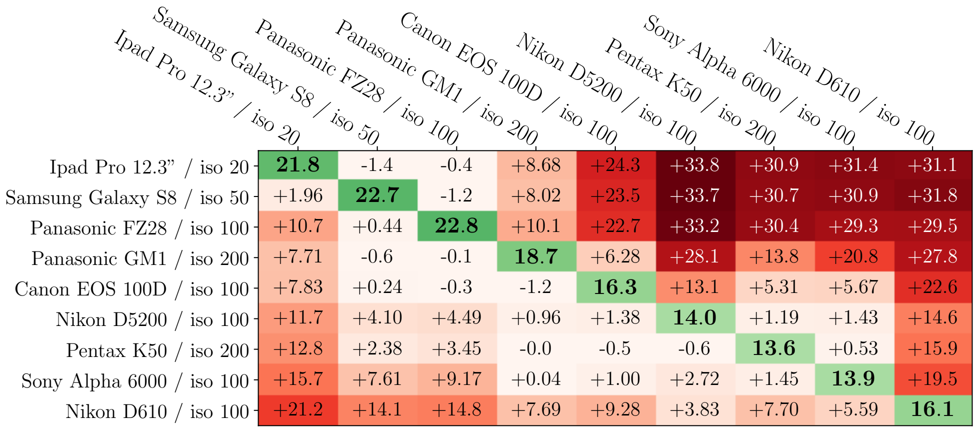
Unsurprisingly, one can note that the P_E increases with the ISO sensitivity, with a possibly dramatic impact of the ISO on the intrinsic difficulty, from a P_E of about 19% at ISO 100 and 200 to about 37% for a high ISO of 6400. Those results can easily be explained by the fact that, for a given camera model, the ISO is directly related to the amplification of signals from the light sensor and, hence, to the sensor noise magnitude.

One can also observe a quite important inconsistency since the change of a single ISO step may cause an increase in the P_E of more than 10%. It is also worth observing that this phenomenon holds true for all ISO sensitivities.

Eventually, it is interesting to look at the first row and column in

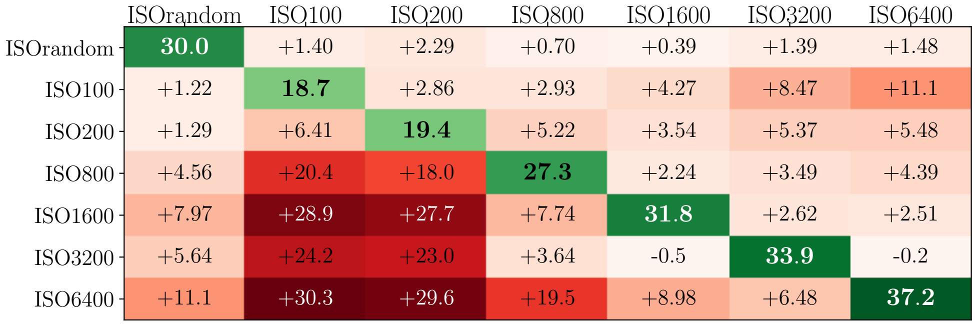
Camera name	Year	Sensor Size, mm	# of Pixels	photo-site area
Apple Ipad Pro	2015	4.80×3.60	12 Mpix	1.42 μm^2
Samsung GalaxyS8	2017	5.61×4.21	12 Mpix	1.96 μm^2
Panasonic FZ28	2008	6.08×4.56	10.1 Mpix	2.76 μm^2
Panasonic GM1	2013	17.30×13.0	16 Mpix	14.05 μm^2
Canon EOS100D	2013	22.3×14.9	18 Mpix	18.46 μm^2
Nikon D5200	2013	23.5×15.6	24.1 Mpix	15.21 μm^2
Pentax-K50	2013	23.7×15.7	16.3 Mpix	22.83 μm^2
Sony Alpha 6000	2014	23.7×15.7	24.3 Mpix	15.08 μm^2
Nikon D610	2013	35.9×24	24.3 Mpix	35.46 μm^2

Table 2: Main characteristics of the camera models used to assess the impact of sensors in Figures 4.



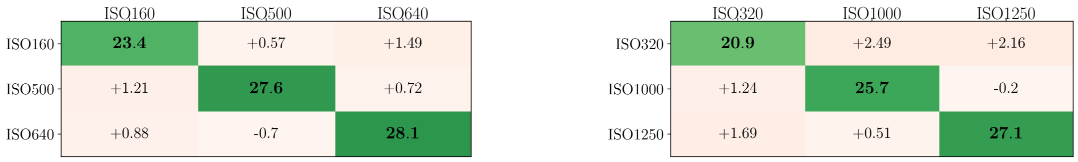
	Ipad Pro 12.3” / iso 20	Samsung Galaxy S8 / iso 50	Panasonic FZ28 / iso 100	Panasonic GM1 / iso 200	Canon EOS 100D / iso 100	Nikon D5200 / iso 100	Pentax K50 / iso 200	Sony Alpha 6000 / iso 100	Nikon D610 / iso 100
Ipad Pro 12.3” / iso 20	**21.8**	-1.4	-0.4	+8.68	+24.3	+33.8	+30.9	+31.4	+31.1
Samsung Galaxy S8 / iso 50	+1.96	**22.7**	-1.2	+8.02	+23.5	+33.7	+30.7	+30.9	+31.8
Panasonic FZ28 / iso 100	+10.7	+0.44	**22.8**	+10.1	+22.7	+33.2	+30.4	+29.3	+29.5
Panasonic GM1 / iso 200	+7.71	-0.6	-0.1	**18.7**	+6.28	+28.1	+13.8	+20.8	+27.8
Canon EOS 100D / iso 100	+7.83	+0.24	-0.3	-1.2	**16.3**	+13.1	+5.31	+5.67	+22.6
Nikon D5200 / iso 100	+11.7	+4.10	+4.49	+0.96	+1.38	**14.0**	+1.19	+1.43	+14.6
Pentax K50 / iso 200	+12.8	+2.38	+3.45	-0.0	-0.5	-0.6	**13.6**	+0.53	+15.9
Sony Alpha 6000 / iso 100	+15.7	+7.61	+9.17	+0.04	+1.00	+2.72	+1.45	**13.9**	+19.5
Nikon D610 / iso 100	+21.2	+14.1	+14.8	+7.69	+9.28	+3.83	+7.70	+5.59	**16.1**

Figure 4: Steganalysis error rate, P_E, for different camera models. Results obtained using cc-JRM features and nsF5 embedding scheme at 0.04 bpAC.



	ISOrandom	ISO100	ISO200	ISO800	ISO1600	ISO3200	ISO6400
ISOrandom	**30.0**	+1.40	+2.29	+0.70	+0.39	+1.39	+1.48
ISO100	+1.22	**18.7**	+2.86	+2.93	+4.27	+8.47	+11.1
ISO200	+1.29	+6.41	**19.4**	+5.22	+3.54	+5.37	+5.48
ISO800	+4.56	+20.4	+18.0	**27.3**	+2.24	+3.49	+4.39
ISO1600	+7.97	+28.9	+27.7	+7.74	**31.8**	+2.62	+2.51
ISO3200	+5.64	+24.2	+23.0	+3.64	-0.5	**33.9**	-0.2
ISO6400	+11.1	+30.3	+29.6	+19.5	+8.98	+6.48	**37.2**

Figure 5: Steganalysis error rate, P_E, as a function of the ISO sensitivity of inspect images. Images from both BOSS and ALASKA base have been used to have as many images as possible for the widest possible range of ISO sensitivities. Results obtained using jpeg quality factor 100, cc-JRM features set and nsF5 at payload 0.04 bpAC.



	ISO160	ISO500	ISO640
ISO160	**23.4**	+0.57	+1.49
ISO500	+1.21	**27.6**	+0.72
ISO640	+0.88	-0.7	**28.1**

(a) Leica M9 images set #1, all with the same content.

	ISO320	ISO1000	ISO1250
ISO320	**20.9**	+2.49	+2.16
ISO1000	+1.24	**25.7**	-0.2
ISO1250	+1.69	+0.51	**27.1**

(b) Leica M9 images set #2, all with the same content.

Figure 6: Steganalysis error rate, P_E, as a function of the ISO sensitivity for two image datasets from the same model and the same scene shoot with different sensitivity and exposure time. Results obtained using jpeg quality factor 100, cc-JRM features set and nsF5 at payload 0.04 bpAC.

Figure 5 which represents the results obtained when using raw images with all possible ranges of ISO sensitivities. Indeed, the very first row shows that the “merging” strategy, or more precisely the holistic approach, which consists in creating a dataset with all possible ISO sensitivities seems quite successful in the present case; however this should be balanced by the fact that when studying only a single parameter, here the ISO, one faces a very limited range of possible parameters, see more results in Section 9 on the strategies to face CSM.

However the pitfall of such results is that ISO sensitivity rating actually does not compare from a camera model to another. In fact the underlying amplification factor used, for a given ISO sensitivity, depends on camera models; typically it is likely for a smartphone camera the amplification factor of signals is actually larger than for a full-frame camera. This ISO is typically set such that, for a given images scene, the same aperture and shutter speed is well exposed with the same ISO regardless of the camera model, see the ISO Standard [20]. Furthermore, the ISO sensitivity is also very often related to the image content, outdoor images with sunlight have usually low ISO, indoor images have a more important ISO while night images may reach the maximal ISO.

Therefore, it is proposed to conduct a study using a single camera model, namely the *Leica M9*, and images of the same scenes with various ISO sensitivity ratings. This is much more time consuming but allows isolating the unique impact of ISO sensitivity rating. Those results are reported in Figure 6 under the same setting of nsF5 embedding scheme and DCTR feature set. On those figures, one can observe the same evolution of the P_E when ISO increases; one thus observes a similar impact of the

ISO on the intrinsic difficulty as compared with Figure 5. The inconsistency is, however, much lower for this second experimentation; this may be explained by the facts that (1) all images comes from the very same sensor and (2) we used a much smaller image set due to acquisition time.

6. Assessment of Processing Pipeline Impact

6.1. Assessment of Image Processing Software Impact

In this section it is proposed to study the impact of the image processing pipeline on the CSM. It is proposed to start with a "coarse grain" assessment of the development software. To this end, we have used three different software, *Adobe Lightroom*, version 6.12 (2015), since it is the mostly widely used commercial software ; its main competitor *DxO PhotoLab*, OpticsPro version 11.4.5, which originally aims at correcting aberrations, has also been used. Eventually, we used rawtherapee, version 5.4, as a representative open-source free software.

The extent of the CSM observed by applying these three software, with their default parameters, has been shown in figure 1. Those figures show that the choice of the software does indeed have an important impact on both the intrinsic difficulty and the inconsistency.

However, the study of raw images processing software offers only a "coarse grain" analysis since, for most of them, the user does not know exactly the set of algorithms that are used to convert each and every image as well as the parameters of those algorithms. Therefore, it is likely that using two different software results in the application of a numerous of differences in the processing operations which cannot be distinguished.

6.2. Assessment of Individual Image Processing Tool

Therefore, in this section, it is proposed to use one single software and to change the algorithms used along with their parameters. We have used the software rawtherapee (version 5.4) because it is an open-source software that can be used in command-line for batch processing, it is very well documented and it allows an extremely broad tuning[2].

Note that, because there is potentially a very wide range of algorithms that can be used to enhance the visual quality of images, we focused on the techniques that are (1) the most commonly used and (2) that have shown to influence the most both the intrinsic difficulty and the source inconsistency. To be clear, the latter point means that we have first tried a wide range of image enhancement tools (from color adjustment, overexpose and underexpose area compensation, color balance, gamma correction, etc…). We have found that the main influential common algorithms are (1) demosaicing, (2) denoising, (3) sharpening and (4) resizing.

RawTherapee was run using the command-line interface, in order to adjust the different parameters automatically, and 22 processing pipelines are compared, see Figure 7-15:

- Four with different demosaicing algorithms [12], two with high accuracy at a price of higher computational complexity, namely AMaZE and DCB2, one optimized for low illumination and high noise, IGV, and "fast" which is a linear demosaicing

- The denoising operation, for which we used the Directional Pyramid Denoising based on wavelet decomposition [32], with four levels of denoising strengths, from 30 to 90 out of 100.

- Two image sharpening, or edge enhancement, algorithms have been used. The first being a modified version of the well-known unsharp masking [35], also referred to as USM in this paper, and the second being based on the celebrated Richardson-Lucy blind deconvolution [37, 10]. We used three different set of parameters for USM and two different set of parameters for RL deconvolution (sorted by ascending "strength").

- Image resizing (upscaling and downscaling) using the commonly found Lanczos filter. Two rescaling factors have been used of 60% and 130% respectively.

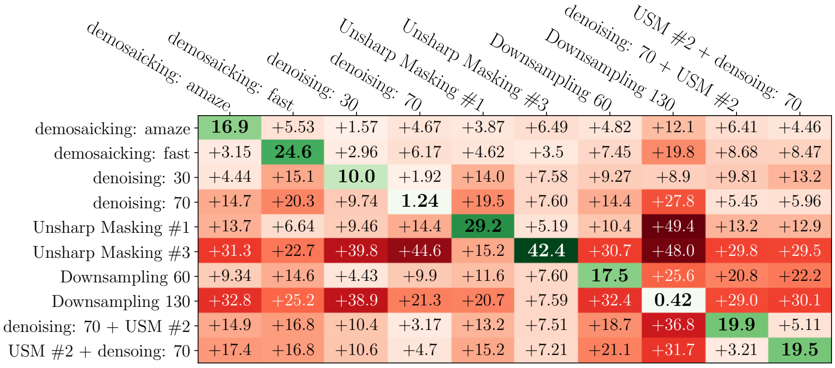
- We also defined a set of 7 "combined" processing pipelines, which all consist in application of both denoising and edge enhancement. Three are made with denoising first (with the same strength) and then using different parameters for USM. Four processing pipelines are made by sharpening first (with the same set of parameters) and then denoising.

For readability's sake, the results presented in Figures 7-10 only show a subset of those pipelines, the rest of the results will be made available online upon acceptance of the paper.

The immediate and most obvious conclusion from those results is the high impact of the processing pipeline on both the intrinsic difficulty and inconsistency. Indeed, in the case of Figure 7 that presents results obtained using J-UNIWARD with payload 0.3 and JPEG QF 100, the intrinsic difficulty ranges from 0.4% (when using upsampling or the strongest denoising) to 42.4% (when using the strongest sharpening). The inconsistency, on the other hand, is stronger among pipelines that have opposite effects, for example, the deviation from the diagonal can be as high as 19% between USM #1 and denoising 70. Unsurprisingly, the inconsistency decreases for processing close to the one used for training; for instance, inconsistency between different parameters of denoising is rather limited. This impact holds irrespectively of the quality factor, embedding scheme or steganalysis features.

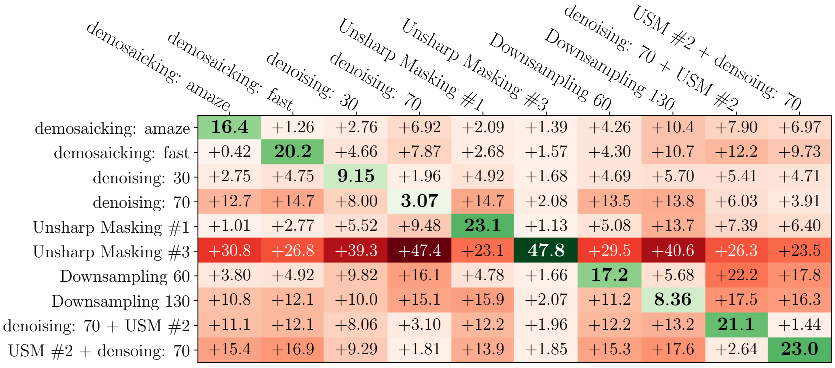
Secondly, one can note that lowering the quality factor has the impact of lowering the inconsistency between different processing. To observe this, one can compare columns of Figure 7 and Figure 8 which only differ by the JPEG QF: the P_E is much more uniform on each column for QF75 than for QF100. This can be attributed to the denoising effect of JPEG compression which is applied equally on each database, thus lowering the impact of preceding development steps.

[2] RawTherapee is available at: `https://rawtherapee.com/` and a comprehensive documentation can be found at: `https://rawpedia.rawtherapee.com`.



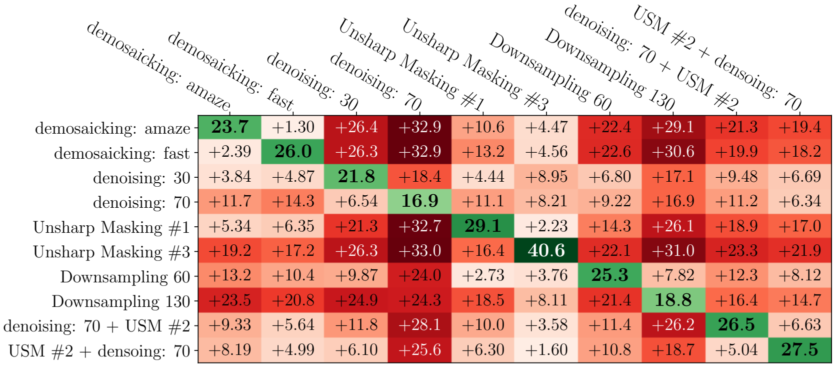
	demosaicking: amaze	demosaicking: fast	denoising: 30	denoising: 70	Unsharp Masking #1	Unsharp Masking #3	Downsampling 60	Downsampling 130	denoising: 70 + USM #2	USM #2 + densoing: 70
demosaicking: amaze	**16.9**	+5.53	+1.57	+4.67	+3.87	+6.49	+4.82	+12.1	+6.41	+4.46
demosaicking: fast	+3.15	**24.6**	+2.96	+6.17	+4.62	+3.5	+7.45	+19.8	+8.68	+8.47
denoising: 30	+4.44	+15.1	**10.0**	+1.92	+14.0	+7.58	+9.27	+8.9	+9.81	+13.2
denoising: 70	+14.7	+20.3	+9.74	**1.24**	+19.5	+7.60	+14.4	+27.8	+5.45	+5.96
Unsharp Masking #1	+13.7	+6.64	+9.46	+14.4	**29.2**	+5.19	+10.4	+49.4	+13.2	+12.9
Unsharp Masking #3	+31.3	+22.7	+39.8	+44.6	+15.2	**42.4**	+30.7	+48.0	+29.8	+29.5
Downsampling 60	+9.34	+14.6	+4.43	+9.9	+11.6	+7.60	**17.5**	+25.6	+20.8	+22.2
Downsampling 130	+32.8	+25.2	+38.9	+21.3	+20.7	+7.59	+32.4	**0.42**	+29.0	+30.1
denoising: 70 + USM #2	+14.9	+16.8	+10.4	+3.17	+13.2	+7.51	+18.7	+36.8	**19.9**	+5.11
USM #2 + densoing: 70	+17.4	+16.8	+10.6	+4.7	+15.2	+7.21	+21.1	+31.7	+3.21	**19.5**

Figure 7: Steganalysis error rate, P_E, and mismatched between all the 22 tested development pipeline. Results obtained using jpeg quality factor 100, DCTR features set and J-UNIWARD at payload 0.3 bpAC with images from both BOSSbase.



	demosaicking: amaze	demosaicking: fast	denoising: 30	denoising: 70	Unsharp Masking #1	Unsharp Masking #3	Downsampling 60	Downsampling 130	denoising: 70 + USM #2	USM #2 + densoing: 70
demosaicking: amaze	**16.4**	+1.26	+2.76	+6.92	+2.09	+1.39	+4.26	+10.4	+7.90	+6.97
demosaicking: fast	+0.42	**20.2**	+4.66	+7.87	+2.68	+1.57	+4.30	+10.7	+12.2	+9.73
denoising: 30	+2.75	+4.75	**9.15**	+1.96	+4.92	+1.68	+4.69	+5.70	+5.41	+4.71
denoising: 70	+12.7	+14.7	+8.00	**3.07**	+14.7	+2.08	+13.5	+13.8	+6.03	+3.91
Unsharp Masking #1	+1.01	+2.77	+5.52	+9.48	**23.1**	+1.13	+5.08	+13.7	+7.39	+6.40
Unsharp Masking #3	+30.8	+26.8	+39.3	+47.4	+23.1	**47.8**	+29.5	+40.6	+26.3	+23.5
Downsampling 60	+3.80	+4.92	+9.82	+16.1	+4.78	+1.66	**17.2**	+5.68	+22.2	+17.8
Downsampling 130	+10.8	+12.1	+10.0	+15.1	+15.9	+2.07	+11.2	**8.36**	+17.5	+16.3
denoising: 70 + USM #2	+11.1	+12.1	+8.06	+3.10	+12.2	+1.96	+12.2	+13.2	**21.1**	+1.44
USM #2 + densoing: 70	+15.4	+16.9	+9.29	+1.81	+13.9	+1.85	+15.3	+17.6	+2.64	**23.0**

Figure 8: Steganalysis error rate, P_E, under the same settings as Figure 7, except for the jpeg quality factor set to 75 J-UNIWARD embedded payload adjusted consequently to 0.03 bpAC.



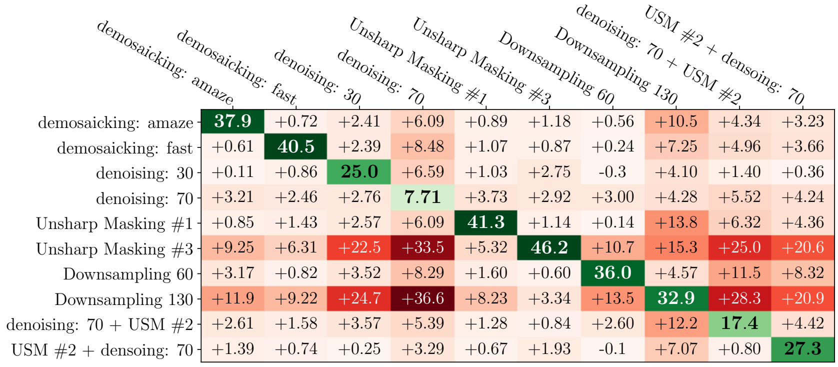
	demosaicking: amaze	demosaicking: fast	denoising: 30	denoising: 70	Unsharp Masking #1	Unsharp Masking #3	Downsampling 60	Downsampling 130	denoising: 70 + USM #2	USM #2 + densoing: 70
demosaicking: amaze	**23.7**	+1.30	+26.4	+32.9	+10.6	+4.47	+22.4	+29.1	+21.3	+19.4
demosaicking: fast	+2.39	**26.0**	+26.3	+32.9	+13.2	+4.56	+22.6	+30.6	+19.9	+18.2
denoising: 30	+3.84	+4.87	**21.8**	+18.4	+4.44	+8.95	+6.80	+17.1	+9.48	+6.69
denoising: 70	+11.7	+14.3	+6.54	**16.9**	+11.1	+8.21	+9.22	+16.9	+11.2	+6.34
Unsharp Masking #1	+5.34	+6.35	+21.3	+32.7	**29.1**	+2.23	+14.3	+26.1	+18.9	+17.0
Unsharp Masking #3	+19.2	+17.2	+26.3	+33.0	+16.4	**40.6**	+22.1	+31.0	+23.3	+21.9
Downsampling 60	+13.2	+10.4	+9.87	+24.0	+2.73	+3.76	**25.3**	+7.82	+12.3	+8.12
Downsampling 130	+23.5	+20.8	+24.9	+24.3	+18.5	+8.11	+21.4	**18.8**	+16.4	+14.7
denoising: 70 + USM #2	+9.33	+5.64	+11.8	+28.1	+10.0	+3.58	+11.4	+26.2	**26.5**	+6.63
USM #2 + densoing: 70	+8.19	+4.99	+6.10	+25.6	+6.30	+1.60	+10.8	+18.7	+5.04	**27.5**

Figure 9: Steganalysis error rate, P_E, under the same settings as Figure 7, but using nsF5 embedding scheme with a payload adjusted to 0.04 bpAC and the cc-JRM feature set.

Thirdly, the choice of steganalysis feature does also have an impact on the properties of the inconsistency. If one compares Figure 7 and Figure 9, one can conclude that cc-JRM is far more sensitive to the processing pipelines, as the inconsistency between processing is larger than for DCTR. This result will actually be confirmed in the following sections where we show that the cc-JRM is an excellent feature set to detect the processing pipeline while DCTR is far less accurate. We emphasize that this effect hold true for all embedding schemes (nsF5, J-UNIWARD and JPHide) and all JPEG quality factors.

Finally, we will note that the adaptivity of an embedding scheme does not seem to play a clear role with regard to inconsistency. When testing on nsF5, a weakly adaptive scheme, which embeds only on non-zero AC coefficient, the inconsistency is on average higher than for J-UNIWARD which is a highly adaptive embedding scheme. However, the inconsistency for Jpeg Hide&Seek presented in Figure 10, a non-adaptive scheme, is on average the smallest of the three studied embedding schemes.

On the other, the impact of adaptivity is very clear on the intrinsic difficulty: contrasting Figure 7 and 9, one can note that J-UNIWARD becomes almost undetectable when medium



	demosaicking: amaze	demosaicking: fast	denoising: 30	denoising: 70	Unsharp Masking #1	Unsharp Masking #3	Downsampling 60	Downsampling 130	denoising: 70 + USM #2	USM #2 + densoing: 70
demosaicking: amaze	**37.9**	+0.72	+2.41	+6.09	+0.89	+1.18	+0.56	+10.5	+4.34	+3.23
demosaicking: fast	+0.61	**40.5**	+2.39	+8.48	+1.07	+0.87	+0.24	+7.25	+4.96	+3.66
denoising: 30	+0.11	+0.86	**25.0**	+6.59	+1.03	+2.75	-0.3	+4.10	+1.40	+0.36
denoising: 70	+3.21	+2.46	+2.76	**7.71**	+3.73	+2.92	+3.00	+4.28	+5.52	+4.24
Unsharp Masking #1	+0.85	+1.43	+2.57	+6.09	**41.3**	+1.14	+0.14	+13.8	+6.32	+4.36
Unsharp Masking #3	+9.25	+6.31	+22.5	+33.5	+5.32	**46.2**	+10.7	+15.3	+25.0	+20.6
Downsampling 60	+3.17	+0.82	+3.52	+8.29	+1.60	+0.60	**36.0**	+4.57	+11.5	+8.32
Downsampling 130	+11.9	+9.22	+24.7	+36.6	+8.23	+3.34	+13.5	**32.9**	+28.3	+20.9
denoising: 70 + USM #2	+2.61	+1.58	+3.57	+5.39	+1.28	+0.84	+2.60	+12.2	**17.4**	+4.42
USM #2 + densoing: 70	+1.39	+0.74	+0.25	+3.29	+0.67	+1.93	-0.1	+7.07	+0.80	**27.3**

Figure 10: Steganalysis error rate for QF=100, DCTR feature set and JPHide at a payload of 0.02 bpAC

sharpening is applied on images but also extremely detectable when sufficient denoising is applied, while the change in intrinsic difficulty for nsF5 is nowhere near as dramatic.

7. Impact of the JPEG QF

To study the impact of the quality factor on CSM we compressed BOSSBase with 10 different quality factors : 75 to 79 and 96 to 100. The demosaicing used for every database was AmAZe and images centrally cropped to 512x512. No other processing was otherwise applied. Images were embedded with J-UNIWARD with payload 0.03 bpac for quality factors from 75 to 79 and with payload 0.3 bpac for quality factors 96 to 100. Figures 11 illustrate the obtained results.

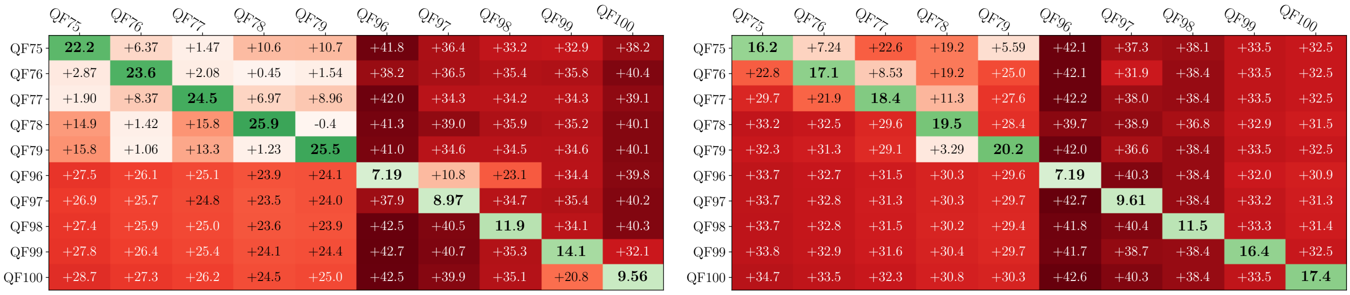
Irrespectively of the steganalysis employed, one can observe that the inconsistency between quality factors is stronger for higher quality factors than for lower ones. This can be explained by the fact that the statistical properties of the quantized DCT coefficients will be impacted more for high quality than for low quality factors. For example, the quantization step of mode (7,7) fluctuates from 2 to 1 between QF99 and QF100 and between 50 and 48 between QF75 and QF76. In the first case the variance of the quantization noise is increased by 400%, whereas in the second case it is only increased by 8.5%. One can also note that the intrinsic difficulty of each image base decreases with decreasing quality factors. The speed of this decrease is also faster for higher quality factors for the same reason.

The properties of the CSM also depend on the chosen steganalysis methodology. Indeed, one will observe when comparing that the inconsistency between quality factors is always lower when using cc-JRM features than with DCTR features. This can be explained by how these features are built. The JRM features only build co-occurrence matrices of the DCT coefficients without using any knowledge of the quality factor. On the other hand, DCTR builds histograms of noise residuals and quantizes those using the knowledge of the quality factor, thus specializing the classifier trained on those features toward a specific quality factor. This relative robustness of the JRM features, however, comes at the price of a lesser performance in terms of detectability and thus a higher intrinsic difficulty.

8. Impact of Semantic Content on CSM

So far, we focused only on CSM factors that come from the acquisition and processing pipeline. However one might rightly ask about the impact of the content of the scene of the images themselves. For example, one might wonder about the impact of the CSM when training on a dataset composed only of smooth images (e.g blue skies or out of focus images) and testing on highly textured images. In this section, we thus study the impact of the semantic content on the cover-source mismatch and we compare its impact with the impact of the processing pipeline. To that end, we used two datasets reflecting the worst case scenario of the aforementioned example. The two datasets were taken exclusively with Camera Model Sony Alpha ILCE-7R at constant ISO800. The first one named FLAT is composed of 455 RAW images of blue skies, captures with no lens attached to the camera and harshly out-of-focus blurred scene. The second dataset, named TEXTURE, is composed of 402 RAW images of highly textured scenes (such as grass, concrete, stone details, forests). In addition, those datasets were subject to two different, yet simple, processing pipelines: the first, consists in IGV demosaicing followed by light denoising (corresponding to ”denoising 30” in Figure 15) and is refered to as “Denoised” while the second is refered to “Sharpen” and consists in Amaze demosaicing followed by a light sharpening (corresponding to ”Unsharp masking #1” in Figure 15). Finally, each image was exported to a 16-bit TIFF, and randomly cropped several times without overlap in order to produce four datasets of 10 000 images compressed with quality factor 100. The embedding was done using J-UNIWARD at 0.3 bpp and classification using the Low Complexity Linear Classifier (LCLC) with DCTR features.

From Figure 8, we clearly see that inconsistency is almost always the highest when mismatch is due to the pipeline instead of the content (the only exception being for the denoised TEXTURE case, in which case the inconsistency is alway higher than 15% for all types of mismatch). For example, training and testing on the FLAT dataset with different pipelines induces a higher inconsistency (close to +15% in terms of P_E when compared to the intrinsic difficulty of each dataset) than training on FLAT and testing on TEXTURE but keeping the same processing pipeline (+3% in the case of Amaze + sharpening and +8%



(a) Results obtained with cc-JRM features.

(b) Results obtained with DCTR features.

Figure 11: Steganalysis error rate for different QF with J-UNIWARD at payload of 0.03 bpAC for the five lowest QF and 0.3 bpAC for the five highest.

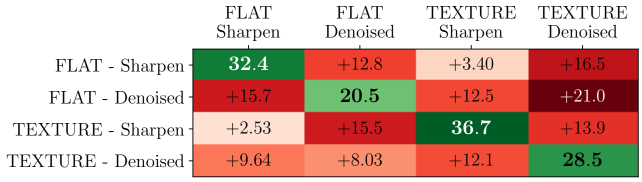


Figure 12: Assessment of CSM wrt dramatic change in the semantic content and comparison with a light modification of the processing pipeline.

in the case of IGV + denoising).

Those results thus show that, though semantic content does have an impact on CSM, the impact of the processing pipeline is also significant. Note that the visual impact of the processing algorithms used in this experiment is mild, showing that even a slight change in the processing pipelines is as important as a totally different semantic content. Note also that in practice, the steganalyst does have access to the ”test set” and can thus easily craft a training set with similar semantic contents. Finding solutions to cope with the processing pipeline is not trivial and is presented in the next section.

9. Practical Strategies for Mitigating the Processing Pipeline Mismatch in Steganalysis

In this paper three strategies for mitigating the CSM due to the Image Processing Pipeline (IPP) are investigated: the atomistic approach (which can be decomposed into two sub-strategies) and the holistic approach. Because of the very important inconsistencies due to differences between quantization tables, in this section the same quality factor is used for training and testing. The holistic approach applied in the current paper consists of a “mixed training”, i.e. feature sets of all investigated IPPs are blended together for constructing a single training set.

One naïve atomistic approach is to use the EXIF metadata to identify the processing pipeline: software such as Adobe Lightroom stores all the parameters used during the development from RAW to JPEG into the EXIF headers. However, we did not include this approach because EXIF metadata are very unreliable since one can easily modify them, using for instance *exiftool* program.

We proposed, instead, another blind atomistic approach which exploits the fact that correlation-based feature sets such as the ones used in steganalysis have proven to be very effective for performing various tasks in digital image forensics [4, 31, 36]. The approach thus exploits the power of steganalysis feature sets for designing an “IPP-informed” steganalytic detector, similar to the one proposed in [3]. The approach consists of two steps. In the first step a classifier is trained for distinguishing between the different considered IPPs. In the second step the classifier is applied for determining the IPP that best fits a given image under inspection and then a steganalytic detector trained on images produced with that IPP is applied. In [3] the IPP-classifier was applied for distinguishing a limited set of 11 IPPs and only one feature set (CC-JRM) was used. The used detector was the Ensemble Classifier.

In the current paper we investigate the performance of the IPP-classifier for distinguishing the 22 IPPs mentioned in section 6.2 and we compare results obtained using DCTR and CC-JRM feature sets and with both the EC and the linear classifier. The results of the IPP-classifier based using both CC-JRM and DCTR as well as jpeg qualify factors 75 and 100 are summarized in figure 13. The numbers in the figure represent the diagonal of the confusion matrix for the IPP-classifier. This figure shows, unsurprisingly, a much higher accuracy with highest QF and, interestingly, a much higher accuracy using CC-JRM feature set than DCTR.

To complement the results presented in figure 13, table 3 presented the overall accuracy for identification of processing pipeline over both over cover and stego images for cc-JRM and DCTR as well as for JPEG QF 100 and QF 75. The third and fourth column of the table show the accuracy obtained on cover images when using respectively the ensemble classifier and the linear classifier. The last three columns show the results obtained on the various stego-images. Please note that these results are obtained with the classifier trained on only cover images. The table shows that results obtained by the linear classifier are very similar to those obtained by the ensemble classifier. For cover images and stego images based on nsF5 and JPHide, cc-JRM performs better than DCTR. Interestingly, for these steganographic methods the accuracy obtained on stego-images is very close to that obtained on cover images; this is especially important to ensure it can be used for “atomistic”

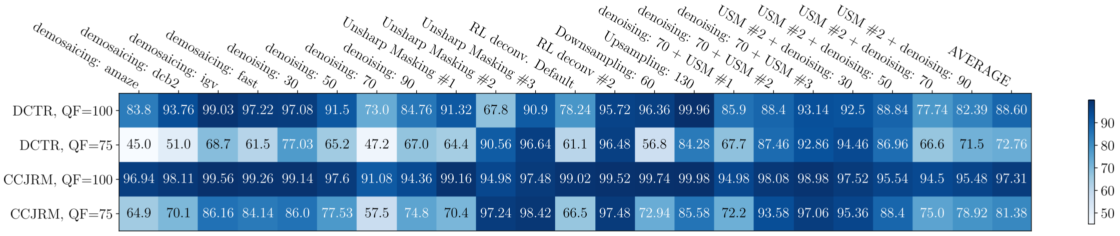


Figure 13: Classification accuracy (%) of Image Processing Pipelines for different feature sets, JPEG QF and development pipelines.

steganalysis.
Note that JPHide at QF 75 suffers an important issue, probably because of overhead information hiding, the detectability becomes obvious which results in lower accuracy for identification of processing pipeline.

Figures 14 and 15 show the results obtained by applying different steganalysis strategies:

- Targeted strategy: this is also referred to as the "fully-matched case" and corresponds to the situation where training and testing sets are extracted from the same source, i.e. the same development pipeline.
- IPP classification aided steganalysis: this corresponds to the atomistic approach discussed above.
- Holistic steganalysis: this correspond to gathering, during training, features from all IPP. The number of training samples equal, respectively to 5000, corresponding to the number of images used for training a single targeted detector, and 110000, which is the total of samples and matches the effective number of features used in atomostic approach.

The results shown in figures 14 and 15 represent a subset of the results obtained for various steganographic methods, quality factors (QF) and feature sets (DCTR and cc-JRM). This subset of results is chosen by the authors in order to highlight some conclusions. Upon acceptance of the paper, the full set of results will be made available on-line. In each figure the different steganalysis strategies are represented by different symbols and the columns represent the 22 different IPPs. The last column is the average of P_E over all development pipelines for a given steganalysis strategy so that one can easily contrast the efficiency of each approach. It was noted above that in general cc-JRM produces better results than DCTR for classifying the IPP. However, section 6 shows that DCTR provides better steganalysis results. Therefore, in the atomistic approach based on DCTR we also included an approach where the IPP-classification is based on cc-JRM while the actual steganalysis step is based on DCTR. The figures show that for the holistic approach it is very important to have a large training database. Under the experimental conditions examined in this paper, the atomistic approach and the holistic approach (with 110000 training samples) lead to equivalent results. However, in some cases the atomistic approach seems to give better results. This is especially the case when its first step is based on the cc-JRM features and the second on DCTR. Surprisingly, we obsetrved that this also holds true for JPHide at QF=75 where neither DCTR or cc-JRM provide good results for IPP classification in stego-images. This suggests that, even when the IPP-classifier is unable to identify the correct IPP, it does assign inspected images to a class with similar properties. This hypothesis should be verified in further works. The authors think that the atomistic approach is mainly promising in cases where the steganalyst is able to gain some knowledge about the IPP applied to images under inspection. Such knowledge could be obtained from the image (EXIF) metadata or from digital image forensics analysis. On the other hand, the holistic approach is likely to provide better generalization capabilities. The generalization capabilities of both strategies are a topic for further research.

Feature	QF	EC	Linear classifier			
set		Cover	Cover	J-UNI	nsF5	JPHide
DCTR	100	88.2	88.6	86.9	88.8	85.9
DCTR	75	73.7	72.8	72.8	75.1	40.3
CCJRM	100	97.6	97.3	94.9	98.1	95.5
CCJRM	75	82.4	81.4	74.8	82.1	55.4

Table 3: Comparison of image processing pipeline classification accuracy (Acc in %) according to feature set and JPEG quality factor.

10. Summary of the results and conclusions

The goal of this paper was to provide a deep analysis of the effects of the CSM and the parameters that induce it. By using different steganographic schemes and steganalysis feature sets, we have learned that several factors impact considerably the inconsistency between two sources, and that all feature sets are not robust to the same processes. Our extensive tests show that a steganalyst wanting to use "as it" steganalysis tools should be extremely cautious. Steganalysis accuracy may dramatically decrease whenever images under scrutiny and images from the training set are inconsistent w.r.t. either sensor, JPEG quality factor, or development operations (upsampling, sharpening, demosaicing).

We now summarize the conclusions for each of the studied parameters – camera sensor, ISO, processing pipeline, quality factors and semantic content. We refer the reader to the relevant sections for the discussions of the results which led to those conclusions. (The conclusions hold when the parameter in question is fixed while the other parameters remain unspecified, except for the quality factor which has always been fixed

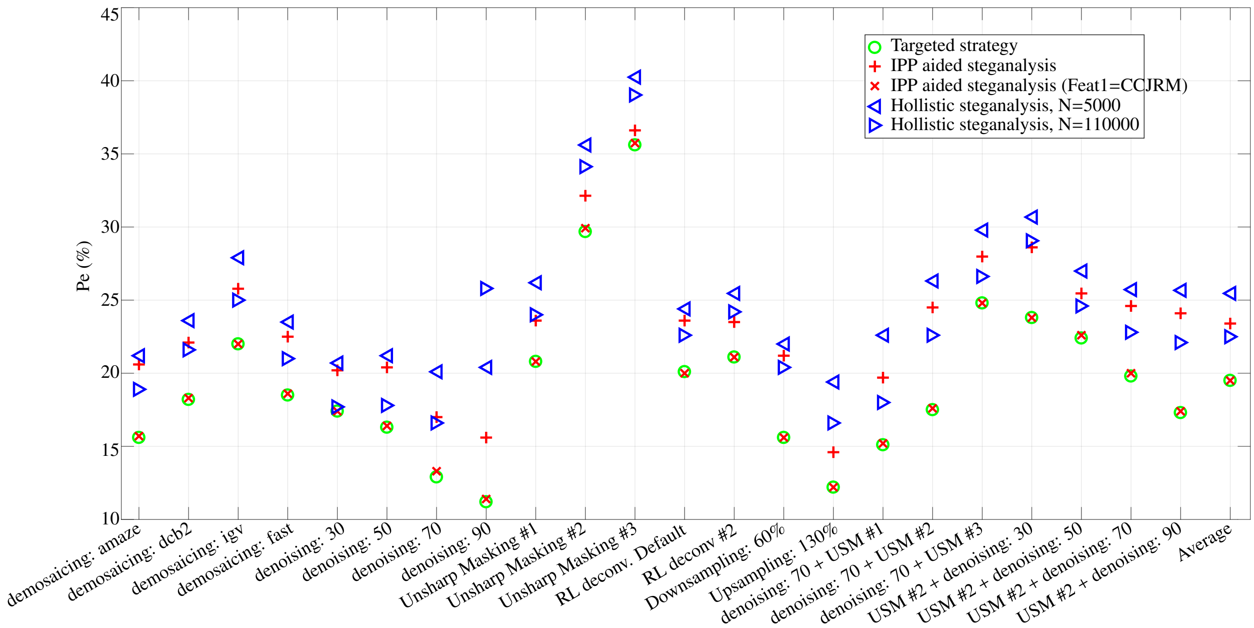


Figure 14: Comparison of detection accuracy, in terms minimal probability of error under equal prior P_E (%), for the three proposed strategies and different training sizes ; Embedding method is J-UNIWARD, JPEG QF is 75, DCTR features are used for steganalysis.

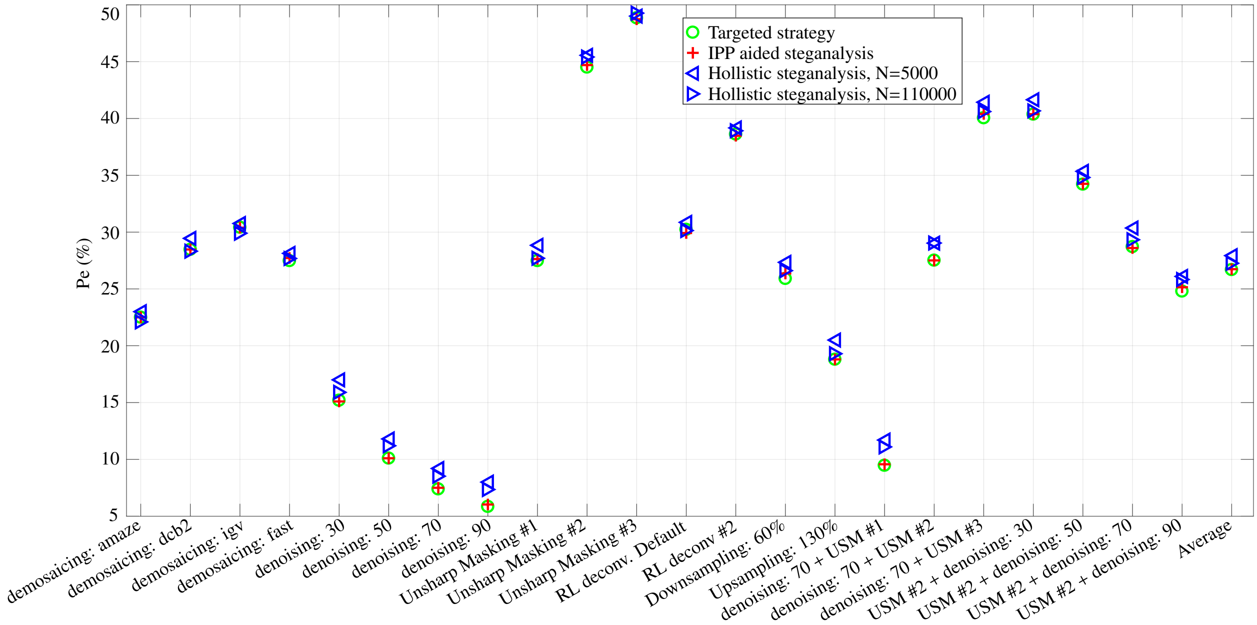


Figure 15: Comparison of detection accuracy, in terms minimal probability of error under equal prior P_E (%), for the three proposed strategies and different training sizes ; Embedding method is nsF5, JPEG QF is 100 and ccJRM features are used (both both IPP classification and steganalysis).

in our experiments.)

Impact of the Camera model:.

- The intrinsic difficulty of a dataset is correlated to the photo-site area of the sensor used. Together with the manufacturing year, this parameter assess the quality of the sensor at the photo-site level.

- The inconsistency between two datasets will decrease as the photo-site area of the cameras in the training and testing set get closer.

Impact of the ISO sensitivity.

- For a given sensor, the intrinsic difficulty of a dataset is correlated to the ISO value, higher ISO leading to higher intrinsic difficulty.

- There is no inconsistency between two datasets with different ISO if every other parameter is fixed. However, if the other parameters remain unspecified, inconsistency will increase as a function of the distance between the training and testing set ISO values.

Among the major findings of the present paper, the fundamental impact on the Cover-Source-Mismatch of the image processing pipeline is certainly the one that has been the least studied while it may give birth to the most important impact.

Main finding about the impact of Processing pipeline:.

- The processing pipeline has the highest impact on both the intrinsic difficulty and inconsistency – we observed a possible variation of the intrinsic difficulty from 0 to 40%.
- Processing pipelines that are closer in terms of function (e.g. : same denoising algorithm with different parameter values) will produce datasets with lower inconsistency than processing pipelines with different functions (e.g. denoising and sharpening).
- The order of the operation also matters; datasets which were processed with the same tools but in a different order are not necessarily coherent.
- The Final JPEG compression step may reduce significantly the inconsistency, especially when using a lower quality factor.
- Some feature sets are more sensitive to fluctuation of the processing pipeline, hence lead to more inconsistency overall when used to perform steganalysis (e.g., DCTR is less sensitive than cc-JRM to the processing pipeline).

Quality factor.

- Inconsistency is higher for high quality factors – typically, for quality factors higher than 90 – or more generally, quantization table with quantization steps close to 1.
- Inconsistency depends on the feature set used for steganalysis, hence a feature set like the JRM which does not integrate the quantization steps in its construction will be more robust to a difference in quality factors between the training and testing sets as compared to DCTR which makes explicit use of them.
- However, this robustness comes at the price of a higher overall intrinsic difficulty ; a feature set that is not tailored for a specific quality factor is more efficient when the quality factor is not known accurately and vice-versa.

Semantic content.

- While having an impact on both intrinsic difficulty and inconsistency in the worst case scenario we studied, semantic content has less impact than the processing pipeline on both those quantities.

To alleviate the CSM, we have investigated three different strategies: (1) targeted steganalysis which assumes that the training source is known, (2) a strategy inspired from forensic steganalysis which identifies first the processing pipeline and (3) an “holistic” strategy which aims at generating a diverse training set. Our experiments conclude that for a same training set size (1) yields to optimal results while (2) offers better performances than (3). By increasing the training set size however strategy (3) tends to offer better generalization properties.

These experimental results give some directions as to how future experiments studying CSM should be designed :

- Studying the camera as the only source of CSM is far from sufficient. If only the acquisition paramaters are of interest, then a source should be considered as the combination of the **camera, the ISO sensitivity, and the processing pipeline**.
- The processing pipeline has the greatest impact compared to every other parameter studied in this paper. As such it should be the first parameter to be taken into account when designing datasets in a CSM setting. In particular, vanilla BOSSBase is ill-suited to such a study as each camera follows different white balance parameters and different resizing factor, together with a processing pipeline not reflecting the variety of pipelines proposed in real-life forensics scenarios.

We hope that this work will motivate the community to investigate more the problem of *robustness* in steganalysis in order to find other solutions to cope with the huge diversity of digital image, since it is a requirement for practical use.

Acknowledgements

All the source codes and images datasets used to obtain the results presented will be made available on the Internet upon acceptance of the paper. The (raw) images dataset as well as the codes used for conversion to JPEG and application of several processing pipeline are already available at: `https://alaska.utt.fr`.
This work has been funded in part by the French National Research Agency (ANR-18-ASTR-0009), ALASKA project, by the French ANR DEFALS program (ANR-16-DEFA-0003), and by the Belgian Royal Higher Institute for Defense (project DAP18-01).

References

[1] M. Barni, G. Cancelli, A. Esposito, Forensics aided steganalysis of heterogeneous images, in: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1690–1693.

[2] P. Bas, T. Filler, T. Pevný, Break our steganographic system — the ins and outs of organizing boss, in: Information Hiding, 13th International Workshop, Lecture Notes in Computer Science, LNCS vol.6958, Springer-Verlag, New York, Prague, Czech Republic, 2011, pp. 59–70.

[3] D. Borghys, P. Bas, H. Bruyninckx, Facing the cover-source mismatch on jphide using training-set design, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec ’18, ACM, New York, NY, USA, 2018, pp. 17–22.

[4] M. Boroumand, J. Fridrich, Scalable processing history detector for jpeg images, in: ISandT (Ed.), Proc. IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics.

[5] G. Cancelli, G. Doerr, I. Cox, M. Barni, Detection of ±1 LSB steganography based on the amplitude of histogram local extrema, in: Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, pp. 1288 –1291.

[6] R. Cogranne, J. Fridrich, Modeling and extending the ensemble classifier for steganalysis of digital images using hypothesis testing theory, Information Forensics and Security, IEEE Transactions on (in press) (2015).

[7] R. Cogranne, Q. Giboulot, P. Bas, The alaska steganalysis challenge: A first step towards steganalysis, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'19, ACM, New York, NY, USA, 2019, pp. 125–137.

[8] R. Cogranne, V. Sedighi, J. Fridrich, T. Pevný, Is ensemble classifier needed for steganalysis in high-dimensional feature spaces?, in: Information Forensics and Security (WIFS), IEEE 7th International Workshop on, pp. 1–6.

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

[10] D.A. Fish, A.M. Brinicombe, E.R. Pike, J.G. Walker, Blind deconvolution by means of the richardson–lucy algorithm, J. Opt. Soc. Am. A 12 (1995) 58–65.

[11] A. Foi, M. Trimeche, V. Katkovnik, K. Egiazarian, Practical poissonian-gaussian noise modeling and fitting for single-image raw-data, Image Processing, IEEE Transactions on 17 (2008) 1737–1754.

[12] A. Forsey, S. Gungor, Demosaicing images from colour cameras for digital image correlation, Optics and Lasers in Engineering 86 (2016) 20 – 28.

[13] J. Fridrich, J. Kodovskỳ, Rich models for steganalysis of digital images, Information Forensics and Security, IEEE Transactions on 7 (2012) 868 –882.

[14] J. Fridrich, T. Pevný, J. Kodovský, Statistically undetectable jpeg steganography: Dead ends challenges, and opportunities, in: Proceedings of the 9th Workshop on Multimedia & Security, MM&Sec '07, ACM, New York, NY, USA, 2007, pp. 3–14.

[15] Q. Giboulot, R. Cogranne, P. Bas, Steganalysis into the wild: How to define a source?, in: Media Watermarking, Security, and Forensics, Proc. IS&T, pp. 318–1 – 318–12.

[16] M. Goljan, J. Fridrich, T. Holotyak, New blind steganalysis and its implications, in: in Proc. of the SPIE, Security, Steganography, and Watermarking of Multimedia Contents VIII, pp. 1–13.

[17] V. Holub, J. Fridrich, Low-complexity features for jpeg steganalysis using undecimated dct, Information Forensics and Security, IEEE Transactions on 10 (2015) 219–228.

[18] V. Holub, J. Fridrich, T. Denemark, Universal distortion function for steganography in an arbitrary domain, EURASIP Journal on Information Security 2014 (2014) 1–13.

[19] X. Hou, T. Zhang, G. Xiong, B. Wan, Forensics aided steganalysis of heterogeneous bitmap images with different compression history, KSII Transactions on Internet and Information Systems (TIIS) 6 (2012) 1926–1945.

[20] ISO 12232:2006(E), Photography – Digital still cameras – Determination of exposure index, ISO speed ratings, standard output sensitivity, and recommended exposure index, Standard, International Organization for Standardization, Geneva, CH, 2006.

[21] A. Ker, T. Pevny, The steganographer is the outlier: Realistic large-scale steganalysis, Information Forensics and Security, IEEE Transactions on 9 (2014) 1424–1435.

[22] A.D. Ker, P. Bas, R. Böhme, R. Cogranne, S. Craver, T. Filler, J. Fridrich, T. Pevný, Moving steganography and steganalysis from the laboratory into the real world, in: Proceedings of the first ACM workshop on Information hiding and multimedia security, IH&MMSec '13, ACM, New York, NY, USA, 2013, pp. 45–58.

[23] A.D. Ker, T. Pevný, A mishmash of methods for mitigating the model mismatch mess, in: Media Watermarking, Security, and Forensics 2014, Proc. SPIE, volume 9028, pp. 1601–1615.

[24] J. Kodovskỳ, J. Fridrich, Steganalysis of jpeg images using rich models, Media Forensics and Security, Proc. SPIE, volume 8303, pp. 83030A–83030A–13.

[25] J. Kodovsky, J. Fridrich, Effect of image downsampling on steganographic security, IEEE Transactions on Information Forensics and Security 9 (2014) 752–762.

[26] J. Kodovskỳ, J. Fridrich, V. Holub, Ensemble classifiers for steganalysis of digital media, Information Forensics and Security, IEEE Transactions on 7 (2012) 432–444.

[27] J. Kodovský, V. Sedighi, J. Fridrich, Study of cover source mismatch in steganalysis and ways to mitigate its impact, in: Media Watermarking, Security, and Forensics, volume 9028 of *Proc. SPIE*, p. 90280J.

[28] X. Kong, C. Feng, M. Li, Y. Guo, Iterative multi-order feature alignment for jpeg mismatched steganalysis, Neurocomputing 214 (2016) 458 – 470.

[29] X. Li, X. Kong, B. Wang, Y. Guo, X. You, Generalized transfer component analysis for mismatched jpeg steganalysis, in: 2013 IEEE International Conference on Image Processing, pp. 4432–4436.

[30] I. Lubenko, A.D. Ker, Steganalysis with mismatched covers: Do simple classifiers help?, in: Proceedings of the on Multimedia and Security, MM&Sec '12, ACM, New York, NY, USA, 2012, pp. 11–18.

[31] F. Marra, G. Poggi, C. Sansone, L. Verdoliva, A study of co-occurrence based local features for camera model identification, Multimedia Tools and Applications 76 (2017) 4765–4781.

[32] T.T. Nguyen, S. Oraintara, The shiftable complex directional pyramid—part ii: Implementation and applications, IEEE Transactions on Signal Processing 56 (2008) 4661–4672.

[33] J. Pasquet, S. Bringay, M. Chaumont, Steganalysis with cover-source mismatch and a small learning database, in: 2014 22nd European Signal Processing Conference (EUSIPCO), pp. 2425–2429.

[34] T. Pevný, T. Filler, P. Bas, Using high-dimensional image models to perform highly undetectable steganography, in: R. Böhme, P. Fong, R. Safavi-Naini (Eds.), Information Hiding, volume 6387 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2010, pp. 161–177.

[35] A. Polesel, G. Ramponi, V.J. Mathews, Image enhancement via adaptive unsharp masking, IEEE Transactions on Image Processing 9 (2000) 505–510.

[36] X. Qiu, H. Li, W. Luo, J. Huang, A universal image forensic strategy based on steganalytic model, in: Proc. IH&MMSEC, Salzburg.

[37] W.H. Richardson, Bayesian-based iterative method of image restoration∗, J. Opt. Soc. Am. 62 (1972) 55–59.

[38] V. Sedighi, J.J. Fridrich, R. Cogranne, Toss that bossbase, alice!, in: Media Watermarking, Security, and Forensics, Proc. IS&T, pp. pp. 1–9.

[39] T.H. Thai, R. Cogranne, F. Retraint, Statistical model of quantized DCT coefficients : Application in the steganalysis of jsteg algorithm, Image Processing, IEEE Transactions on 23 (2014) 1–14.