Batch steganography and pooled steganalysis in JPEG images

Ahmad Zakaria

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITÉ DE MONTPELLIER

En Informatique

École doctorale : Information, Structures, Systèmes

Unité de recherche : LIRMM

Batch Steganography and Pooled Steganalysis in JPEG

Présentée par Ahmad ZAKARIA

Le 26/11/2020

Sous la direction de Marc CHAUMONT et Gérard SUBSOL

Devant le jury composé de

Patrick BAS	DR-CNRS	Ecole centrale de Lille	Rapporteur
Philippe CARRE	PR	Université de Poitiers	Rapporteur
Sandra BRINGAY	PR	Université Paul Valéry Montpellier III	Examinatrice
Rémi COGRANNE	MCF	Université de Technologie Troyes	Examinateur
Marc CHAUMONT	MCF-HDR	Université de Nîmes	Directeur de thèse
Gérard SUBSOL	CR-CNRS	LIRMM	Encadrant

UNIVERSITÉ DE MONTPELLIER

Dedication

My father, my mother, your love and prayers are always with me pushing me forward.

Acknowledgement

Grateful to Almighty God for giving me the power and ability to achieve this goal.

To my supervisor, Dr. Marc CHAUMONT, and to my co-supervisor, Dr. Gérard SUBSOL, my sincere thanks and gratitude. With their guidance, assistance, advice and patience, I was able to face the challenges and difficulties of my research. Working under their guidance has been an honour for me.

Also, I would like to thank my examination committee: Professor Patrick BAS, Professor Philippe CARRE, Professor Sandra BRINGAY and Dr. Rémi COGRANNE for their time and efforts.

Special thanks to Dr. Rémi COGRANNE for his discussions and collaboration on this thesis. I would also like to thank the MESO@LR computing center in Montpellier, France, which provided a considerable amount of computing resources. I would particularly like to thank the French Direction Générale de l'Armement (DGA) for its support through the ANR Alaska project (ANR-18-ASTR-0009). Many thanks also to the administrative staff of the LIRMM laboratory and to all the colleagues of the ICAR team for their gentleness and helpfulness.

I lovingly thank my wonderful family, my mother, my father, my brothers and my precious sister. I could not have done this without your love, prayers and encouragement. Thank you for being by my side all the time. My friends, it is a blessing to have you in my life.

Finally, I would like to thank the Lebanese Ministry of Higher Education and scientific research, the Union of Municipalities of Jered Al-Kaytee, Akkar, Lebanon, and all the collaborators in the scientific research and higher education support program for funding my PhD thesis.

Contents

List of Figures

List of Tables

Part I

Introduction

Preface

Since ancient times, people have sought to communicate with each other in secret, because the communicated subject is sensitive and must be kept as a secret from third parties (adversaries).

The twenty-first century has experienced considerable technical progress in terms of the use of digital devices. In particular, these devices are used to store and share multimedia files such as text, audio, video and image files.

With the emergence of the Internet and social media which are widely used *communication channels*, this has led digital file sharing to become a daily standard practiced by many people in all sectors. The need for confidential and secure communications has then become a crucial aspect. Studies concerning the security, confidentiality and integrity of information and communications are an essential part of research under the headings of information hiding and communication security.

One of the critical demand in communication is the confidentiality of the information content. This confidentiality can be achieved through cryptography, which encode the message in such a way that it cannot be understood by unauthorized parties without knowing a password. But, in a cryptography application, we can see immediately that a message is present even if we cannot read it.

Another idea, is to hide the message in an imperceptible way. This is where the role of *steganography* lies, which propose methods to conceal a message in a file content so that communication appears normal and does not attract the attention of third parties, even if the message is not encrypted. Only authorized persons have both some hints on the presence of the *message* and its characteristics (as for example its size), they can use the right method and eventually a key (called the stego-key) to locate and extract the message.

Steganography history

Steganography literally means, "covered writing" which is derived from the Greek language steganos-graphy, steganos means (covered or secret) and graphy means (writing or drawing). Ancient steganography was applied to a physical cover; a message was hidden in this cover so that it did not appear for the eye and therefore did not raise suspicions of untrustworthy persons.

An ancient example of the use of steganography in ancient times, is the use of wooden tablets to write a letter and then cover it with wax [Babington, 1996] to hide it, as shown in *Figure* 1.

FIGURE 1: Hidden message in wax tablet.

Throughout history, many steganography techniques based on making invisible changes in a visible document has been proposed. One example is to use the text lines offset up or down by 1/300 of an inch to code zeros and ones [Brassil *et al.*, 1995], or changing the height of letter strokes and or marking letters in a text using small holes, or by signs [Whitehead, 2003]. Another example is the microdots technology introduced by Germans between World War I and World War II as shown in *Figure* 2, they were attached into a regular document and may contain a full document or image. Another example is the invisible ink, which was used by applying natural liquid that requires heat to appear, like orange juice and milk, or chemicals that reacts to certain chemicals [Kahn, 1996].

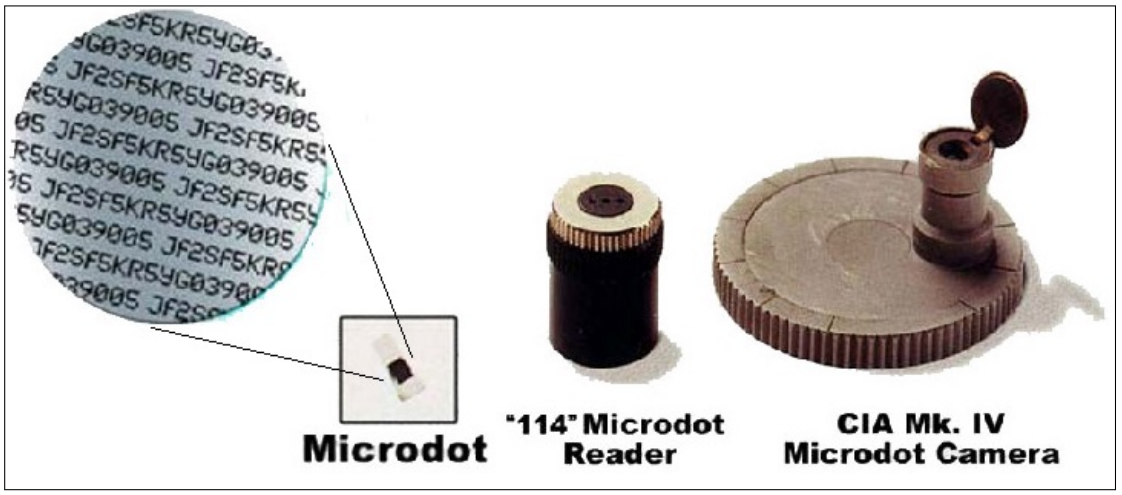

FIGURE 2: Examples of ancient steganography using German microdots.

Eventually, steganography techniques have transformed as the world has evolved into digital technology. Modern steganography has become in digital objects. *Figure* 3 and *Figure* 4 show the annual number of steganography and steganalysis research articles reported by Google Scholar[1]. Through these tables, we see the enormous progress in the number of articles published annually, which highlights the importance of the field of steganography/steganalysis, especially with enormous evolution of the digital technologies.

Modern Steganography

Modern steganography is associated with digital files and the use of the internet and smart devices, as they have replaced the old communication techniques. The use of steganography differs from legitimate and malicious, despite that in media, steganography is considered a secret communication between criminals or extremists. In general, steganography is a hot topic in the media because of its seriousness and critical objectives.

In this section, we discuss some recent news articles and reports that point to the use of modern steganography for legitimate and malicious purposes.

[1]We did an online search with the keyword "steganography" on Google scholar, combining all publications, year by year, from 2000 to 2018 years.

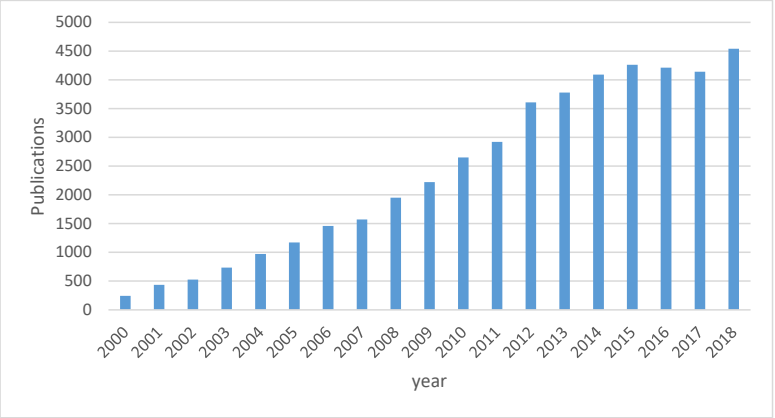

FIGURE 3: The annual number of research articles on the Google scholar search engine for the keyword steganography.

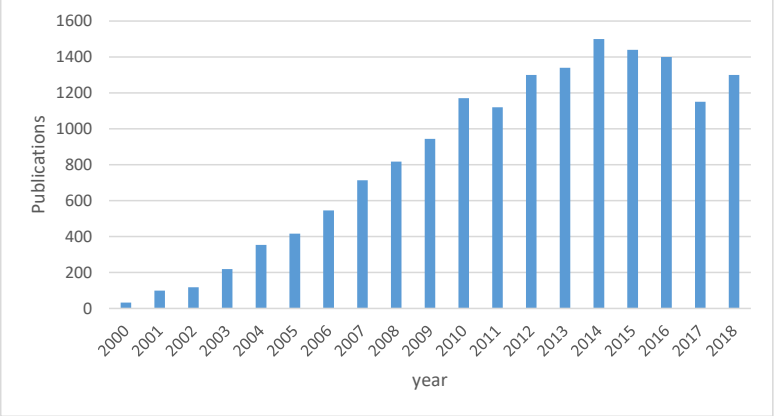

FIGURE 4: The annual number of research articles on the Google scholar search engine for the keyword steganalysis.

Not all uses of steganography are considered malicious or illegal, for example, academic researchers, security agents who apply steganography techniques, or simply curious people who learn and use steganography just out of curiosity. In [Kennedy, 2004], the author cited various purposes for the legitimate use of steganography and data hiding techniques.

The legitimate use of steganography does not attract the media and news agencies, unlike its illegal or malicious use. With a simple online search for "news articles on steganography", one can obtain a large number of articles citing steganography, as expected, all of which speak of malicious use. For example, one can find articles

about the use of steganography by spies, terrorists, hackers, drug dealers ...

In 2010, a newspaper article [Leyden, 2010] cited a spy swap event between the United States and Russia. The author mentioned the use of steganography: "The illegal network used a variety of technology to keep in touch with their controllers, including steganography and ad-hoc Wi-Fi networks". In the same year, more articles mentioned similar spying actions, we are not sure if they are the same due to the ambiguity of the articles, as this is a sensible subject to be published in all details. Fox-news [Fox-News, 2010] cited that the American authorities arrested alleged Russian spies in Massachusetts. The authors also mentioned the use of steganography in images: "During a 2006 search of the couple's townhouse, law enforcement agents found traces of deleted electronic messages on computer disks that were believed to be drafts of messages that were later sent to the Moscow headquarters using a process called steganography, in which encrypted data can be placed in images on publicly available websites, without the data being visible to a casual user.". In addition, [Fox, 2010] cited spies who have hidden codes in online photos. The author mentioned that this discovery marks the first real use of steganography in espionage. Also, the author mentioned that "The accused spies posted the seemingly mundane photos on publicly accessible websites, but then extracted coded messages from the computer data of the pictures, according to the criminal complaint filed by the FBI. ".

Recently, hackers have begun to take advantage of steganography to reinforce their malware, as evidenced by the large number of articles published on the Internet. In 2017, two cybersecurity reports *Kaspersky* [Shulmin and Krylova, 2017] and *McAfee* [McAfee, 2017] cited the relation between steganography and malware and the importance they will have in the future. In [DELL, 2014], the author provides an analysis of the *Lurk Downloader* malware. "Lurk is a malware downloader that uses digital steganography" according to author. In [Stein, 2019], the author revealed how steganography is used for a JavaScript malware that hides in image files: "Recent months have seen an uptick in reports of JavaScript malware that hides in image files. This is often referred to as *image based malware* or *steganography malware* in more technical contexts.".

In [SentinelOne, 2019], the author explains how malware hides code inside images and how it uses steganography. He cited some of the latest ones: "Zbot – appends data to the end of a JPEG file containing hidden data", "Cerber – embeds malicious code in image files", "ZeroT – Chinese malware that uses steganography to hide malware in an image of Britney Spears".

Despite that majority of steganography techniques are for digital images, other trends are becoming available, such as steganography in VoIP (Voice over Internet Protocol) streams [Huang *et al.*, 2017, 2011; Mazurczyk and Szczypiorski, 2008] or in TCP/IP [Murdoch and Lewis, 2005; Dhobale *et al.*, 2010; Lu *et al.*, 2016] which make the issue more complicated and requires more efforts to control.

The spreading of the problem of steganography in this suspicious form requires close cooperation between law enforcement agencies, research centres and specialised professionals from the industry. This cooperation has begun to become evident, and we see that in the *Criminal Use of Information Hiding (CUIng)*[2] initiative.

Steganalysis challenges

In the latest two decades, research on steganography has made remarkable progress. In particular, progress has been made on steganography in digital images.

In this Ph.D. thesis we deal with the steganalysis of digital images. Very soon, images have been a media which were used to insert quite long messages in an efficient way, thanks to the advanced and in-depth studies on embedding algorithms and techniques [Cole, 2003], [Fridrich, 2009], [Desoky, 2016]. In this thesis, we focus on digital images coded in JPEG, which is the most common format nowadays.

In general, the evolution of steganography techniques coincides with the evolution of *steganalysis*. The purpose of steganalysis is to determine whether or not

[2] http://cuing.org/

the message is present and eventually to know some parameters (as the *message length*). The slightest doubt about the existence of secret information is a failure of the entire steganography system. By analyzing the characteristics of a digital image, the steganography detector algorithm will be able to detect if there was an insertion (embedding) of a message in the image. This is a complicated task because of the image content and then, its characteristic can be very diverse due to the image construction process or acquisition. In addition, in JPEG images, lossy compression adds specific characteristics to the images that must also be distinguished by the *steganography detector*.

When the purpose of steganalysis is to distinguish between clean images (named *cover* images in the following) and embedded images (named *stego* images in the following), it is considered to be a classification problem, and it is the most studied in this field. We name it *binary steganalysis* in the this thesis. Binary steganalysis algorithms are named *binary detectors*. But a steganalyst may want more information than just the presence or absence of the hidden data in a media file, such as the size of the hidden message (called the *payload* size) or other details. Finding out the size of the message (the payload size) is a task named *quantitative steganalysis*, and can be considered as a prediction problem. In this case we have *quantitative steganalysis detector*. Today, additionally to the steganography, we can think that steganalysis has reached a high level of development [Yahya, 2019], [Hassaballah, 2020], especially with the latest AI techniques based on Deep Learning [Chaumont, 2020].

Since 2006 [Ker, 2006], some more realistic questions starts to be asked in the domain of steganography/steganalysis. For example, when the steganographer hides a message in a set of images (named *bag* of images), how can he spread his message into this set, and what is the best way to do it? In this case, the steganalyst must examine the set of images. But, is it better to steganalyze the images one after the other in order to see if some of them contain a part of the message or is it better to develop an algorithm which will work with all the images? As a result, the problem of steganography and steganalysis is reformulated, and it becomes *batch steganography* and *pooled steganalysis*.

Batch steganography consists of hiding a message by spreading it over several images using a *spreading strategy*, while pooled steganalysis consists of analyzing a set of images to conclude whether a message is present or not, using a *pooling function* which aggregates clues (*scores*) from the set of images.

In image pooled steganalysis, one can reasonably assume that the steganalyst does not know the exact strategy used to spread the payload across images, but he can guess that the steganographer and his peer use a known embedding scheme.

What now if the steganalyst could discriminate the spreading strategy? Could the steganalyst get results close to the scenario where it is assumed that he knows the spreading strategy? In this thesis, we try to answer these questions by studying different possibilities for a pooled steganalysis architecture that is built using several blocks or algorithms.

Content of the thesis

The dedication of Part II is to discuss the state-of-the-art works in the domain of steganography/steganalysis, where we introduce the basic concepts and tools that will be necessary to understand the rest of the thesis.

In Part II-*Chapter* 1, we present the widely used JPEG standard format, which is strongly related to steganography and steganalysis. Because of the specificity of this format, only specific algorithms are effective. hence, we need to base our work on these algorithms. We recall the basics of the JPEG compression and decompression processes.

In Part II-*Chapter* 2, we describe some steganographic techniques, in JPEG, from the early development of steganography as a research discipline to the present day. We begin with an overview of the basic tools still in use in this field. We continue by discussing some of the naive techniques. Next, we discuss the adaptive steganography – the most modern steganography. We discuss the elementary building blocks of adaptive modern embedding schemes, focusing on the distortion

function. We conclude this chapter by presenting J-UNIWARD, the state-of-the-art embedding scheme which is a very efficient technique tuned for JPEG that we will use.

In Part II-*Chapter* 3, we discuss the concept of conventional steganalysis where steganalysis has taken the form of a warden who attempts to analyze an image to detect the presence of a hidden message using binary steganalysis or to estimate its size using quantitative steganalysis. We begin with an overview of steganalysis, discussing the warden's scenarios. In addition, we present some earlier binary and quantitative detectors. Next, we discuss the latest approaches to binary steganalysis based on feature-based machine learning (two-steps learning) and deep learning approaches (one-step learning). Finally, we discuss the latest approaches to quantitative steganalysis also based on two-steps learning and one-step learning.

The objective of Part II-*Chapter* 4 is to understand the basics of batch steganography, on how the steganographer can spread the message payload within a set of images. Next, we redefine some of the concepts of conventional steganography to make them relevant to batch steganography. Finally, we motivate our work by listing many of the state-of-the-art batch spreading strategies.

In Part II-*Chapter* 5, we present the pooled steganalysis. First, we define the *Single-Image Detector* (SID) as the first block of the architecture of the pooled steganalysis, and the pooling function as the second part. Next, we detail the already proposed pooling functions with their corresponding hypotheses. We also highlight the pooling functions that are supposed to be optimal for certain spreading strategies.

Part III is devoted to the contributions we have made to the problem of batch steganography and pooled steganalysis.

The Part III-*Chapter* 6, is the first contribution of this thesis. In this chapter, we performed a comparison between quantitative steganalysis algorithms in order to decide which one is best suited for pooled steganalysis. We also propose to extend this comparison to binary steganalysis algorithms. For this we propose

a methodology to use both types of algorithms for both binary and quantitative steganalysis.

The Part III-*Chapter 7* is the core of the thesis lies in the second contribution. We study the scenario where the steganalyst does not know the spreading strategy. We propose a discriminative pooling function, optimized on a set of spreading strategies, that allows to improve the accuracy of the pooled steganalysis compared to a simple average. This pooling function is learned using supervised learning techniques.

The thesis is concluded in Part IV, where we summarize the contributions, discuss possible future directions based on our results.

Notations

In this section, we provide the notations used throughout this manuscript. More specific notations are also introduced.

- Scalar b: lowercase letter in italic represent scalars.
- Matrix or vector of scalars $\mathbf{x}$: Bold is used for vectors (lower case) or matrices (upper case).
- Set $\mathcal{X}$: calligraphic font is reserved for sets. As for example, a set of cover images $\mathcal{C}$, a set of cover images $\mathcal{S}$, a set of keys $\mathcal{K}$, a set of messages $\mathcal{M}$.
- $f(.)$: a function which can be vectorial (for example to return a feature vector from an image) or scalar (to compute a score, given a feature vector or an image).

Part II

State Of The Art

Chapter 1

JPEG, A Standard Format

Contents

The Joint Photographic Expert Group (JPEG) format is the most commonly used compression standard for images nowadays. JPEG compression is used to reduce the size of the image by degrading the image in an imperceptible way, i.e. invisible to the naked eye perception as the human visual system is insensitive to small colour changes or high spatial frequency noise [Pennebaker and Mitchell, 1992].

The inverse process is the JPEG decompression, which is applied to reconstruct image pixels from a compressed JPEG file. *Figure* 1.1 illustrates the JPEG compression/decompression cycle for a given image. In this chapter, we briefly explain the two processes with some hints on where exactly the steganography process can be integrated, and how a steganalysis process can be applied.

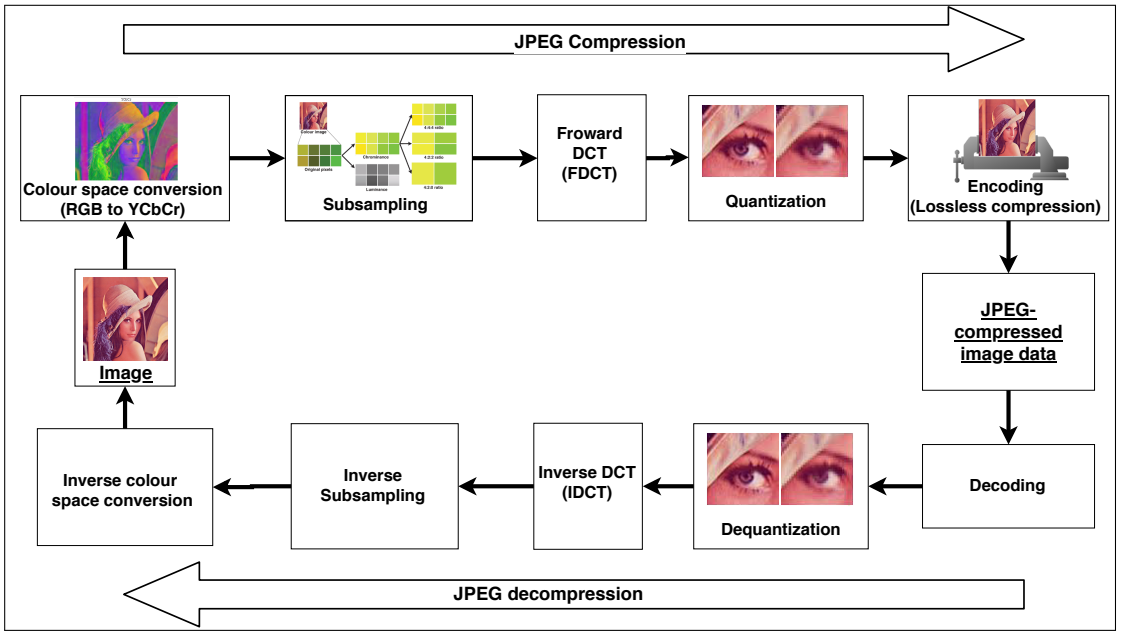

FIGURE 1.1: JPEG compression/decompression cycle.

1.1 JPEG compression

The JPEG compression process for a RGB colour image is composed of five steps:

1. YC_bC_r conversion,
2. Subsampling and block splitting,
3. Discrete cosine transform (DCT),
4. Quantization,
5. Lossless compression,

1.1.1 YC_bC_r conversion

The colour is converted from RGB (Red, Green, Blue) to YC_bC_r using the following equation:

$$\begin{cases} Y = 16 + \frac{65.738R}{256} + \frac{129.057G}{256} + \frac{25.064B}{256}, \\ C_b = 128 - \frac{65.738R}{256} + \frac{129.057G}{256} + \frac{25.064B}{256}, \\ C_r = 128 + \frac{65.738R}{256} + \frac{129.057G}{256} + \frac{25.064B}{256}, \end{cases}$$

where Y represents the luminance channel, C_b the blue chrominance channel (the hue saturation of an image) and C_r the red chrominance channel (the colour saturation of an image). The importance of this conversion is that it limits the luminance information to a single Y channel which matches more closely the perception of colour in the human visual system [Ryan, 2012].

1.1.2 Subsampling and block splitting

After the YC_bC_r conversion, the chrominance and luminance channels are separated. As mentioned before, the human eye is more perceptible for luminance.

Hence it is possible to delete some colour information from C_b and C_r channels without losing too much quality; this is the purpose of chrominance subsampling (also called downsampling).

More deeply, suppose that we have 4×2 block of pixels for chrominance channels (C_r, C_b), and we need to subsample the chrominance channels of the original pixels, to do that, there are three possible resulted ratios as shown in *Figure* 1.2:

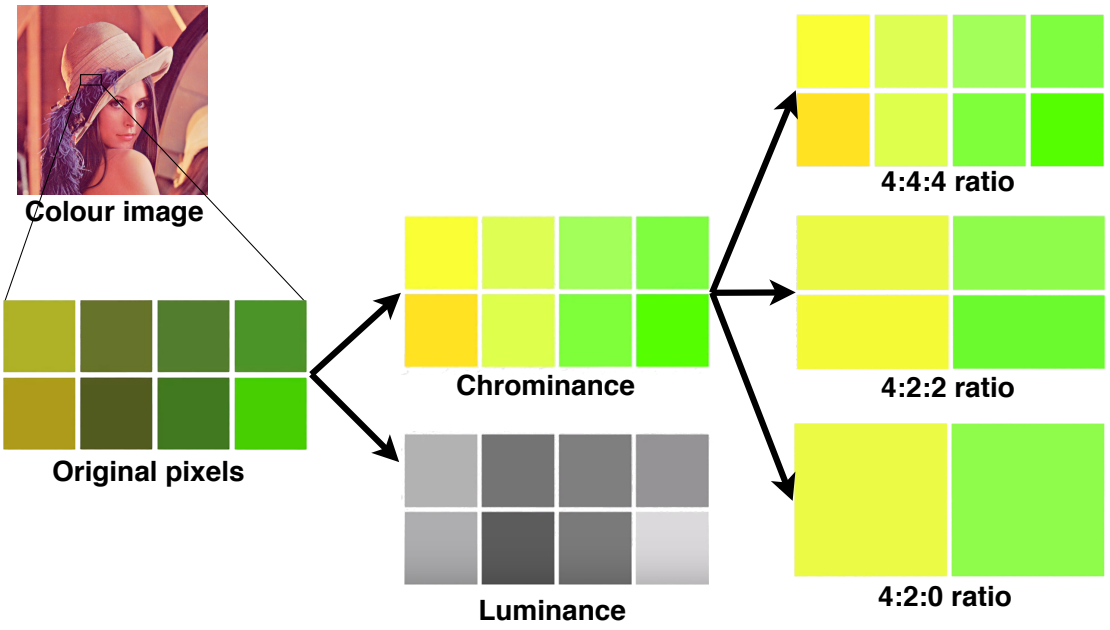

FIGURE 1.2: Chroma subsampling types.

- **4:4:4 :** A ratio of 4:4:4 means that no subsampling is done, this is equivalent to the original image, i.e. each pixel keeps its colour information. The first value (4) is for the width of the 4×2 block, the second value (4) means that 4 pixels from the first row keep their values, the third value (4) means that 4 pixels from the second row keep their values, as shown in *Figure* 1.3.

- **4:2:2 :** The first value (4) is for the width of the 4×2 block, the second value (2) means that 2 pixels from the first row keep their values, the third value (2) means that 2 pixels from the second row keep their values, as shown in *Figure* 1.4.

- **4:2:0 :** The first value (4) is for the width of the 4×2 block, the second value (2) means that 2 pixels from the first row keep their values, the third

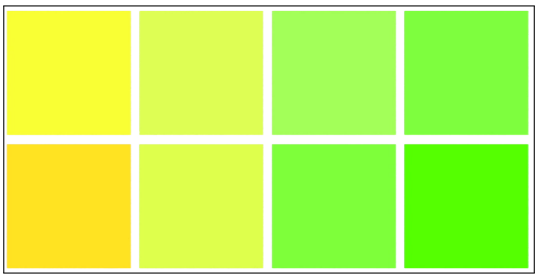

FIGURE 1.3: 4:4:4 ratio

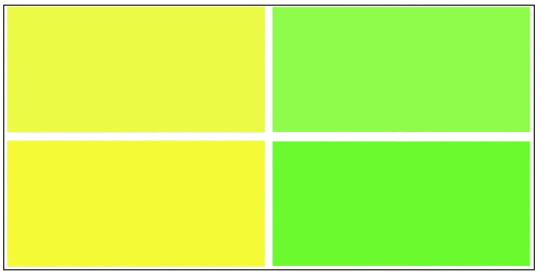

FIGURE 1.4: 4:2:2 ratio

value (0) means that non of the pixels from the second row keep their values which mean that the second row has the same colours of the first one, as shown in *Figure* 1.4.

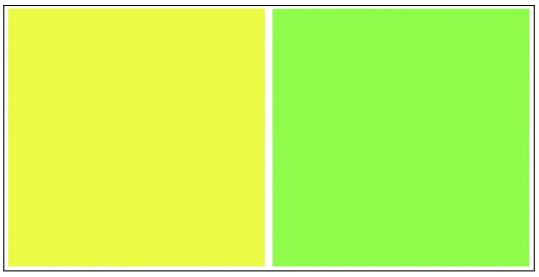

FIGURE 1.5: 4:2:0 ratio

Finally, each channel is divided into blocks of 8×8 pixels. Obviously, for grayscale images, there is no need for chroma subsampling, only the Y channel is retained for the next step, also sampled in blocks of 8×8.

1.1.3 Discrete cosine transform (DCT)

The goal of the DCT transform is to convert the pixel YC_bC_r values of the channels to frequency values. To do so, the Discrete Cosine Transform (DCT) is used. DCT

is a sinusoidal function, it takes as input a pixel value $P[i,j]$, $0 \leq i \leq 7$, $0 \leq j \leq 7$ of an 8×8 block of pixels P, and convert it to a frequency value $F[k,l]$, $0 \leq k \leq 7$, $0 \leq l \leq 7$ from an 8×8 block of frequencies F.

Before applying the DCT function, the pixel values in P are shifted from range $[0, 255]$ to $[-128, 127]$ by merely subtracting the values by 128. In the compression phase, the DCT function is called Forward DCT (FDCT) [Wallace, 1991] and is given by:

$$F[k,l] = \frac{1}{4}\gamma(k)\gamma(l)\sum_{i=0}^{7}\sum_{j=0}^{7} P[i,j]\cos\left(\frac{(2i+1)k\pi}{16}\right)\cos\left(\frac{(2j+1)l\pi}{16}\right), \qquad (1.1)$$

where

$$\gamma(k),\ \gamma(l) = \begin{cases} \frac{1}{\sqrt{2}},\ \text{if } k,l = 0, \\ 1,\ \text{otherwise.} \end{cases}$$

Each pixel information $P[i,j] \in P$ is then spread to all the 8×8 DCT block since each value $F[k,l]$ depends on all values in the pixel block, as we see in *Equation* (1.1). The first coefficient of the DCT block $F[0,0]$ is called the DC coefficient which is the coefficient with zero frequency in both dimensions. The remaining 63 coefficients are called AC coefficients, these are the coefficients with non-zero frequencies.

1.1.4 Quantization

The purpose of the quantization step is to reduce the number of bits used to store an image by reducing the amount of information in the high-frequency components (ACs). This method is related to the fact that the naked eye cannot distinguish the exact intensity of a variation in brightness at high-frequency [Godse and Godse, 2009].

Technically, quantization is done by individually dividing its DCT coefficients by values from a quantization matrix Q, and then round them to the nearest integer, as shown in *Equation* (1.2).

$$T[k,l] = \text{round}\left(\frac{F[k,l]}{Q[k,l]}\right) \tag{1.2}$$

The resulting matrix T, called the quantized DCT matrix, is used in the next step. The values in Q depends on the quality factor of the compressed image. *Figure* 1.6 shows a quantization table for quality factor 75.

$$\begin{bmatrix} 8 & 6 & 5 & 8 & 12 & 20 & 26 & 31 \\ 6 & 6 & 7 & 10 & 13 & 29 & 30 & 28 \\ 7 & 7 & 8 & 12 & 20 & 29 & 35 & 28 \\ 7 & 9 & 11 & 15 & 26 & 44 & 40 & 31 \\ 9 & 11 & 19 & 28 & 34 & 55 & 52 & 39 \\ 12 & 18 & 28 & 32 & 41 & 52 & 57 & 46 \\ 25 & 32 & 39 & 44 & 52 & 61 & 60 & 51 \\ 36 & 46 & 48 & 49 & 56 & 50 & 52 & 50 \end{bmatrix}$$

FIGURE 1.6: Quantization table Q for quality factor 75.

1.1.5 Lossless Compression

The latest step to create a compressed JPEG file is the lossless compression, see *Figure* 1.7. This step starts with the *Zig-Zag Scan*, a way to rearrange the DCT coefficients, in order to cluster the coefficients with similar frequencies together. Next, the *Run-Length Encoding* (RLE) algorithm compresses the AC coefficients, and the *Differential Pulse Code Modulation* (DPCM) algorithm compresses the DC coefficient. Finally, a Huffman or arithmetic coding algorithm is applied to create the final JPEG file.

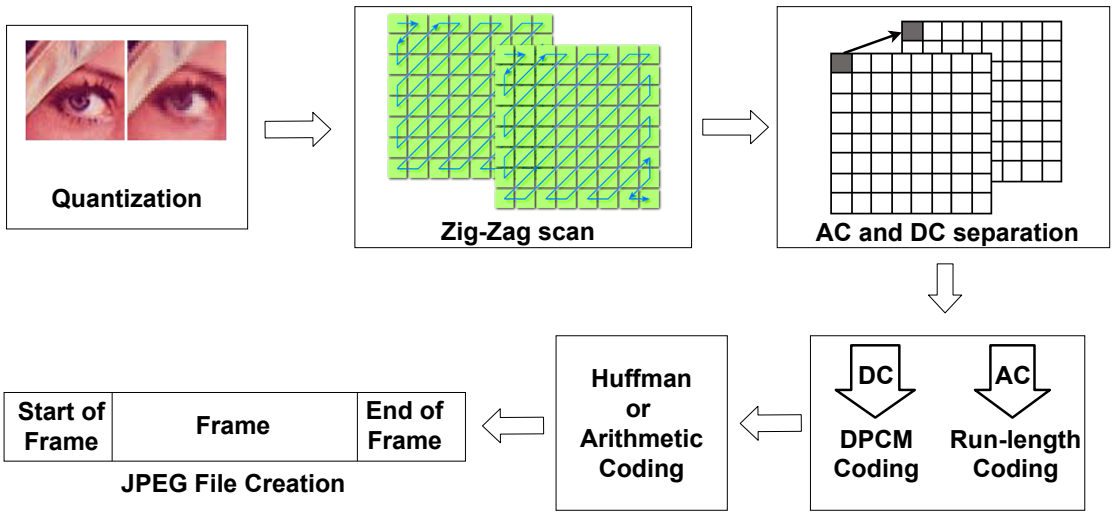

FIGURE 1.7: Lossless compression, just after the quantization step.

1.2 JPEG decompression

The decompression process is applied to obtain the original image from a compressed one. The same above steps are applied but oppositely, as shown in *Figure* 1.1, in particular, The DCT become the Inverse DCT (IDCT) [Wallace, 1991]:

$$P[i,j] = \frac{1}{4}\sum_{k=0}^{7}\sum_{l=0}^{7}\gamma(k)\gamma(l)F[k,l]\cos\left(\frac{(2i+1)k\pi}{16}\right)\cos\left(\frac{(2j+1)l\pi}{16}\right) \tag{1.3}$$

1.3 Conclusion

This chapter recalls the basics of the JPEG compression and decompression processes in color images, it also notes how these processes differ with grayscale images. In this thesis, we will use only grayscale images that are coded on an unique channel instead of three channels.

Therefore, in the compression process, this set of intensities is given directly to the DCT transformation step after the block splitting step [Dilpazir *et al.*, 2012].

Chapter 2

Conventional Steganography In JPEG

Contents

In this chapter, we describe some steganographic techniques in JPEG, from the beginning of the development of steganography as a research discipline to the present day. We begin with an overview of the basic tools still in use in the field. We continue by discussing some of the naive techniques used in steganography and how they have developed over time, and we conclude by discussing state of the art techniques.

2.1 Overview of steganography in JPEG

2.1.1 Steganographic channel

A steganographic channel is an environment necessary to perform steganographic communication. In this environment, messages are hidden in innocent-looking images, which are then communicated transparently.

Alice, the steganographer, needs a *cover* image (a JPEG image in this thesis), to embed her *message* by using an *embedding* algorithm, and a *stego key*. Besides, before sending stego images, Alice shares with the receiver, Bob, some information about the steganographic operation: the *extraction algorithm* and the *stego key* before transferring the stego image. The shared information will allow Bob to extract the message from the cover image. All elements are represented in *Figure* 2.1.

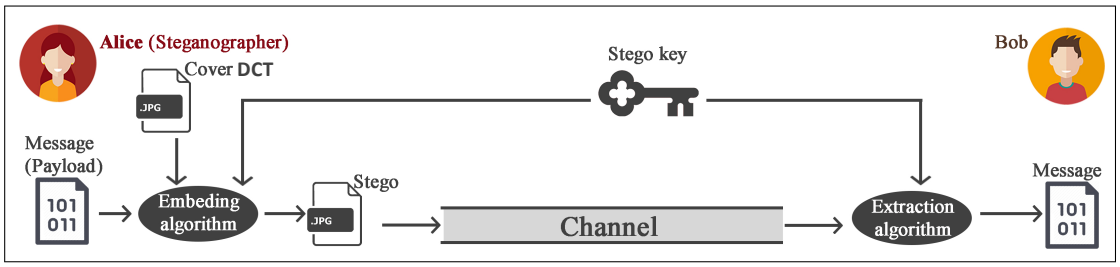

FIGURE 2.1: The steganographic channel and the principal elements

Operations in a steganographic channel are theoretically formulated as functions that take images as inputs, manipulate it (embed, extract) and give results as outputs. Given a set of cover images $\mathcal{X}$, a set of keys $\mathcal{K}$, associated with the

cover images, and a set of messages $\mathcal{M}$, that Alice can embed, the embedding and extraction are expressed as :

$$Emb : \mathcal{X} \times \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{X} \tag{2.1}$$

and

$$Ext : \mathcal{X} \times \mathcal{K} \longrightarrow \mathcal{M} \tag{2.2}$$

2.1.2 LSB replacement

The LSB replacement is one of the first embedding techniques. The LSB approach was introduced in the watermarking discipline in 1993 [Tirkel *et al.*, 1993] where two techniques have been presented for hiding data in spatial domain images. These methods were based on modifying the least significant bit (LSB) of the pixel using an algorithm proposed by [Kurak and McHugh, 1992]. This algorithm was known as image downgrading [Zheng *et al.*, 2007; Bamatraf *et al.*, 2011].

In the following, we explain how the LSB replacement technique works in JPEG domain. The embedding operation (*Equation* (2.1)), that describes the LSB replacement works as follows: first, the message to be embedded is converted to bit-stream. Then, the message bits overwrite the least significant bits in the image quantized DCT values. The message can be embedded sequentially or pseudo-randomly using a stego key.

The extraction operation (*Equation* (2.2)) works by simply extracting the LSBs from the stego image and rebuilding it using the stego key that describes the order of the message bits.

We can see from *Figure* 2.2 how the LSBs, the bold green values, are modified to handle all the 8-bit $(0, 1, 0, 0, 1, 1, 1, 1)^T$ binary values (in bold red).

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & \mathbf{0} \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & \mathbf{1} \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & \mathbf{0} \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & \mathbf{1} \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

FIGURE 2.2: LSB replacement example. The quantized DCT values are embedded by replacing their least significant bits by the message bits (in bold red). The values in bold green color are the new LSB values of the quantized DCTs.

2.1.3 LSB matching

The LSB matching is almost similar to the LSB replacement technique; the difference is in what the algorithm does after converting the image quantized DCT values and message data to bit-stream. The LSB matching embedding method was proposed for the first time by [Sharp, 2001]. It is worth mentioning that all modern and secure steganographic schemes are based on embedding by *LSB matching*.

The embedding operation (*Equation* (2.1)) that describes the LSB matching works as follows: first, the message to be masked is converted to a bit-stream. Next, for each value of the image and its corresponding message bit, if the LSB and its corresponding message bit are different, a random binary operation (binary addition or binary subtraction[1]) is performed; otherwise, nothing is done. When choosing the random binary operation, a condition is respected, which is to avoid bypassing the range of values in image quantized DCTs, which must be belonging to {-1023, . . . , 1024}.

The extraction operation (*Equation* (2.2)) has the same complexity than the LSB replacement. It works by extracting the LSBs from the stego image and rebuilding the message using the stego key that describes the order of the message bits.

[1]The LSB matching is also called ±1 *embedding*.

In *Figure* 2.3, we apply the LSB matching embedding on the example of the *Figure* 2.2. The bold green values, are modified due to the random binary operations with the 8-bit $(0,1,0,0,1,1,1,1)^T$ binary values (in red). In this Figure, we can see that the algorithm perform the binary operations only when the values of the message and the LSBs are different.

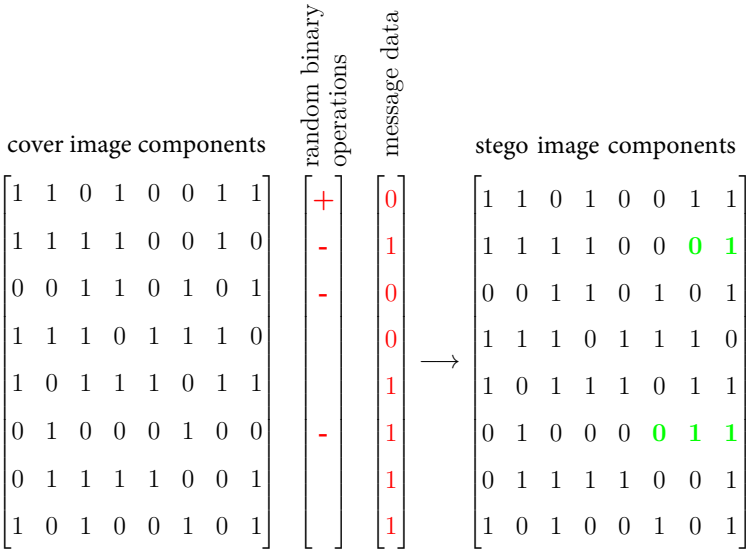

FIGURE 2.3: LSB matching example

2.1.4 Syndrome coding: matrix embedding

The idea of using Hamming codes in steganography was first proposed by [Crandall, 1998] as a matrix coding technique to enhance the embedding efficiency for a steganographic scheme. The goal is to have as few modifications as possible while inserting as many bits as possible. In fact, the idea was studied the same year in a previously unpublished article [Bierbrauer, 2001].

Hamming code is a linear error-correcting code that can detect and correct single-bit errors. The $(n, n-k)$ Hamming code uses n bits to transmit $n-k$ message bits, the remaining k bits used for error-correcting purpose are called *parity check* bits, where $n = 2^k - 1$ on the binary field, which contains binary numbers.

Given a message bits $\mathbf{m}$ of size $n-k$. To transmit $\mathbf{m}$ to a receiver via a noisy communication channel, $\mathbf{m}$ is transformed to n bits sized code word $\mathbf{y}$ using a

$(n-k) \times n$ *code generator matrix* $\mathbb{G}$ as follows:

$$\mathbf{y} = (\mathbf{m}^T \times \mathbb{G})^T \tag{2.3}$$

Let $\mathbf{x}$ of size n, be the code word received by the sender of $\mathbf{y}$. The $k \times n$ sized *parity check matrix* $\mathbb{H}$ is used to compute the k sized *syndrome* vector $\mathbf{z}$ for checking an error as follows:

$$\mathbf{z} = \mathbb{H} \times \mathbf{x} \tag{2.4}$$

If the syndrome vector is $\mathbf{z} = 0_k$, the receiver can conclude that no error has occurred. Otherwise, given a vector position $i \in \{1, \ldots, n-k\}$, if the generator $\mathbb{G}$ and the parity check matrices are organised correctly, and if $\mathbf{z}$, interpreted as a number, then an error is detected at the ith position of $\mathbf{x}$. Thus $\mathbf{x}$ is corrected by flipping the ith value in $\mathbf{x}$. Technically, the flipping can be made by applying $\mathbf{x} = \mathbf{x} \oplus e_i$, where $\oplus$ is the XOR operation, and e_i, the error pattern, i.e. a unit vector of length n.

The problem of matrix embedding is formalised as follows: The principle is to modify the cover support $\mathbf{x}$ to $\mathbf{y}$, given $\mathbf{m}$ so that :

$$\mathbb{H}\mathbf{y} = \mathbf{m}, \tag{2.5}$$

with $\mathbb{H}$ parity-check matrix of the code. The transformation of the vector $\mathbf{x}$ into $\mathbf{y}$ is then done by searching for the modification vector e:

$$\mathbf{y} = \mathbf{x} + e. \tag{2.6}$$

From *Equation* (2.6) and *Equation* (2.5), we then obtain the following formula:

$$\mathbb{H}(\mathbf{x} + e) = \mathbf{m} \Longleftrightarrow \mathbb{H}e = \mathbb{H}\mathbf{x} - \mathbf{m}. \tag{2.7}$$

The problem is therefore to find the vector e, which is a code word having as a syndrome ($\mathbb{H}\mathbf{x} - \mathbf{m}$), since the goal is to make as minimum modifications as possible on the support $\mathbf{x}$. Once Alice has computed $\mathbb{H}\mathbf{x} - \mathbf{m}$, she converts the vector to a number, i, of base 10 and that number is the i^{th} bit of $\mathbf{x}$ that needs to be flipped. For more details we refer the reader to [Westfeld, 2001; Fridrich, 2009].

We give a simple example on the Hamming code to clarify the concept. Suppose that the message $m = (1, 0, 1, 1)^T$ is transferred to a receiver, a (7, 4) Hamming code is used to code the message at the receiver side, and correct the code if necessary. First, the code word $\mathbf{y}$ for the message $\mathbf{m}$ is calculated using *Equation* (2.3):

$$\mathbf{y} = \left(\begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}^T \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \right)^T = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

The receiver receives a code word $\mathbf{x}$. Let us suppose that there is an error, such that the received word is $\mathbf{x}$= $(1, 0, 1, 1, 1, 0, 1)^T$. Now the *Equation* (2.4) is used to calculate the syndrome $\mathbf{z}$ to check if any error in $\mathbf{x}$:

$$\mathbf{z} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$\mathbf{z}$ =$(1, 0, 0)^T$, $\mathbf{z}$ interpreted as 5, and so identical to the 5th column in $\mathbb{H}$. The correction is $(1, 0, 1, 1, 1, 0, 1)^T \oplus (0, 0, 0, 0, 1, 0, 0)^T = (1, 0, 1, 1, 0, 0, 1)^T = \mathbf{y}$.

In steganography, with the matrix embedding approach, the message to be embedded in a set of quantized DCT values is related to the *syndrome*. Hence, for a message of size k, we need a $(2^k, 2^k - k)$ Hamming code. Staying in the same Hamming concept, Alice uses the Hamming code to find the position needed to change in order to simulate the same change (error) in the destination message.

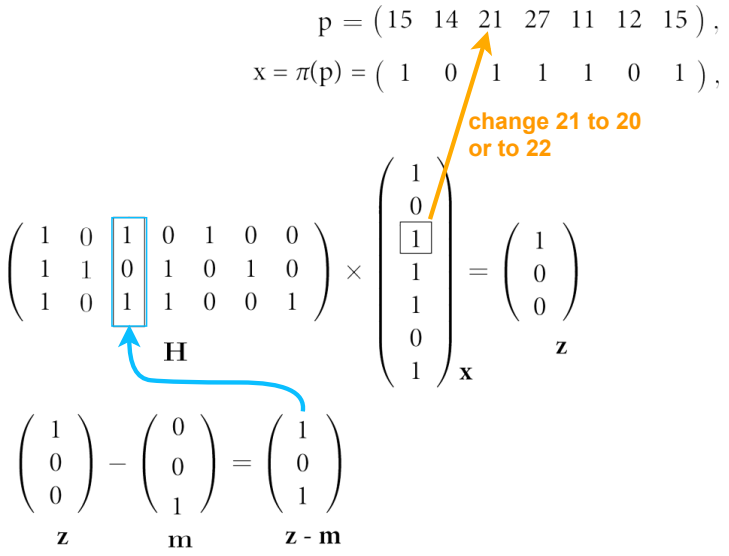

FIGURE 2.4: Embedding using binary Hamming codes: a simple example.

Let us give an example of matrix embedding for the set of quantized DCT values $p = (15, 14, 21, 27, 11, 12, 15)$ and a message $\mathbf{m} = (0, 0, 1)^T$. The example is illustrated in *Figure* 2.4. Alice first converts the quantized DCT values to bits, $\mathbf{x} = p \ mod \ 2 = (1, 0, 1, 1, 1, 0, 1)^T$; the values of $\mathbf{x}$ are thus the LSBs of p. Next, Alice computes the syndrome of $\mathbf{x}$, which gives $\mathbf{z} = (1, 0, 0)^T$, using the $\mathbb{H}$ matrix from the previous example. Then, she calculates the vector $\mathbf{z} - \mathbf{m} = (1, 0, 1)^T$ and tries to find it between the columns in $\mathbb{H}$. It is the third column. Thus, all what she needs to embed $\mathbf{m}$ in the block, is to flip the LSB of the third quantized DCT value. For example, she may change $p[3] = 21$ to 20. From 21 to 22, she thus gets the same result for both choices.

2.1.5 Wet paper coding

After the explanation of the linear error correction codes and the syndrome coding by an example based on Hamming codes, and the generalisation of the problem in *Equation* (2.7), we discuss a solution to improve the problem of the *Equation* (2.7) which is called the wet paper coding.

It was proposed by [Fridrich *et al.*, 2005] to improve the undetectability of embedding by allowing Alice to lock certain components of the cover data, in accordance with the nature of the cover image and message [Augot *et al.*, 2011]. Generally, it consists in preventing the use of certain elements of the host sequence, but in a strict way ("yes" or "no" only), to ensure a high undetectability (elements are called "wet" when one cannot modify them).

In the original article [Fridrich *et al.*, 2005], the basic concept of the wet paper code is defined as follows: Given a cover image $\mathbf{x} = \{x_i\}_{i=1}^{n}$ of n components, $x_i \in \xi$, $\xi = \{-1023, ..., 1024\}$, Alice wants to create a stego image $\mathbf{y} = \{y_i\}_{i=1}^{n}$ using $\mathbf{x}$ and send it to Bob. She uses what is called a Selection Rule (SR) to select k changeable elements x_j, $j \in C \subset \{1, \ldots, n\}$, $|C| = k$. Only x_j elements may be changed during embedding.

Alice and Bob use a shared stego key to generate a pseudo-random binary matrix $\mathbf{D}$ of dimensions $q \times n$, and a public parity function P, which is a mapping $P : \xi \rightarrow \{0, 1\}$. When embedding, Alice replaces x_j with y_j such that $P(x_j) = 1 - P(y_j)$.

Let $b_i = P(x_i)$ be the parity of x_i, and $\mathbf{b} = \{b_i\}_{i=1}^{n}$ be the sequence of parities of all n elements from the image $\mathbf{x}$. All the bits b_i are known Alice, which communicates q bits $\mathbf{m} = (m_1, ..., m_q)^T$. Alice will modify some $b_j, j \in C$, so that the modified binary column vector $\mathbf{b}' = \{b'_i\}_{i=1}^{n}$, where $b'_i = P(y_i)$, satisfies

$$\mathbf{D}\mathbf{b}' = \mathbf{m}. \tag{2.8}$$

Thus, she needs to solve a system of linear equations in the Galois field of two elements (GF(2)).

Bob extract message bits from the bit-stream of parities of elements from the stego image $\{P(y_i)\}_{i=1}^{n}$. Bob first forms the vector $\mathbf{b}' = \{b'_i\}_{i=1}^{n}$ and then obtains the message $\mathbf{m} = \mathbf{D}\mathbf{b}'$ using the shared matrix $\mathbf{D}$.

Later, we will discuss the even more general solution in order to embed in a less strict way by using the Syndrome Trellis Codes (STC) [Filler *et al.*, 2010, 2011]. We kept the discussion of this solution based on what is called ρ cost map, in *Section* 2.3.2.

2.2 Overview of the earlier algorithms in JPEG

In this section, we discuss some of the most famous JPEG steganographic algorithms designed specifically for JPEG images, from the rise of steganography as a research discipline. The first and widely studied algorithms are (among others) Jsteg by [Upham, 1997b], JPHide&Seek (JPHS) 1998 by Allan Latham, Outguess by [Provos, 2001], F5 by [Westfeld, 2001] for embedding into a JPEG file. The embedding takes place in the quantized DCT coefficients.

In 1997, the first JPEG steganographic algorithm, Jsteg was created by [Upham, 1997b]. In its first version, the algorithm embeds a message using the LSB replacement technique with a sequential order for the message, i.e. without applying the notion of a stego key. Later, Jsteg had some improvements by [Upham, 1997a] with the JPEG-Jsteg algorithm where the message was encrypted and then sequentially embedded into the standard zigzag scanning order using the LSB replacement technique. The algorithm embeds message in the LSBs of the quantized DCT coefficients whose values are not in $\{0, 1, -1\}$.

[Chang *et al.*, 2002] improved the JPEG-Jsteg algorithm by increasing the size of the hidden message while maintaining the same security level. The method consists in embedding the secret message in the mid-frequency part of the quantized DCT coefficients instead of using the standard zigzag order, see *Figure* 2.5. Another

improvement for JPEG-Jsteg has been made to the JPEG-Jsteg-v4 algorithm by [Upham, 2008].

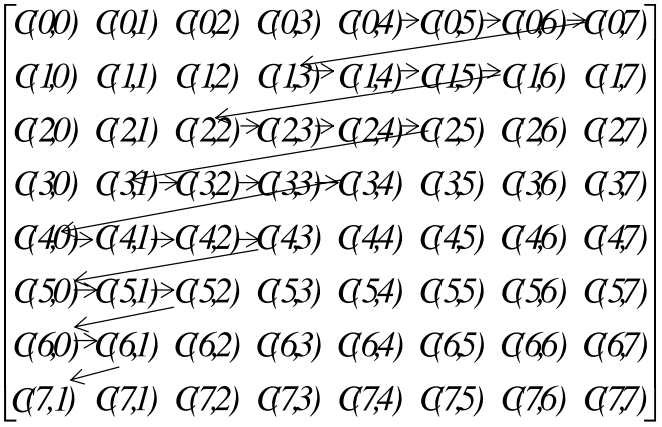

FIGURE 2.5: Embedding sequence from [Chang *et al.*, 2002].

In general, the improvement of steganographic schemes has consisted in bringing the stego image closer and closer to the original image. In other words, a perfect steganographic scheme must preserve all the statistical properties of the image, but it is complicated to obtain appropriate statistical models of the images.

One approach to improve the steganographic schemes is to embed messages to reduce the number of artefacts handled by Eve, *the steganalysis-aware schemes*. An example of this approach is the LSB Matching (*Section* 2.1.3) which has significantly increased the security of the LSB replacement (*Section* 2.1.2) approach since the histogram does not change significantly. The difference is visible in the examples of *Figure* 2.2 and *Figure* 2.3.

Schemes that approximately preserve the cover source model adopted by the steganographer are called *model-preserving schemes* [Hetzl and Mutzel, 2005]. One example is the Outguess steganographic scheme proposed by [Provos, 2001]. The main purpose was to preserve the χ^2 statistic of the embedded image so it can survive against χ^2 attacks. The algorithm is composed of two main embedding steps: first of all, Outguess splits all the blocks in two. In the first block, Outguess uses the LSB replacement technique to randomly embeds message bits into the LSBs of T matrices, where the values of $T \notin [0, 1]$. The positions are chosen using a pseudo-random number generator. Next, the algorithm makes statistic

restoration (correction) with the second set of blocks to T matrices to make the stego image histogram match the cover image histogram. An improvement of the Outguess algorithm was proposed by [Fard *et al.*, 2006]. The authors use a Genetic Algorithm based approach to optimise the embedding positions for a better histogram matching instead of using a pseudo-random number generator. Many model-preserving schemes have been proposed over years, like those by [Hetzl and Mutzel, 2005; Solanki *et al.*, 2005; Sallee, 2005; Solanki *et al.*, 2006; Sarkar *et al.*, 2007; Kodovský and Fridrich, 2008].

The improvements in steganographic schemes were for the *Minimal-impact steganographic schemes* which are the schemes that minimise the *embedding impact* [Fridrich and Filler, 2007], also called *distortion* D. The embedding impact is a probabilistic metric of how an image is statistically detectable after embedding. Given $\mathcal{C}$ and $\mathcal{S}$ the sets of covers and their corresponding stegos respectively, the distortion is described by

$$D : \mathcal{C} \times \mathcal{S} \rightarrow \mathbb{R}. \tag{2.9}$$

The improvement for the Minimal-impact steganographic schemes was with the incorporation of matrix coding into the embedding process, namely *the matrix embedding* (see *Section* 2.1.4). This technique was first introduced in the F5 steganographic scheme [Provos, 2001]. The author of F5 had previous works starting with the F2 embedding scheme, which was improved by F3, F4, and F5, where a significant matrix embedding-based improvement (Note that related to the F5 algorithm - nsF5 has been proposed to avoid shrinkage [Fridrich *et al.*, 2007]). In the F5 algorithm, firstly Alice uses a password-driven permutation ψ (the stego key) to shuffle all the coefficients followed by pseudo one time pad for a uniformly distributed message. Next, Alice uses the matrix encoding with minimal embedding rate, as explained in *Figure* 2.4, to embed the message in the permuted sequence. Next, Alice uses the inverse permutation ψ^{-1} process to get the original sequence of the embedded coefficients. Finally, the original sequence is delivered to the Huffman coder to continue the JPEG compression process. The whole process is illustrated in *Figure* 2.6.

This technique has opened the door to content-adaptive steganographic schemes, which are the leading schemes in the field. These schemes will be discussed in the following section.

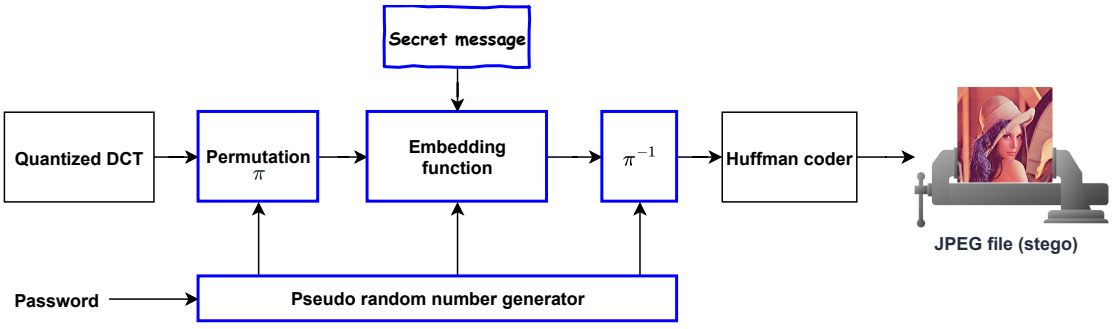

FIGURE 2.6: The F5 Embedding scheme [Westfeld, 2001].

2.3 Adaptive Steganography in JPEG

In adaptive steganography, Alice adapts the embedding process to embed the secret message in "secure" areas, i.e. areas where changes are more difficult to detect based on the quantized DCT statistics of the image. The idea of adaptive steganography dates to the early days of digital steganography [Franz *et al.*, 1996].

Generally, for the grey-scale images, Alice follows an embedding process starting with selecting the image quantized DCT values $\mathbf{x} = (x_1, ..., x_n) \in \{-1023, ..., 1024\}^n$ where to embed the secret message $\mathbf{m} = (m_1, ..., m_m) \in \{0, 1\}^{|m|}$, in order to produce the stego image with quantized DCT values $\mathbf{y} = (y_1, ..., y_n) \in \{-1023, ..., 1024\}^n$. Next, Alice selects the most secure areas by constructing a *cost map* ρ which models the embedding impact in terms of security for each quantized DCT coefficient. ρ will help Alice to embed the message in specific positions where it is difficult to detect the statistical changes. She selects the quantized DCT coefficients that have the weakest values of ρ, and which allows the coding and the embedding of the message.

2.3.1 Cost map ρ

The embedding impact can be modeled by a function that measures the distance between the cover image $\mathbf{x}$ and its corresponding stego image $\mathbf{y}$. *Equation* (2.9) can then be formally rewritten as:

$$\begin{aligned} D : \{-1023, ..., 1024\}^n \times \{-1023, ..., 1024\}^n \rightarrow \mathbb{R} \\ D(\mathbf{x},\mathbf{y}) = ||\mathfrak{f}(\mathbf{x}) - \mathfrak{f}(\mathbf{y})||, \end{aligned} \tag{2.10}$$

with $\mathfrak{f}$ a function that returns a vector of descriptors describing the image. These descriptors encode valuable information of the image, and act as a kind of digital "fingerprint" that can be used to distinguish one particular characteristic from the other.

This mathematical formulation of the distortion function (*Equation* (2.10)) is a major problem for the steganographer because it is non-additive and non-local. In order to overcome and simplify the problem, one approach is to make the hypothesis that the modification of a quantized DCT value does not affect the detectability of neighbouring quantized DCT values so that the distortion can be modified in an additive version. In this perspective, [Filler *et al.*, 2010] and [Filler *et al.*, 2011] propose an additive version of distortion that uses a cost map

$$\rho = \{\rho_i \in [0, \infty[\}_{i=1}^n. \tag{2.11}$$

The principle consists in assigning, for each image quantized DCT value x_i, a detectability cost $\rho_i \in [0, \infty[$. Its value models the impact of the modification of the i^{th} quantized DCT value of cover on the global security. These costs form the so-called **cost map**. Considering a binary embedding ($|x_i - y_i| \leq 1$), *Equation* (2.10) can be then rewritten:

$$D(\mathbf{x},\mathbf{y}) = \sum_{i=0}^{n} \rho_i \, |x_i - y_i|, \tag{2.12}$$

where $D(\mathbf{x},\mathbf{y})$ measures the embedding impact induced by the embedding process. In [Fridrich and Filler, 2007] the minimal expected distortion, for a *fixed payload* message $\mathbf{m}$ is presented in the following form:

$$min\ D(x,y) = \sum_{i=0}^{n} \rho_i\, \psi_i, \tag{2.13}$$

with ψ_i the probability of modification of the *ith* quantized DCT value, which is directly related to the detectability value ρ_i of the quantized DCT value at position i since ψ_i is defined as follows [Fridrich and Filler, 2007]:

$$\psi_i = \frac{e^{-\lambda\rho_i}}{1+e^{-\lambda\rho_i}}, \tag{2.14}$$

with $\lambda > 0$ a constant determined by the constraint on the message's size:

$$-\sum_{i=0}^{n}(\psi_i\ \log_2\psi_i + (1-\psi_i)\ \log_2(1-\psi_i)) = |\mathbf{m}|, \tag{2.15}$$

The modification probability map can be then defined as follows: $\mathbf{p} = \{\psi_i \in \mathbb{R}_+\}_{i=1}^{n}$, where quantized DCT values $\{x_i\}_{i=1}^{n}$ with higher modification probability $\{\psi_i\}_{i=1}^{n}$ have a higher chance of being modified.

As it stands, we can quickly conclude that the significant difficulty of this approach is the computation of the cost map $\rho = \{\rho_i \in [0,\infty[\}_i^n$. The computing of the costs ρ_i, which reflects the impact of the embedding process on security is still an open issue.

2.3.2 Syndrome Trellis codes

In *Section* 2.1.4 we gave a simple explanation of the linear error correction codes and the syndrome coding by an example based on Hamming codes, then we generalised the problem in *Equation* (2.7). We also discussed the wet paper codes as a solution to ensure a higher undetectability.

In what follows, we discuss other solution to generalize the problem of the *Equation* (2.7), by using the Syndrome Trellis Codes. We kept the discussion in this section since it is based on the ρ cost map.

STCs are an improved version of the syndrome coding, based on trellis coding. It provides a more subtle and adaptability to the cost map, not like wet paper codes. In its basic version, the STC code aims to preferentially modify the quantized DCT values of the cover JPEG image whose cost values are small. The STC coding allows a more precise control than wet paper codes as it is not limited to "yes" or "no" but to an indicator in the form of a real number.

Here, we give the codes description as described by [Filler *et al.*, 2010, 2011]. The so-called STC codes (Syndrome Trellis Codes) are codes whose decoding algorithm (Viterbi decoding) is based on a trellis structure. Let $\mathbf{m} = (m_1, ..., m_m) \in \{0,1\}^{|m|}$ be a secret message, we want to find the vector $\mathbf{v}_s$, such that during the reception, the syndrome:

$$\mathbf{z} = \mathbb{H}\mathbf{v}_s = \mathbf{m}, \tag{2.16}$$

where $\mathbf{v}_s$ represent the stego vector. In order to do this, the trellis approach uses a check-parity matrix $\mathbb{H}$ of particular shape; consisting of zeros and a binary sub-matrix $\hat{\mathbb{H}}$ of size $h \times w$ shared between the sender and the receiver. $\mathbb{H}$ is obtained by placing $\mathbf{m}$ copies of the sub-matrix $\hat{\mathbb{H}}$ next to each other and shifting each time by a line at the bottom. The rest of the matrix is set to 0. See *Figure* 2.7 for a matrix $\hat{\mathbb{H}}$ of size 22.

$$\hat{\mathbb{H}} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \mathbb{H} = \begin{pmatrix} 1 & 0 & & & & \\ 1 & 1 & 1 & 0 & & \\ & & 1 & 1 & 1 & 0 \\ & & & & 1 & 1 \\ & & & & & \ddots \end{pmatrix}$$

FIGURE 2.7: $\hat{\mathbb{H}}$ and $\mathbb{H}$ matrices

STC approach use graph structure to find $\mathbf{v}_s$ while minimising the distortion on the cover medium. The graph holds the corresponding vertices to the possible values for $\mathbf{v}_s$, while the edges hold the costs given in the cost map ρ. The graph is navigated from left to right during the embedding of the message, gradually exploring the relevant possibilities for the choice of the stego vector. The cost map controls the path to follow its end. When a path is created that reaches the end of the trellis, then a possible stego vector is found. Finally, the optimal solution is represented by the lowest cost path (the shortest path) that is identified by the Viterbi algorithm. Currently, the trellis approach is the most effective practical approach in terms of embedding efficiency; it is the closest method to the *theoretical bounds*[2]. Note that the difference between theoretical and practical bounds, in terms of distortion or payload, can be reduced by using the proposal of [Butora *et al.*, 2020]. The example in *Figure* 2.8 is a simplified illustration of the workings principle of the trellis approach (STC).

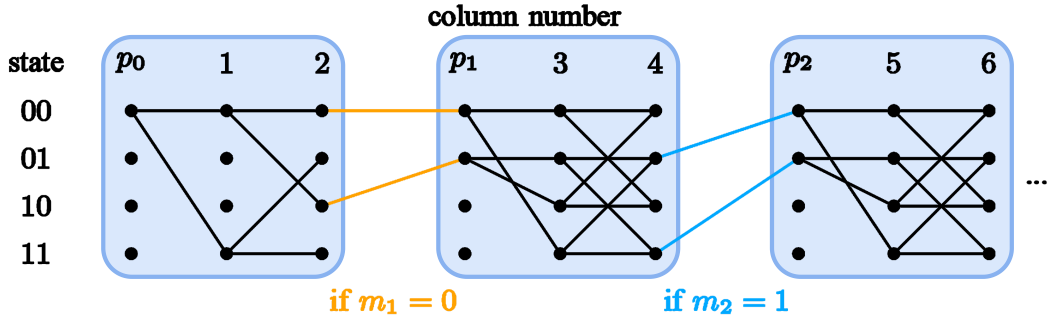

FIGURE 2.8: Illustration of the functioning principle of the trellis approach (STC).

Please note that these codes are used in almost all modern steganography algorithms. They are found in the schemes HUGO [Pevný *et al.*, 2010], S/J/SI-UNIWARD [Holub *et al.*, 2014], HILL [Li *et al.*, 2014], MVGG [Sedighi *et al.*, 2015], MiPOD [Sedighi *et al.*, 2016a], ASO [Kouider *et al.*, 2013] ...

[2]We refer the reader to [Fridrich, 2009] for more details about the theoretical bounds.

2.3.3 Embedding in an adaptive scheme

Message embedding is achieved by modifying the quantized DCT values $\mathbf{x}$ of the cover image. For this purpose, we proceed as follows:

1. $\mathbf{x} = (x_1, ..., x_n) \in \{-1023, ..., 1024\}^n$ is the cover quantized DCT image.
2. consider the least significant bits (LSB) of $\mathbf{x}$'s quantized DCT values to generate the so-called cover vector and noted as $LSB(\mathbf{x}) = \mathbf{v}_c = (v_{c_1}, ..., v_{c_n}) \in \{0, 1\}^n$.
3. use the secret key k (shared between both Alice and Bob) to shuffle $\mathbf{v}_c$ and the cost map ρ associated (as explained in sub-section. *Section* 2.3.1), $\mathbf{v}'_c$ and ρ' are then obtained.
4. use STC coding to generate the stego vector $\mathbf{v}_s = (v_{s_1}, ..., v_{s_n}) \in \{0, 1\}^n$ from $\mathbf{v}'_c$, ρ' and $\mathbf{m}$.
5. deshuffle $\mathbf{v}_s$.
6. embed the deshuffled $\mathbf{v}_s$ within the cover quantized DCT values $\mathbf{x}$, to obtain the stego image $\mathbf{y}$.

In step 6, the stego vector is embedded within the cover image by modifying some selected quantized DCT values. These selected quantized DCT values are modified by LSB matching. We illustrate this embedding process in an example of adaptive embedding in *Figure* 2.9.

2.3.4 JPEG adaptive schemes

In *Section* 2.3, we have explained the general process that every adaptive steganographic scheme follows. We summarised it in *Figure* 2.9. We also detailed the main steps; the cost map computation, STC encoding and finally, the embedding of the encoded message. In this subsection, we cite some of the current state-of-the-art steganographic algorithms based on an embedding scheme. We discuss

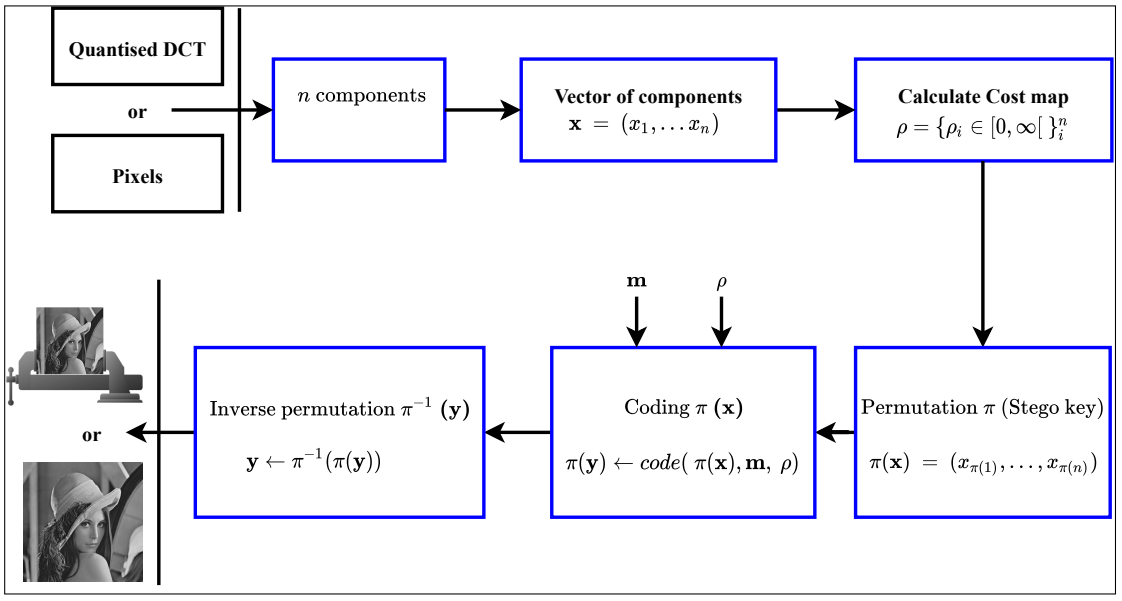

FIGURE 2.9: adaptive embedding process.

those used in the JPEG domain, highlighting how to calculate the cost map, and defining the distortion function of each. As for the encoding and embedding parts, it will not be discussed here as it remains the same as explained in the previous section.

As mentioned in *Section* 2.2, the first scheme that uses the matrix embedding was the F5. In this approach, Alice does not make any hypothesis on the detection probability for each cover quantized DCT value; all of them have the same probability of being detected when modifying the cover image. Hence, this approach is not adaptive and the cost map is constant:

$$\rho = \{\rho_i \in \{1\}\}_{i=1}^{n} \tag{2.17}$$

Furthermore, F5 had significant weakness; the shrinkage. It occurs each time Alice decreases the absolute value of 1 and -1 to 0. Bob cannot distinguish a zero coefficient that is initially in the quantized DCT values of the cover from a zero value produced by the shrinkage, so he skips all zero coefficients. Therefore, Alice repeatedly inserts the same bit until no shrinkage occurs. The shrinkage problem has been solved by an improved version of the F5 scheme proposed by [Fridrich *et al.*, 2007] and called the nsF5 (non-shrinkage F5). nsF5 consists of using the wet

paper technique with the cost map in *Equation* (2.18) instead of *Equation* (2.17) in order to avoid embedding in unwanted quantized DCT values. So,

$$\rho = \{\rho_i \in \{0, \infty\}\}_{i=1}^{n}. \tag{2.18}$$

Since the introduction of the STC coding, most embedding schemes use STC coding accompanied by a cost map, taking into account the content of the cover image. This is why they are called adaptive schemes. The first content adaptive scheme was proposed at the end of 2010, during the BOSS competition [Bas *et al.*, 2011a] using the HUGO algorithm [Pevný *et al.*, 2010]. The scheme was for spatial domain. Since then, many schemes has been proposed. Examples of image embedding in the spatial domain: EA [Luo *et al.*, 2010], ASO [Kouider *et al.*, 2013], MVG [Fridrich and Kodovský, 2013], S-UNIWARD [Holub *et al.*, 2014], HILL [Li *et al.*, 2014], MVGG [Sedighi *et al.*, 2015], Sync-HILL [Denemark and Fridrich, 2015], MiPOD [Sedighi *et al.*, 2016a]. For embedding into the JPEG domain: UED [Guo *et al.*, 2014], J-UNIWARD [Holub *et al.*, 2014], UERD [Guo *et al.*, 2015], IUERD [Pan *et al.*, 2016], HDS [Zichi Wang, 2016], RBV [Wei *et al.*, 2018] ...

2.3.5 J-UNIWARD a state of the art JPEG steganographic scheme

The J-UNIWARD scheme is among the most powerful JPEG embedding schemes. The cost map $\rho = \{\rho_i \in [0, \infty[\}_i^n$ is computed with a directional filter bank decomposition as a sum of relative changes between the cover $\mathbf{x} = (x_1, ..., x_n) \in \{-1023, ..., 1024\}^n$ and stego $\mathbf{y} = (y_1, ..., y_n) \in \{-1023, ..., 1024\}^n$, applied in a wavelet domain, where three filtering directions for wavelet decomposition are applied: horizontal, vertical, diagonal.

ρ is defined in such a way that for a quantized DCT coefficient i the cost of modifying it is:

$$\rho_i = \sum_{k=n}^{3} \sum_{u=1}^{n_1} \sum_{v=1}^{n_2} \frac{|W_{uv}^{(k)}(IDCT(\mathbf{x})) - W_{uv}^{(k)}(IDCT(\mathbf{x} \sim x_i))|}{\sigma + |W_{uv}^{(k)}(IDCT(\mathbf{x}))|}, \tag{2.19}$$

with σ a numerical stabilisation constant having an influence on the security of the scheme, $IDCT(.)$, the inverse DCT transform (see *Chapter* 1, *Equation* (1.3));

$W_{uv}^{(k)}(IDCT(\mathbf{x}))$ the wavelet coefficient at the position $(u, v) \in \{1, ..., n_1\} \times \{1, ..., n_2\}$ for the k^{th} sub-band of the $IDCT(\mathbf{x})$ image,

$W_{uv}^{(k)}(IDCT(\mathbf{x} \sim x_i))$ the wavelet coefficient at the position $(u, v) \in \{1, ..., n_1\} \times \{1, ..., n_2\}$ for the k^{th} sub-band of the image

$IDCT(\mathbf{x} \sim x_i)$ the inverse DCT of $\mathbf{x}$ whose coefficient i has been modified.

2.4 Conclusion

Among all the embedding schemes explained above, we chose the J-UNIWARD adaptive one, because it is one of the most powerful embedding schemes of the state of the art. Moreover, it has been used in studies similar to those we have been working on for this thesis.

Chapter 3

Conventional Steganalysis

Contents

Since the rise of digital image steganography, which consists of embedding a secret message in a cover image, and sending it to a peer, the concept of steganalysis has taken the form of a warden who attempts to analyze a cover image in order to detect the presence of this message or estimate its size [Cogranne *et al.*, 2020a]. This concept is called conventional steganalysis. In this chapter, we discuss this concept, the theory behind it, and the most studied techniques.

3.1 Overview of steganalysis

3.1.1 The Warden scenarios

In 1983, the steganography process was formally modelled as the prisoner's problem [Simmons, 1983]. The problem is that two prisoners, Alice and Bob, are imprisoned in isolation cells and want to discuss an escape plan, through letter exchange, without being discovered by the warden Eve. Any suspicion Eve may have about this secret communication will lead to actions to stop the operation between Alice and Bob. The act of communicating secretly, for example through harmless letter, is named steganography. *Figure* 3.1 illustrates the general model of the Simmons prisoner problem.

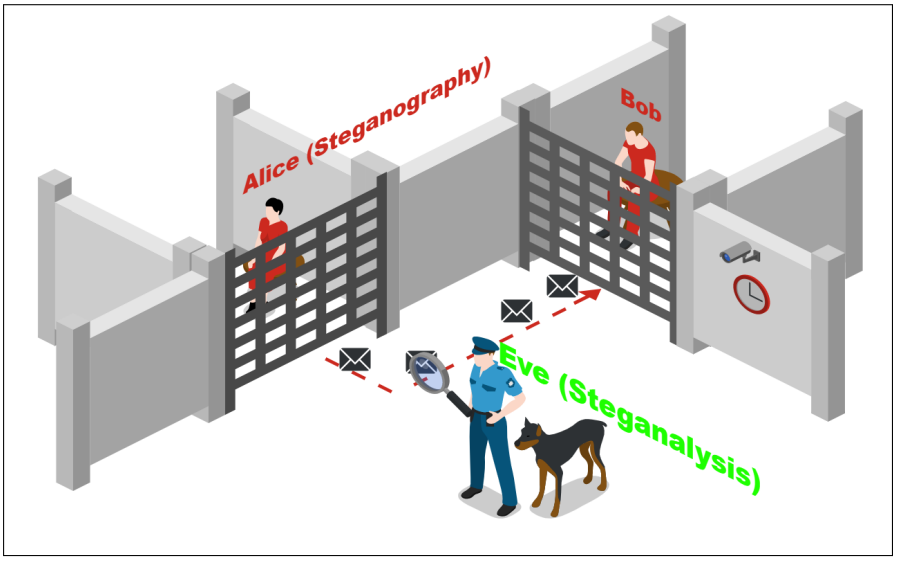

FIGURE 3.1: The general model of Simmons' "prisoner problem".

Warden Eve's job is steganalysis. She has three choices to make when she examines the communication between Alice and Bob. The choices are based on the abilities and actions she can take when she recognizes a secret communication. These choices are described as scenarios. There are three scenarios for a warden: the active warden scenario, the passive warden scenario and the malicious warden scenario, as shown in *Figure* 3.2, *Figure* 3.3 and *Figure* 3.4:

- **The passive scenario:** In this scenario, Eve performs a *binary steganalysis*. She steganalyses the messages, as shown in *Figure* 3.2. If she detects a secret message, she blocks it [Anderson and Petitcolas, 1998]. Otherwise, the message will be delivered. Eve does not interfere in the communication channel between Alice and Bob, but she can try to estimate the message length by *quantitative steganalysis*.

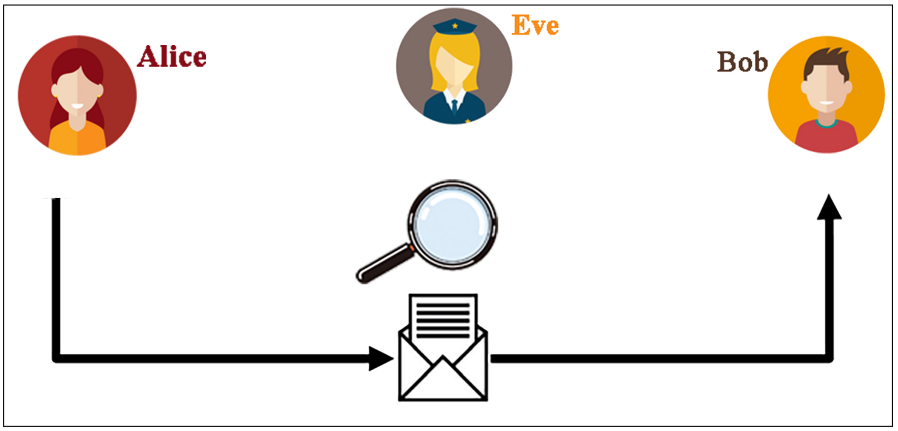

FIGURE 3.2: Passive scenario.

- **The active scenario:** For this situation, Eve can consciously alter the messages sent by Alice to Bob or by Bob to Alice, as appeared in *Figure* 3.3, to ensure that a possible hidden message will be destroyed [Ettinger, 1998; Cachin, 1998].

- **The malicious scenario:** Here, Eve's role goes beyond discovering the existence of the hidden message. It is about trying to interfere with the communication between Alice and Bob. Eve can also extract the content of the hidden message by trying to discover the stego-key, so that she can

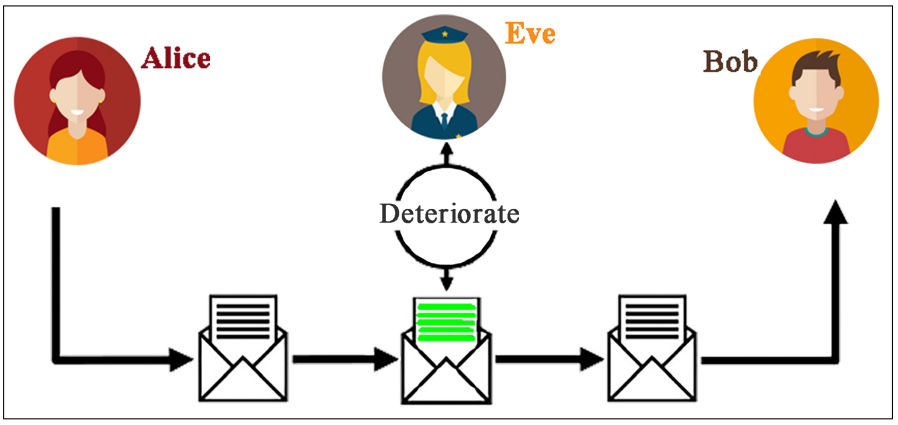

FIGURE 3.3: Active scenario

imitate Alice, change the content of the message and trap Bob [Francia and Gomez, 2006] as shown in *Figure* 3.4.

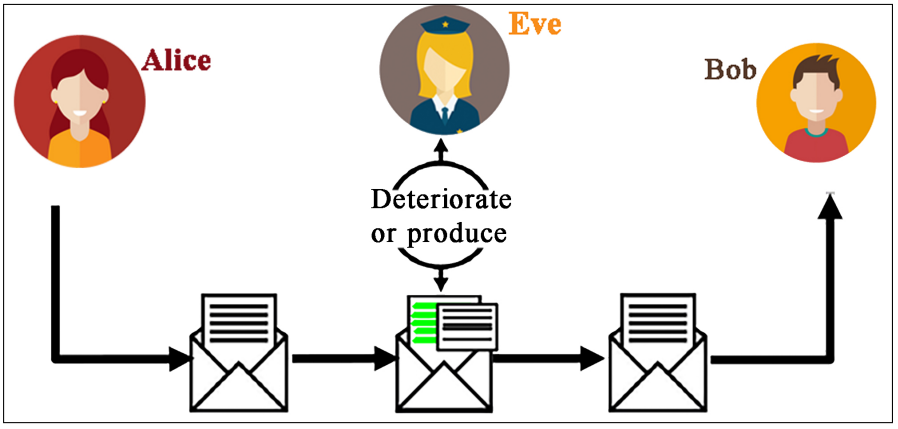

FIGURE 3.4: Malicious scenario

In this thesis, we focus on the passive scenario.

3.1.2 Binary and quantitative steganalysis

Conventional steganalysis approach can be classified into two scenarios: *Binary* and *Quantitative*. Given an image $\mathbf{x}$, the purpose of binary steganalysis is to decide whether $\mathbf{x}$ is a cover or a stego image. A binary steganalysis algorithm is also called a *binary detector*.

Quantitative steganalysis aims to estimate the payload size. A zero payload corresponds to the cover image. A quantitative steganalysis algorithm is also called

a *quantitative detector*. The problem of payload estimation can be formulated as follows. Given a quantitative detector f, f is a real value mapping,

$$f : \mathcal{X} \to \mathcal{P} \subseteq \mathbb{R}, \tag{3.1}$$

where $\mathcal{X}$ is a set of the cover images and $\mathcal{P}$ is a subset that represents the space of the payload size.

In the modern approaches, the binary steganalysis is considered as a binary classification problem, while the quantitative steganalysis is considered as a regression problem.

3.1.3 Clairvoyant hypotheses

Generally, a steganalysis process is called clairvoyant when Eve has a perfect knowledge of the steganographic channel.

The different assumptions for clairvoyant steganalysis are that Eve knows: the embedding scheme, the test distribution, the image relative payload α, and the size of the images.

Let us recall the definition of the steganographic channel explained in *Section* 2.1.1 where the functions $Emb : \mathcal{X} \times \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{X}$ and $Ext : \mathcal{X} \times \mathcal{K} \longrightarrow \mathcal{M}$ define the embedding and extraction operations, given a set of cover images $\mathcal{X}$, a set of keys $\mathcal{K}$ associated with the cover images, and a set of messages $\mathcal{M}$ that Alice can embed.

Given an object $\mathbf{x} \in \mathcal{X}$ to be examined, Eve assumes a distribution of cover images P_c, messages P_m, and keys P_k. These distributions together with a chosen steganographic scheme defines a distribution of stego images P_s. In the *clairvoyant steganalysis*, Eve has knowledge about P_c and P_k and consequently she has knowledge about P_s. The binary steganalysis algorithm is then equivalent to have the following test with two hypotheses.

$$\begin{aligned} H_0 &: \mathbf{x} \sim P_c, \\ H_1 &: \mathbf{x} \sim P_s. \end{aligned} \tag{3.2}$$

3.1.4 Evaluation measures

3.1.4.1 Binary steganalysis evaluation

We mentioned before that the binary steganalysis is considered as a binary classification problem. To assess the results, we can use the Receiver Operating Characteristic curve (ROC curve), which is a graphical plot where each point represents the true positive rate in function of the false positive rate for a given classification threshold. The ROC curves can be used to compare curves, and eventually to find a threshold.

Let us consider a binary classification problem, in which the results are labelled either positive (**stego**) or negative (**cover**). There are four possible outcomes to a binary classifier in steganalysis:

1. **True Positive (TP):** if the result of a prediction is **stego** and the actual class is also **stego**.
2. **False Positive (FP):** if the result of a prediction is **stego** and the actual class is **cover**.
3. **True Negative (TN):** if the result of the prediction and the actual class are both **cover**
4. **False Negative (FN):** if the result of the prediction is **cover** while the actual class is **stego**.

The confusion matrix in *Figure* 3.5, illustrate the TP, FP, TN and FN cases. We use the following formulas to calculate the probability (rate) of each one:

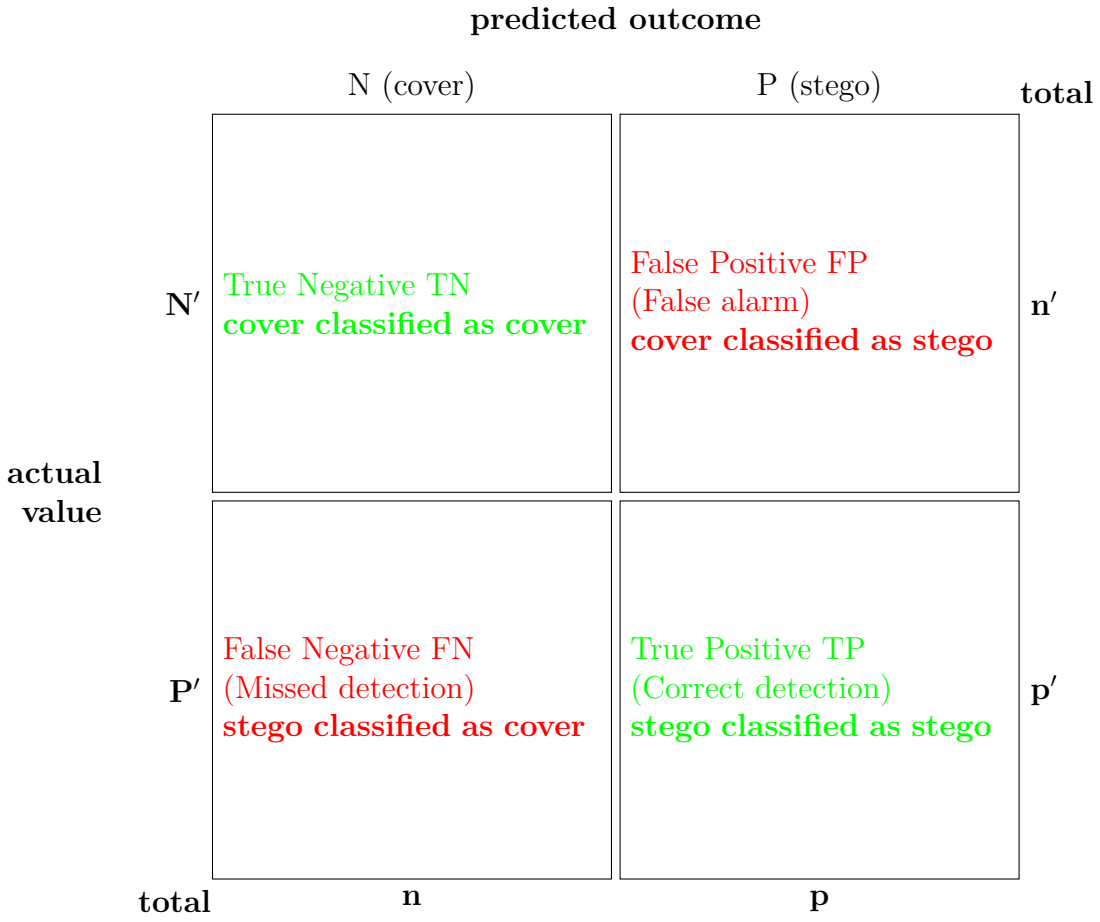

FIGURE 3.5: Confusion Matrix, applied to binary steganalysis.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR} \tag{3.3}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR} \tag{3.4}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = 1 - \mathrm{TNR} \tag{3.5}$$

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR} \tag{3.6}$$

Generally, in the steganalysis domain, different names for the above rates are used. The TPR is called the probability of correct detection P_{cd} or power function. FPR

is called the probability of false alarm P_{fa}, FNR is called the probability of missed detection P_{md}.

Steganalysis algorithms with ROC curves close to the diagonal are not accurate. In contrast, a steganalysis algorithm whom ROC curve has a large area under the curve (AUC) is accurate, see *Figure* 3.6. In this thesis, we use the algorithm explained in *Appendix* B to construct ROC curves.

Note that in *Figure* 3.6, the comparison between the two ROC curves is difficult, as their AUCs are quasi-equal. A solution for this problem has been proposed in the **ALASKA2** steganalysis challenge [1]. This solution, called weighted AUC, consists of dividing the true positive rates into three parts and weighting each part according to its position. The area between the true positive rate of 0 and 0.4 is weighted 2×, the area between 0.4 and 1 is now weighted 1×. The total area is normalized by summing the weights so that the final weighted AUC is between 0 and 1. It can be seen from *Figure* 3.7 how the weighted AUC is able to distinguish the best ROC curve from those that are difficult to compare using the AUC.

In the following, we cite the different evaluation measures used in binary steganalysis:

- **The probability of error** P_e: Researchers proposed different versions of the P_e evaluation measure, they differ by the choice of a fixed values of the probability of false alarm P_{fa} or the probability of missed detection P_{md}. A frequently used measure is the minimum average classification error under equal prior probabilities [Filler *et al.*, 2011]. The minimum is reached at a point where the tangent to the ROC curve has slope $\frac{1}{2}$, [Fridrich, 2009]:

$$P_e = \min_{P_{fa} \in [0,1]} \frac{1}{2}(P_{fa} + P_{md}(P_{fa})). \tag{3.7}$$

- **The FP-50 evaluation measure**: [Pevnỳ and Ker, 2015] proposed the FP-50 evaluation measure, which is the False positive rate at 50% detection

[1] https://www.kaggle.com/c/alaska2-image-steganalysis/overview/evaluation

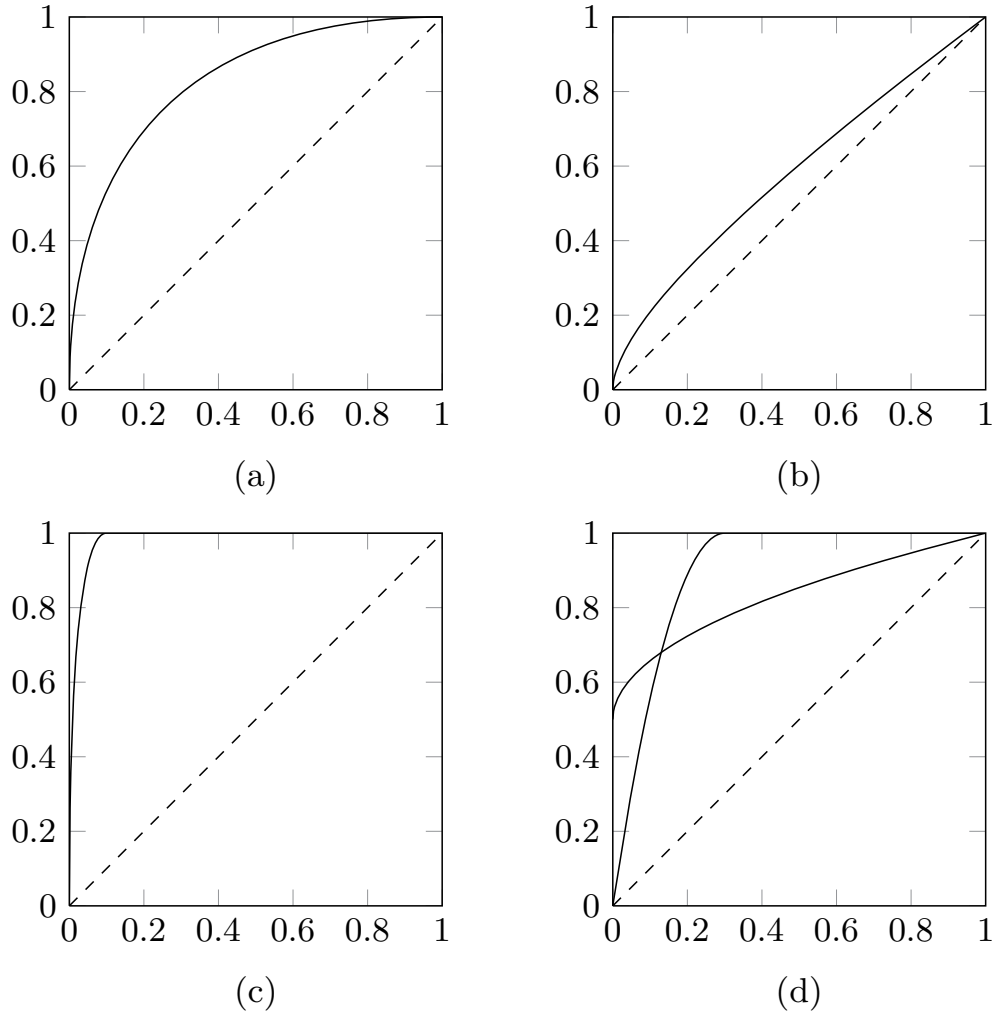

FIGURE 3.6: Examples of ROC curves. The x and y axes in all four graphs are the probability of false alarms, P_{fa}, and the probability of detection, P_{cd}, respectively. (a) Example of a ROC curve; (b) ROC of a poor detector; (c) ROC of a very good detector; (d) two hard-to-compare ROCs. [Fridrich, 2009]

accuracy. This measure offers interesting statistical properties for centered and symmetrical distributions of cover scores [Cogranne *et al.*, 2019].

- **The "Accuracy at the Top" evaluation measure**: The "Accuracy at the Top" measure gives the accuracy detecting one stego content (or actor) among for example the 1% most suspicious contents/actors. This measure is proposed by [Ker and Pevný, 2012a] to measure how often a true guilty actor appears in the top n (or the top x%) of a list presenting the most suspicious actors.

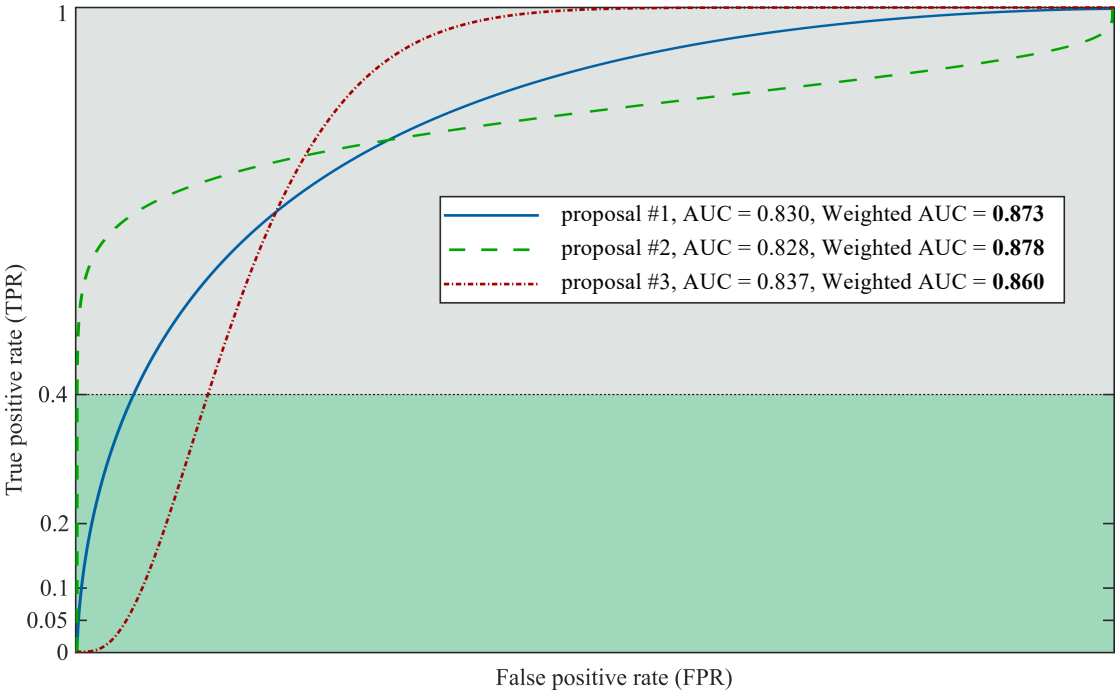

FIGURE 3.7: AUCs and weighted AUCs, for three different ROC curves. Figure from the **ALASKA2** steganalysis challenge.

- **The MD5 evaluation measure**: This measure is proposed in the ALASKA challenge [Cogranne *et al.*, 2019]. MD5 is the missed detection rate for a false positive rate of 5% (abbreviated MD5 for Miss Detection at 5% false alarm rate). The authors inspired the MD5 by the FP-50 and the the "Accuracy at the Top" evaluation measures. This measure has also been used in [Cogranne and Fridrich, 2015] (without being named MD5 measure), P_e is computed based on the Neyman-Pearson bi-criteria approach, which aims to minimize the probability of missed detection for a prescribed false-alarm probability P_{fa}, the authors fixed P_{fa} to 5%.

3.1.4.2 Quantitative steganalysis evaluation

Given an image $\mathbf{x}$ to be examined by a quantitative detector, the actual payload p of $\mathbf{x}$ and its predicted value $\hat{p}$, we denote by $e = p - \hat{p}$ the error prediction for payload size estimation of the image $\mathbf{x}$. To evaluate the overall detector on n images, we use one or more of the following measures:

- The Mean Squared Error MSE:

$$MSE = \frac{1}{n}\sum_{t=1}^{n} e_t^2.$$

- The Root Mean Squared Error RMSE:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}.$$

- The Mean Absolute Error MAE:

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |e_t|.$$

3.1.5 Earlier steganalysis approaches

Before going deeply into the state of the art steganalysis techniques, we discuss some of the earlier ones. The era of steganalysis, before introducing machine learning techniques, was based on analytic approaches.

Statistical Attacks attempts to compare the theoretically expected frequency distribution for the stego image with a sample distribution observed in the cover image that may have been embedded with data. This is the case of the χ^2 attack [Westfeld and Pfitzmann, 1999]. In this article, the statistical attack is applied to the EzStego and Jsteg embedding schemes. The embedding procedure continuously overwrites the least significant bits of the ordered indices, which transforms the values into each other that only differ in the least significant bits, causing the appearance of equal *pairs of values*, called PoVs by the author, see *Figure* 3.8.

The dotted lines in *Figure* 3.8 show the frequency distribution for an image before and after embedding, and they are similar because embedding does not affect them (for a stego image, the frequency is the arithmetic mean of the two frequencies in a PoV). This leads to obtain the theoretically expected frequency distribution from the random sample. The only thing to do to classify the stego from the cover

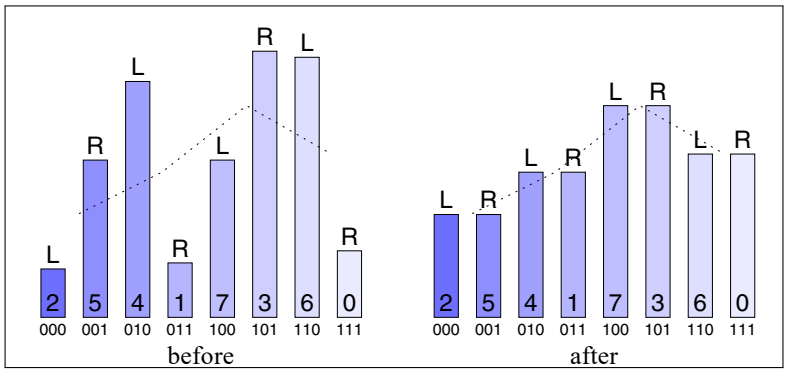

FIGURE 3.8: Histogram of colours before and after embedding a message with EzStego [Westfeld and Pfitzmann, 1999].

images is to apply the χ^2 test to calculate the degree of similarity between the observed sample distribution and the theoretically expected frequency distribution.

The *targeted attacks*, exploit the process used by a scheme to attack it, such as the Jsteg attack by [Wu *et al.*, 2005], LSB Matching attack by [Huang *et al.*, 2007], OutGuess attack by [Fridrich *et al.*, 2002a], F5 attacks by [Fridrich *et al.*, 2002b, 2007]. These targeted attacks have a high-performance rate for a specific scheme, but they are not efficient or useless on other embedding schemes.

We give an example, the F5 attack in [Fridrich *et al.*, 2002b]. F5 cannot be attacked by a simple χ^2 attack because instead of replacing the LSBs of the quantized DCT coefficients by the bits of the message or swapping pairs of fixed values, F5 decreases the absolute value of the coefficient by one. In this F5 attack, the authors therefore use a different way of attacking the algorithm, which consists in taking advantage of the fact that the histogram of the DCT coefficients modified by the F5 algorithm preserves certain essential characteristics of the histogram, such as its monotony and the monotony of the increments [Westfeld, 2001]. The attack is thus constructed by analyzing how F5 modifies the histogram values in order to find distinctive statistical quantities that correlate with the number of modified coefficients. Then the basic values of these statistical quantities are determined. This leads the attacker to estimate the histogram of the cover image from a stego, as shown in *Figure* 3.9 and hence to detect the presence of F5 embedding.

Normally, LSB-based embedding schemes would leave a features structure in the

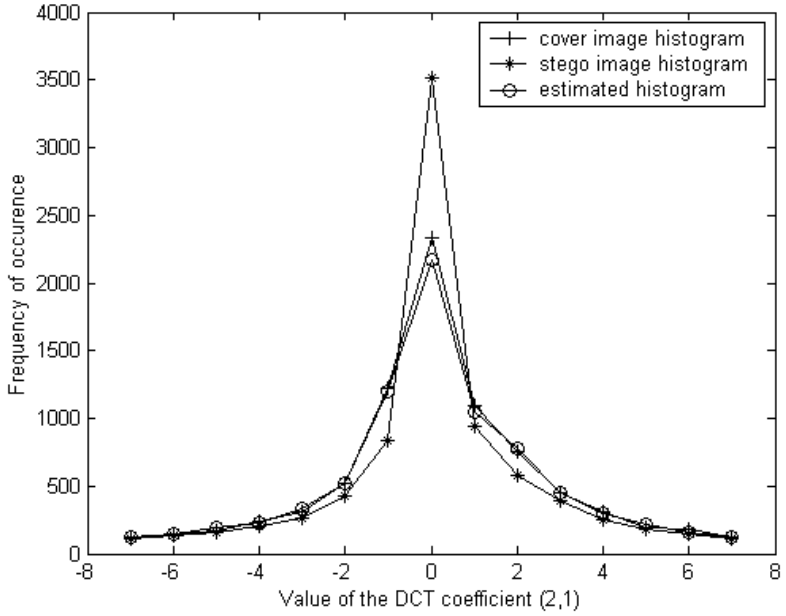

FIGURE 3.9: A comparison of the histogram estimate to the histogram of the original image. The graph shows the original histogram values (crosses), histogram values after applying the F5 algorithm with maximum capacity (maximal possible message), or payload = 0.5 bpnzAC (stars), and the estimate of the original histogram (circles) [Westfeld, 2001].

cover image while embedding the secret message. *Structural attacks* exploit this structure to attempt detecting the presence of a hidden message or predicting the message length by targeting the LSB replacements in an image. The *structural attacks* such as RS analysis [Fridrich *et al.*, 2001], Sample Pairs Analysis (SPA) [Dumitrescu *et al.*, 2002], Triples analysis [Ker, 2005], and the Weighted Stego detector (WS) [Fridrich and Goljan, 2004; Ker and Böhme, 2008; Cogranne *et al.*, 2011; Zitzmann *et al.*, 2011] are considered the first *quantitative detectors*.

3.2 Binary Steganalysis by Machine Learning

Machine learning approaches are also called *two-step learning*. The first step is the data preprocessing and feature extraction which is isolated to the next step, the classifier. A feature is a quantifiable single property or characteristic of a studied steganographic scheme. The choice of valuable features is a crucial step for modeling an image in an efficient way. A set of numerical features is represented by a feature vector.

Generally, the binary classifiers perform a two-phase process, the *training phase* and the *testing phase*. In the training phase, the classifier uses a set of images to train and validate a steganalysis algorithm. Another different set of images is used by the classifier to test and evaluate the trained steganalysis algorithm (see *Figure* 3.10). Without any prior on the class preparation in test, 50% of the feature set is chosen from stego images, and 50% from cover images, in the training and testing phases. Otherwise, the results will be biased in favor of the class with the higher ratio, as it will gain more information than the other.

Furthermore, it is important to ensure that the pairs of cover characteristics and the corresponding stego characteristics are contained in the training set. This specific steganalysis modification is essential because it has been shown that separating the stego-cover pairs into two sets, one used for training and the other for testing, one for error estimation, can lead to a biased estimation error and loss in performance [Schwamberger and Franz, 2010; Kodovskỳ, 2011].

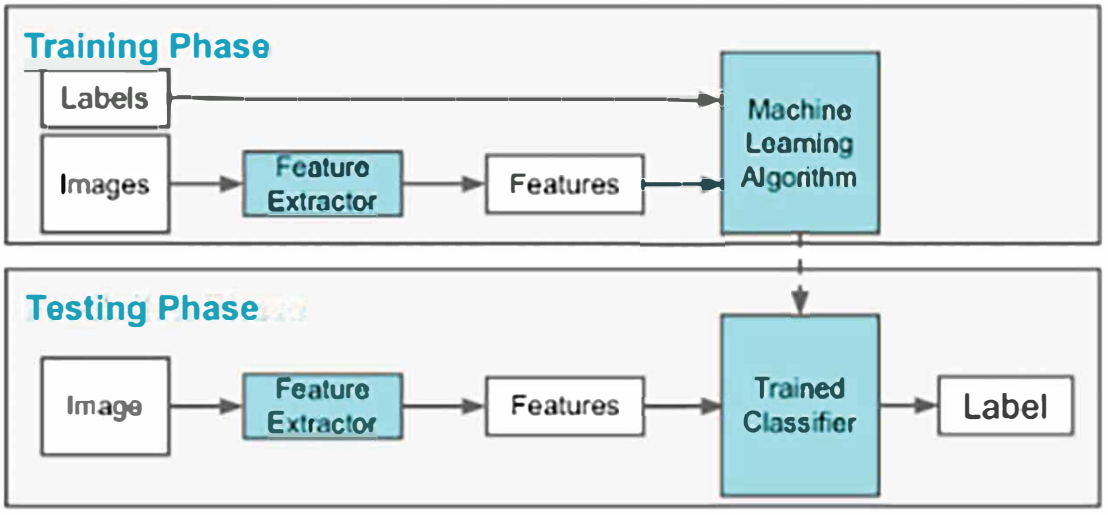

FIGURE 3.10: The two machine learning phases.

The evaluation of the binary detectors is done using the ROC curves and the Pe measure (See *Section* 3.1.4.2).

3.2.1 Binary detectors using Rich Models

One of the first feature based steganalysis was proposed by [Farid, 2002], the author used the Fisher Linear Discriminant (FLD) [Fisher, 1936] to classify a 72-dimensional feature vectors.

This approach has been remarkably improved in recent years with the introduction of Rich Models (RM) for JPEGs (see *Table* 3.1) and Ensemble Classifier (EC) [Kodovský *et al.*, 2012; Cogranne *et al.*, 2015; Cogranne and Fridrich, 2015].

Rich Model	Dimension	reference
CC-C300	48,600	[Kodovský and Fridrich, 2011]
CF*	7,850	[Kodovský *et al.*, 2012]
CC-JRM**	22,510	[Kodovský and Fridrich, 2012]
DCTR	8,000	[Holub and Fridrich, 2015a]
PHARM	12,600	[Holub and Fridrich, 2015b]
GFR	17,000	[Song *et al.*, 2015]
GFR-GW$_6$	17,820	[Xia *et al.*, 2017]
DCTRD	8,751	[Xia *et al.*, 2019]
GFRD	25,448	

TABLE 3.1: A list of high dimensional rich models for JPEG images, and the dimension of each one.

The notion of rich models is attributed to high-dimensional feature vectors constructed to gather more valuable information for steganalysis. One of the weaknesses of this approach was the computational limitations of the available classifiers that are able to handle this huge number of features. Ensemble classifiers solved this limitation, as they independently process each subset of features of size d_{sub} using a weak classifier, and then combine the results from each of them to make the final decision. This approach has far exceeded, in terms of accuracy, the previous approach based on small size features and SVM classifiers.

In this section we discuss recent approaches in two-step learning, rich models and ensemble classifiers. We focus on the JPEG domain since this is the interest of this thesis. We start by presenting the Gabor features residual GFR [Song *et al.*, 2015], a JPEG-rich model, which will be used in all our experiments.

3.2.1.1 Gabor features residuals GFR

Gabor Feature Residuals or GFR, proposed in [Song *et al.*, 2015], is a feature extraction method for steganalysis, based on 2D Gabor filters which can describe the texture components of the image at different scales and orientations (see *Appendix* C for more details on Gabor filters). These texture elements are difficult to preserve by adaptive embedding schemes, so they can be used for steganalysis purposes.

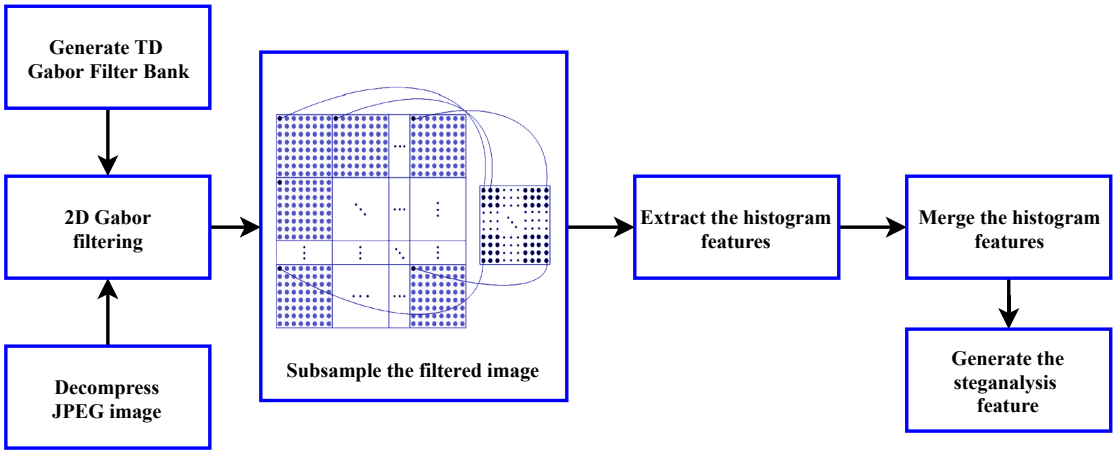

FIGURE 3.11: GFR feature extraction process [Song *et al.*, 2015]

The feature extraction method is shown in *Figure* 3.11 and the detailed extraction procedures are described as follows: First of all, the JPEG image is decompressed into the spatial domain without quantifying the pixel values at $\{0, 1, ..., 255\}$ (i.e. keeping the real value of pixels) to avoid any loss of information. Then, the 2D Gabor filter bank with different scales and orientations is generated. Next, the decompressed JPEG image is convoluted with each 8×8 2D Gabor filter. Next, for the filtered image generated by the 2D Gabor filters with the same scale parameter, the corresponding histogram features are merged according to the symmetric orientations. Finally, all the features of the merged histogram are combined to form the final steganalysis feature.

This approach has been improved, first in [Xia *et al.*, 2017] (GFR-GW$_6$) by merging the histograms according to the symmetries between the different Gabor filters,

which allows for further compacting the features while improving detection accuracy. In addition, by placing a Gaussian on each of the residual samples and using integrals over the quantization intervals, a weighted histogram is obtained that is more sensitive to small variations in residuals [2] than the original GFR histogram.

A second improvement for GFR has been proposed by [Xia *et al.*, 2019]. The authors revisit the GFR approach. From a de-quantized and non-rounded JPEG image, the authors propose to apply Gabor filters. The resulting residual filtered images are used to compute a difference of two residual filtered images and then compute co-occurrences. The resulting features are called GFRD. Experiments show more improvement over the GFR approach, in terms of presentation of properties that allow hidden information to be discovered.

3.2.1.2 Ensemble classifiers and the GLRT algorithm

Ensemble classifiers have become an alternative to the SVM approaches for binary classification, which have become useless with the rise of rich models. For example, there is 17,000 entities for the rich GFR model. SVM classifiers do not converge when trained on such a rich model due to its high dimension. The low complexity of the ensemble classifiers has made it possible to train a very large number of features, which has led to a significant improvement in the detection of modern steganographic schemes.

[Cogranne and Fridrich, 2015] proposed a rich-model based Ensemble classifier, the algorithm is called the GLRT-Ensemble classifier. GLRT stands for the generalised likelihood ratio test; we call it GLRT in the following.

This algorithm takes advantage of the strengths of the optimal detectors and steganalysis machine learning approaches to use an accurate statistical model for base learner projections in an Ensemble classifier [Kodovský *et al.*, 2012]. Each base learner is a Fisher Linear Discriminant (FLD) classifier [Duda, 2001] formed on a subset of uniformly randomly selected characteristics, and then its projection

[2] In steganography, the notion of residuals is referring the differences between cover images and stego images.

$\mathbf{v}$ is cast into hypothesis test theory. The statistical hypothesis test here is a mapping

$$\delta : \mathbb{R}^L \mapsto \{H_0, H_1\},$$

so that the H_i hypothesis is accepted if $\delta(\mathbf{v}) = H_i$. Note that this detector works without any assumptions about the size of the payload. This criterion applies here by adopting the "shift hypothesis" which was first recognised by [Ker, 2006], see *Appendix* A for more details.

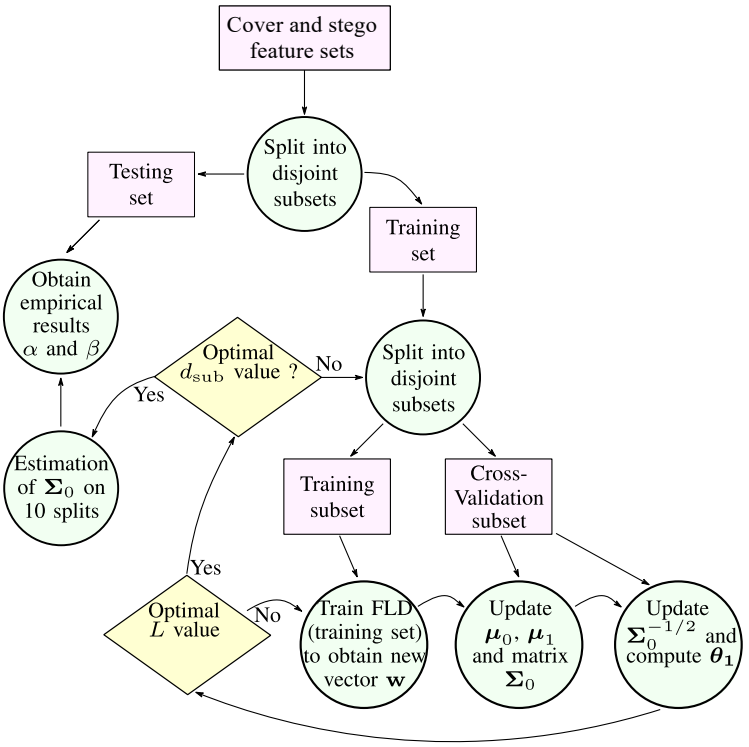

FIGURE 3.12: Diagram from the original article [Cogranne and Fridrich, 2015], showing the implementation of the GLRT ensemble. Note that the rectangles represent the data, the circles are associated with the operations and the diamonds represent the loops and associated tests.

GLRT is an improvement of the original Ensemble classifier [Kodovský *et al.*, 2012], the first binary detector based on Ensemble, but with some technical differences. In the original classifier, each base learner is trained on a bootstrap sample of the training set (also known as the bootstrap aggregation technique) while GLRT does not use the bagging technique but divide the training set into two equally sized subgroups. Besides, GLRT does not use the out-of-bag detection error (OOB) estimate used in the original classifier. Also, the majority voting technique used

by the original classifier is replaced by a likelihood ratio test in the GLRT, which gives it more control over the probabilities P_{fa} and P_{cd}. The most important change for us concerns the ability of the GLRT to detect messages of unknown length, as it makes no assumptions about the possible payload embedded in the image, which the original classifier does not have.

In the following, we briefly explain the implementation of the GLRT ensemble classifier which is described in *Figure* 3.12. We refer the reader to the original article for more details.

First, the features are divided into two subsets of equal size for training and testing. Then, each base learner is trained on only half of the training set, while the other half of the training set is used for cross-validation to validate the performance of the base learners. Next, the optimal values of the number of features used by each base learner d_{sub} and the number of base learners L are calculated. Next, the base learners' projection covariance Σ_0, the expectations μ_0 under hypothesis H_0 and μ_1 under hypothesis H_1 are measured. For a fixed value of d_{sub}, each base learner is added to the ensemble as follows (see the bottom row of *Figure* 3.12):

1. a randomly selected subset of features of size d_{sub} is selected (not shown in *Figure* 3.12).

2. the training set is used to obtain the new projection vector $\mathbf{w}$ (the linear subspace normal of the FLD decision boundary) from an FLD base learner.

3. the cross-validation set is used, with the projection vector $\mathbf{w}$, to update the base learners' projection covariance Σ_0 and expectations μ_0 and μ_1. Once these values have been updated, it is straightforward to re-compute $||\theta_1||$ using equation $\theta_1 = \Sigma_0^{-1/2}(\mu_1 - \mu_0)$.

The goal of this training procedure is to calculate the values of the projection vector $\mathbf{w}$ and the value of θ_1 in order to minimize the classification probability of error P_e in *Equation* (3.7) using the threshold $\tau^{p_e} = \frac{||\theta_1||}{2}$.

Classifying an image is done by projecting each subspace of the feature vector onto projection vector $\mathbf{w}$. This gives a vector $\mathbf{v}$ which is normalized to a vector

$\tilde{\mathbf{v}} = \Sigma_0^{-1/2}(\mathbf{v} - \mu_0)$. Next, the GLR test $\Lambda^{lr}(\tilde{\mathbf{v}})$ is computed which is simply given by the projection onto the vector of the mean projections under H_1, $\Lambda^{lr}(\tilde{\mathbf{v}}) = \frac{\theta_1^T \tilde{\mathbf{v}}}{||\theta_1||}$. Finally, the GLR value is thresholded using the τ^{p_e}.

3.2.2 Binary Steganalysis by Deep Learning

Since 2015, steganalysis with rich models and ensemble classifiers has been competing with deep learning approaches (one-step learning) such as, for the JPEG domain, ReST-Net [Li *et al.*, 2018], Xu-Net-JPEG [Xu, 2017], SRNet [Boroumand *et al.*, 2019] and Low-Complexity-Net [Huang *et al.*, 2019].

The one-step learning approach differs from the two-step learning approach with respect to data preprocessing and feature extraction operations. In two-step learning, these operations are performed manually, before the classification step, whereas in one-step learning, these operations are performed automatically by the classification architecture. The two learning approaches are illustrated in *Figure* 3.13.

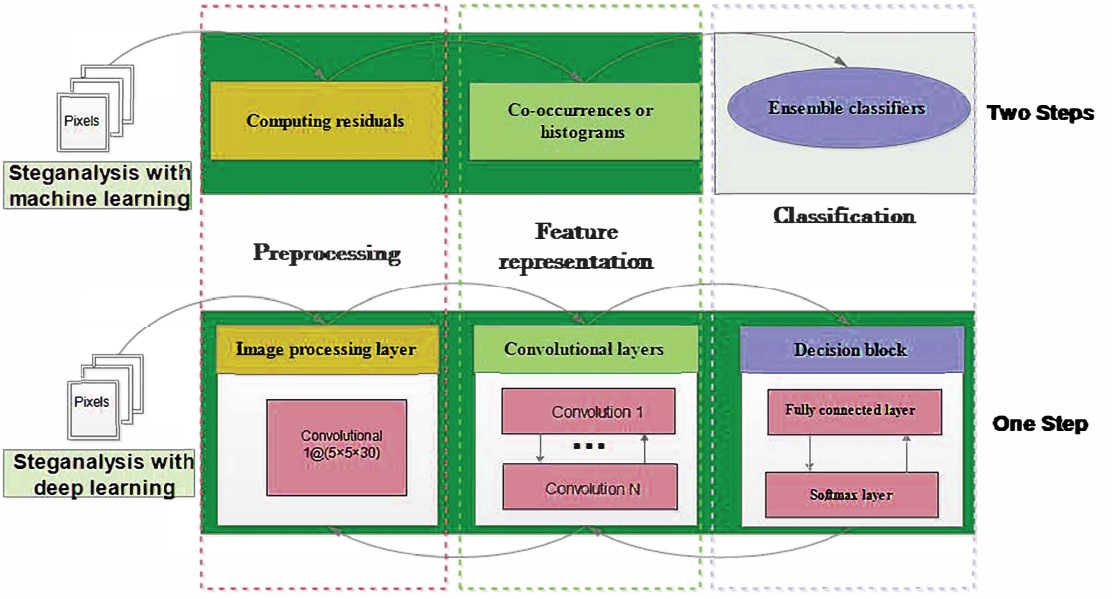

FIGURE 3.13: two steps learning VS one step learning.

In *Section* 3.2.1.2, we discussed how the ensemble classifiers together with the rich models, replaced the SVM classifiers. Since 2015, the CNN-based classifiers

started to replace the ensemble classifiers [Chaumont, 2020]. In this section, we briefly discuss the recent techniques for binary steganalysis with deep learning in JPEG.

The first approach to the application of deep learning techniques for steganalysis dates back to 2014 [Tan and Li, 2014] with auto-encoders. A year later, [Qian *et al.*, 2015] and [Pibre *et al.*, 2016] suggested using convolutional neural networks (CNN). The results remained well below the state of the art until the proposal of [Xu *et al.*, 2016], the results achieved with an ensemble of CNNs were nearly as good as the state of the art.

[Zeng *et al.*, 2017] proposed the first JPEG CNN classifier, the same author published another one in [Zeng *et al.*, 2018], with a proposal of a pre-processing technique inspired by the Rich Models, and the use of a big learning database. The results were close to those of the state-of-the-art. JPEG compression process was the inspiration of another technique, the *phase-split*, used to build the VNet CNN by [Chen *et al.*, 2017].

It took a set of CNNs to obtain slightly better results than state of the art. In Xu-Net-Jpeg [Xu, 2017], a CNN inspired by ResNet [He *et al.*, 2016] with the *shortcut connection* trick, and 20 layers also improve the accuracy results. Note that in 2018 the ResDet [Huang *et al.*, 2018] proposed a CNN different than Xu-Net-Jpeg [Xu, 2017] but with similar results. At the end of 2018, SRNet [Boroumand *et al.*, 2019] has been proposed, which is a network that can be adapted to spatial or JPEG steganalysis. It requires various tricks such as virtual augmentation and transfer learning, and therefore requires a bigger database compared to Yedroudj-Net [Yedroudj *et al.*, 2018]. In 2019, a low complexity network [Huang *et al.*, 2019] for JPEG has been proposed.

For more details about the steganalysis using Deep learning literature, we refer the reader to [Chaumont, 2020].

3.3 Quantitative Steganalysis by Machine Learning

Quantitative steganalysis, which aims to estimate the payload size (zero payload corresponds to the cover image), was introduced by [Fridrich *et al.*, 2003]. In this paper, the authors provide basic concepts for the design of steganalysis techniques to accurately estimate the number of changes to the cover image imposed during embedding. Using these concepts, the authors show how to estimate the length of the secret message for the most common embedding schemes, including OutGuess [Provos, 2001] and F5 [Westfeld, 2001] for JPEG, and other image types in the palette and in the spatial domain.

In the meantime, quantitative steganalysis has not been the subject of as many studies in recent years. [Pevný *et al.*, 2009] and [Pevný *et al.*, 2012] proposed feature-based techniques together with a support vector regressors (SVR) to estimate the message length.

[Kodovský and Fridrich, 2013a] made a significant improvement using the recently proposed Rich Models.

3.3.1 Quantitative Steganalysis Using Rich Models

[Kodovský and Fridrich, 2013a] propose a quantitative detector that uses rich models, which we name in this thesis the *QS algorithm*. The approach adopts the recent advances in binary steganalysis using rich models and ensemble classifiers to extend the prior feature-based (small sized features) quantitative detectors ([Pevný *et al.*, 2009] and [Pevný *et al.*, 2012]) to rich model-based one. The authors propose a machine learning regression framework that brings together, through the gradient boosting process [Friedman, 2001], a large number of simpler base learners built on random sub-spaces of the original high-dimensional feature space.

In the following, we briefly explain the regression framework used to build the QS algorithm. Given $\mathbf{x}_i \in \mathcal{X}$, $y_i \in \mathcal{P} = [0, 1]$, the objective is to find the regression function f minimizing the training error:

$$\sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i)) \tag{3.8}$$

where $L(y_i, f(\mathbf{x}_i))$ is the squared loss function:

$$L(y_i, f(\mathbf{x}_i)) = \frac{1}{2}(y_i - f(\mathbf{x}_i))^2. \tag{3.9}$$

Over M base learners, the calculation of f is performed by applying a generalized additive model (additive expansion)

$$f(\mathbf{x}; \{\mathbf{a}_m\}_{m=1}^{m=M}) = \sum_{m=1}^{M} h(\mathbf{x}; \mathbf{a}_m), \tag{3.10}$$

where $h(\mathbf{x}; \mathbf{a})$, the base function, is a linear Ordinary Least Squares (OLS) regression. The calculation of the values of $\mathbf{a}$ is manipulated as an optimization problem that tries to find parameter vectors $\{\mathbf{a}_m\}_{m=1}^{M}$.

For this purpose, a simplified Gradient Boosting algorithm with the squared loss function from *Equation* (3.9) is applied, see Algorithm 1.

Algorithm 1 Gradient Boosting algorithm with a general loss function $L(y_i, f(\mathbf{x}_i))$.

1: $f_0(\mathbf{x}) = arg\min_{\beta} \sum_{i=1}^{N} L(y_i, \beta)$
2: **for** $m = 1$ *to* M **do**
3: $\quad y_i \leftarrow y_i - f_{m-1}(\mathbf{x}_i), i = 1, ..., N$ (update current error)
4: $\quad \mathbf{a} = \texttt{arg min}^{\mathrm{T}, c_L, c_R} \sum_{i=1}^{N} (y_i - h(\mathbf{x}_i; T, c_L, c_R))^2$ (least square regression)
5: $\quad f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \eta h(\mathbf{x}; \mathbf{a}_m), \ \ 0 < \eta \leq 1.$ (update regression function)
6: **end for**

The algorithm starts by initializing the $f_0(\mathbf{x})$ that minimizes the value of β over the whole set N of training features, see step 1 in Algorithm 1.

Then for each base learner $m \in M$, the training responses y_i are continuously updated to reflect the actual estimation error, $y_i - f_{m-1}(\mathbf{x}_i)$ (step 3). the parameter

vector $\mathbf{a} = (T, c_L, c_R)$ is determined through the least squares in step 4, where T is a threshold, and c_L and c_R are constants for the OLS regression. In step 5, the regression function is updated and regularized using the learning rate η.

3.3.2 Quantitative Steganalysis by Deep Learning

As far as we know, the only Quantitative detector based on deep learning has been proposed by [Chen *et al.*, 2018]. This approach exploits the recent advances in deep learning-based binary detectors. The authors propose a design, *the bucket estimator*, that use features extracted from the activation of such CNN binary detectors to predict the payload size. This approach provides about 30% reduction in the MSE (Mean Squared Error) of the payload estimator when compared to the QS algorithm. The basic concept of the bucket estimator is to switch the outputs of the binary values to real values; This is achieved by replacing the loss function *softmax* with the MSE and using the embedded payloads as continuous value class labels.

The first step in the approach is to build a k bucket of CNN (VNet [Chen *et al.*, 2017]) binary detectors D_{α_i} trained on the cover class and the class of embedded stego images with a fixed payload α_i, $i = 1, ..., k$ as indicated in *Figure* 3.14.

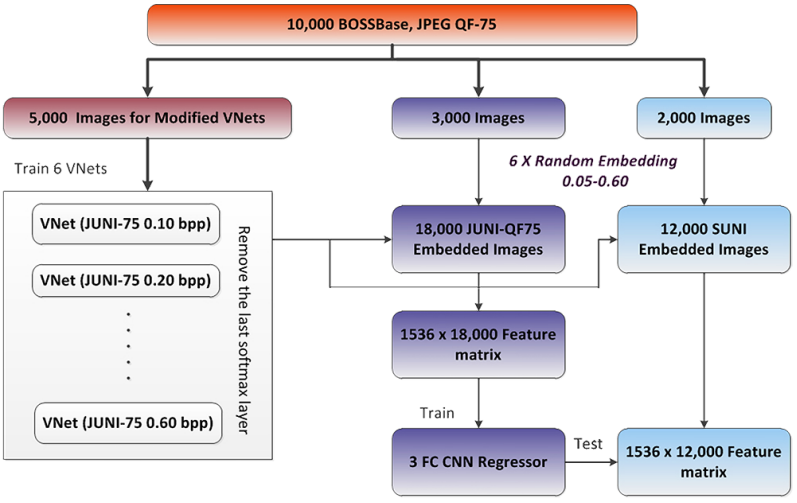

FIGURE 3.14: Data-set preparation and training for J-UNIWARD with quality factor 75. The image is from the original article [Chen *et al.*, 2018]

Next step is to concatenate the last M activation features connected to the classifier part, the fully connected layers (feature extraction part of these detectors), into a $k \times M$ dimensional feature vector and a payload regressor shown in *Figure* 3.15. Finally, the regressor is trained on concatenated features of stego images embedded with payloads α chosen uniformly randomly from some fixed interval I. The regressor is a three-layer fully connected neural network (FNN) with $2kM$ neurons in each layer and an output neuron.

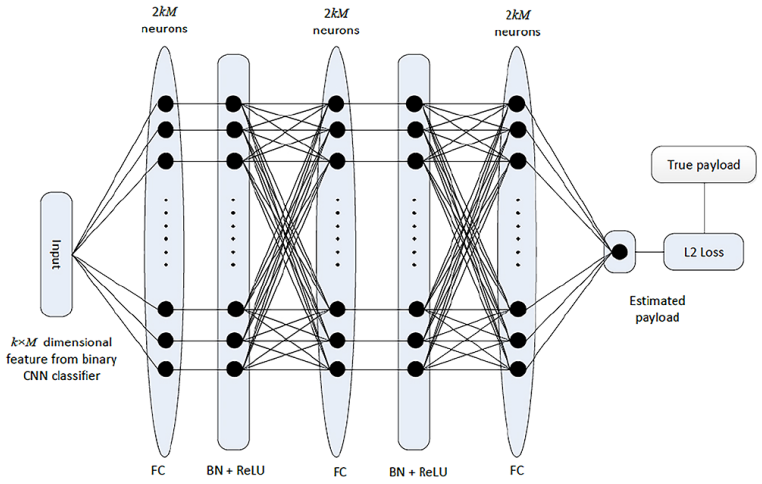

FIGURE 3.15: Three-layer FNN payload regressor used in both stego domains. The image is from the original article [Chen *et al.*, 2018]

3.4 Conclusion

In this thesis, we make assumptions that Eve does not know the payload embedded in each image (see *Section* 3.1.3), but she has to make a decision on a set of images whether this set is cover or stego, so she collects clues from this set, using conventional quantitative steganalysis algorithms in order to make the final decision. The clues can be collected from the estimates of quantitative detectors or from classifiers which do not make assumption of the embedded payload. Among the methods explained in this chapter, we have selected for the experiments proposed in this thesis state-of-the-art algorithms that are suitable with our assumption. These algorithms are the QS [Kodovský and Fridrich, 2013b] and GLRT [Cogranne and Fridrich, 2015] algorithms. They are compared in *Chapter* 6

in the quantitative scenario to see what is the best way for Eve to collect clues. These two algorithms are also compared in a binary scenario.

Chapter 4

Batch Steganography

Contents

In the previous sections, we discussed steganography and steganalysis, their principles, concepts and techniques. As the practical use of steganography inevitably involve several images, Alice may have access to more than one cover image, and Eve may intercept more than one cover or stego image from Alice. This motivates the idea of spreading the secret message into multiple cover images using small embedding rates.

In this chapter, to avoid misunderstanding, the terminology *embedding scheme* is used for the steganographic method of embedding in individual images, and *spreading strategy* for the problem of spreading the message in several cover images using a given *embedding scheme*.

4.1 Historical perspective

Steganography traditionally focused on embedding a message in one cover image at a time, but it is much more realistic for Alice to hide the message by *spreading* it over multiple images, i.e. a sequence of image. This *spreading* process is called *batch steganography* and is to be opposed to the *pooled steganalysis* where a *bag*[1] of images are analyzed in order to gather a set of clues, and thus to conclude to the presence/absence of a hidden message in the *bag of images*. Batch steganography and pooled steganalysis topics were introduced in [Ker, 2006] and became one of the most challenging open problems in the field these last years [Ker *et al.*, 2013a]. We present the pooled steganalysis topic in next chapter.

The batch steganography topic is illustrated in *Figure* 4.1. In the first article [Ker, 2006], the author gives an example of this scenario as follows: suppose a criminal wants to conceal information on his computer, using steganography, which is undeniable. He already has a large number of innocent cover images on his hard drive. To make sure he hides his secret information well, he could split it up into several small pieces and hide a little bit in each of the selected images

[1] In this case, we can use the term "bag" to emphasize that the order of the images is not important when performing the pooled steganalysis.

(batch steganography), thinking that this is safer than the alternative of filling in a smaller number of images for a high payload.

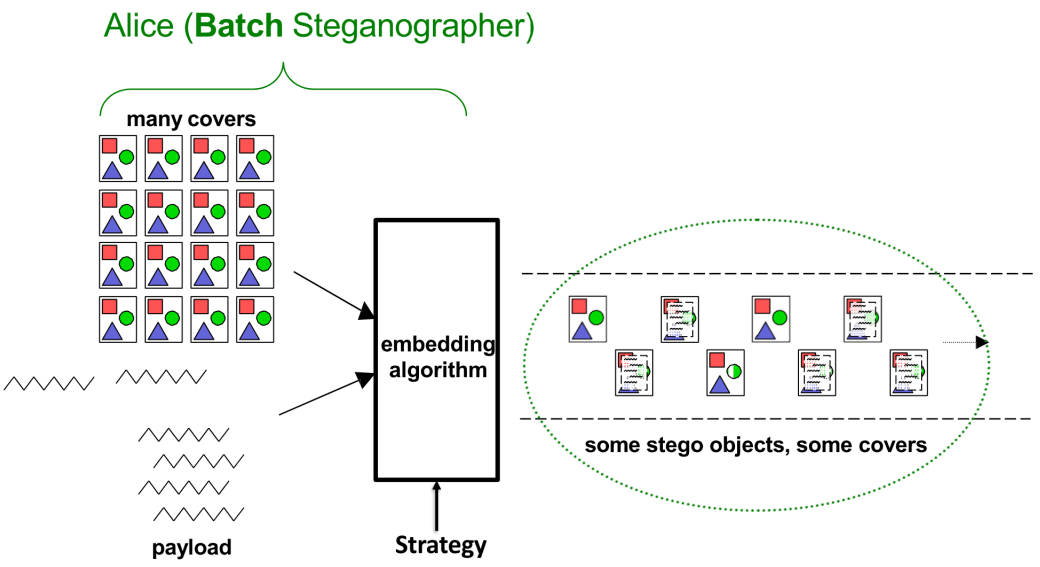

FIGURE 4.1: A scheme that illustrates the batch steganography process.

In [Ker, 2006], Ker proposes a theoretical framework for the undiscovered topic of batch steganography. In order to build this framework, Ker took some hypotheses like the following: he assumes that Eve knows the size of the bag, which is assumed to be fixed, and the message has a fixed length. Ker also assumed that the cover images have the same embedding capacity, i.e. all images can be embedded with a fixed maximum message length.

In [Ker, 2007a, 2008; Ker and Pevný, 2012b], Andrew Ker have pursued the theoretical studies on this topic, and compared the efficiency of some spreading strategies based on *non-adaptive embedding schemes* when the steganalysis detector is created to attack a specific embedding scheme. In this case, Andrew Ker showed that the optimal spreading strategy is to concentrate the message payload into as few cover images as possible (greedy strategy) or, at the opposite, to spread the payload as thinly as possible (linear strategy).

4.2 Some definitions

Recalling the definition of a steganographic channel from *Section* 2.1.1, where Alice uses a *cover* image to embed her *message* by using an *embedding* scheme, and a *stego key*. Generalizing this definition, in batch steganography, Alice needs a bag of *cover* JPEG images, to embed her *message* by using an *embedding* scheme, a *stego key* and a *spreading strategy*. All elements are represented in *Figure* 4.2.

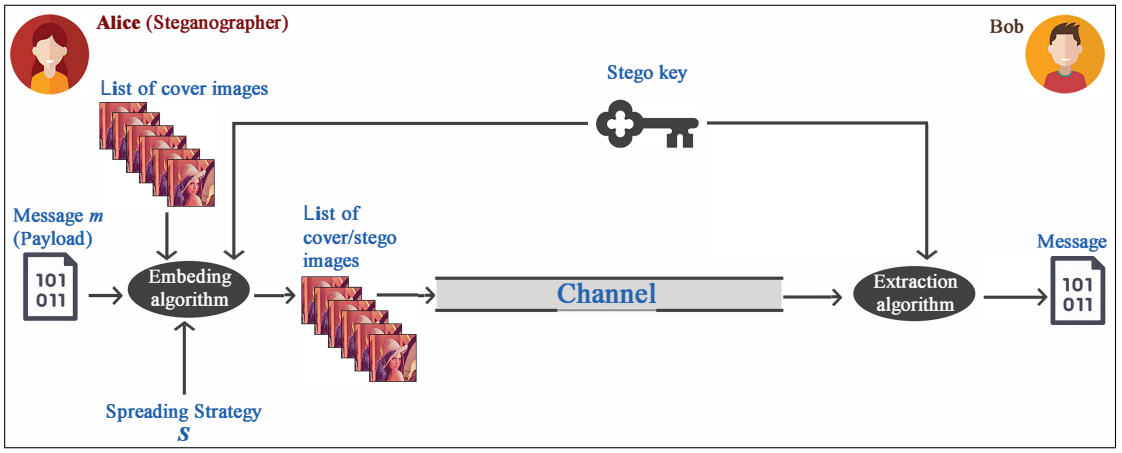

FIGURE 4.2: The steganographic channel and the principal elements for batch steganography.

Given a set of sequences of cover images $\{\mathcal{X}_i\}_{i=1}^{i=b}$, a set of keys $\mathcal{K}$ associated with the cover images (a key for each image), a set of messages $\mathcal{M}$ that Alice can embed, and a set of strategies $\mathcal{S}$ that Alice can use to spread the message, the embedding and extraction are expressed as :

$$Emb : \{\mathcal{X}_i\}_{i=1}^{i=b} \times \mathcal{K} \times \mathcal{M} \times \mathcal{S} \longrightarrow \{\mathcal{X}_i\}_{i=1}^{i=b} \tag{4.1}$$

and

$$Ext : \{\mathcal{X}_i\}_{i=1}^{i=b} \times \mathcal{K} \times \mathcal{S} \longrightarrow \mathcal{M} \tag{4.2}$$

Consequently, *batch steganography* consists in embedding a message $\mathbf{m} \in \{0,1\}^{|m|}$ in a sequence of cover images by using a spreading strategy, $s \in \mathcal{S}$.

4.3 Batch spreading strategies

The purpose of batch steganography is that Alice wants to spread in a sequence B of b images a message of relative total payload α. The value of α is related to the relative payload α_i of each image in the sequence. α_i is expressed in bits per pixels (bpp) for spatial image, in bits per non zeros ACs (bpnzAC) or bits per coefficient (bpc) for JPEG images. The message length, $|\mathbf{m}|$, is spread among a sequence of images $(\mathbf{x}_1, ..., \mathbf{x}_b)$ such that the message fragment lengths $\mathbf{|m|} = (|\mathbf{m}_1|, ..., |\mathbf{m}_b|)$, are given by:

$$|\mathbf{m}| = \sum_{i=1}^{b} (|\mathbf{m}_i|). \tag{4.3}$$

The value of $|\mathbf{m}_i|$ depends also on the steganographic capacity c_i, which is the maximum amount of data Alice can embed in the image $\mathbf{x}_i$. Note that steganographic capacity is a loosely-defined concept, indicating the size of payload which may securely be embedded in an image using a particular embedding scheme [Ker *et al.*, 2008].

c_i has several ways of being defined, depending on the spreading strategy and the embedding scheme used to embed the payload, these definitions will be discussed in the following.

In this section we discuss the spreading strategies that have been proposed between 2007 and 2020. We start by the oldest ones introduced in [Ker and Pevný, 2012c] and [Ker and Pevný, 2014] where the authors identified five strategies including those already introduced in [Ker, 2006], i.e. the *greedy strategy* and the *linear strategy*.

4.3.1 Greedy strategy

In the *greedy strategy*, Alice iteratively chooses the cover images with highest capacity yet to be used, and embeds a portion of the message equal to the estimated capacity of the image. After ordering the images by capacity $c_1 \geq c_2 \geq ... \geq c_b$,

Alice applies the following equation:

$$\begin{cases} |\mathbf{m}_i| = c_i, \forall i \in \{1, ..., I\}, \\ |\mathbf{m}_I| = |\mathbf{m}| - \sum_{i=1}^{I-1} |\mathbf{m}_i|, \\ |\mathbf{m}_i| = 0, \ \forall i \in \{I+1, ..., b\}, \end{cases} \tag{4.4}$$

where I denotes the smallest possible number of images with sufficient capacity, i.e.

$$I = \arg\min_{i} \sum_{j=1}^{i} c_j \geq |\mathbf{m}|,$$

with c_i is the maximum capacity of the cover image $\mathbf{x}_i$. In his empirical experiments, [Ker, 2006] uses the LSB replacement embedding scheme and the values c_i were the maximum amount of replacement that the algorithm can perform, which leads to a fixed value of c_i. Furthermore the author considered a fixed amount of data in each image.

In [Ker and Pevný, 2012c, 2014], the capacities are calculated based on the embedding scheme, the authors exploit the capacity value estimated by the embedding scheme while embedding. For example he used F5, JPHideSeek, and OutGuess schemes. In case where the embedding scheme does not provide a capacity estimate, the author fixed the capacity to certain value. This is the case of the nsF5 scheme, the author fixed the capacity estimate to 0.8 bpnc.

The authors proposed another way to chose the images to embed in the *greedy strategy*, which simulate the case where Alice does not have a previous knowledge on all the capacities of the images, so it is not possible to select the cover with high capacity. In this case, Alice randomly chooses an image $\mathbf{x}_i$ to embed, estimates its capacity c_i and embeds $\mathbf{m}_i$ which is equal to c_i. This may lead to the use of a larger number of images. The authors named it the *random* strategy.

4.3.2 Even strategy

The *linear* strategy is also linked by authors [Ker, 2006; Ker and Pevný, 2012c, 2014] to the capacities by distributing the message $\mathbf{m}$ into all available cover images proportionately to their capacity. This is formulated in the following equation, ignoring the fractional bits in $|\mathbf{m}|$:

$$|\mathbf{m}_i| = \frac{c_i|\mathbf{m}|}{\sum_{j=1}^{b} c_j}. \tag{4.5}$$

Without taking the capacities into consideration, *Equation* (4.5) become

$$|\mathbf{m}_i| = \frac{|\mathbf{m}|}{b}.$$

For this last equation, the message is distributed evenly throughout the bag, so that the size of the message for each image is equal to the length of the message divided by the number of b cover images. The authors named it the *linear* strategy, also named *even strategy*. It is worth noting that in later works, the authors claims that they used the linear strategy in their experiments regardless to the images capacity, because the embedding was done in modern adaptive embedding schemes. In the rest of this document we will reference the linear strategy the one distributing evenly into b covers.

Note that the last strategy proposed by the authors was the *sqroot* strategy, where the message is spread among all the b images with the length of the fragments being proportional to the square root of their capacities, i.e

$$|\mathbf{m}_i| = \frac{\sqrt{c_i}|\mathbf{m}|}{\sum_{j=1}^{b} \sqrt{c_j}}.$$

4.3.3 *uses-*β strategy

[Pevný and Nikolaev, 2015] used the *greedy* and *linear* strategies for spreading the payload. In this article, the authors work under the assumption that Alice

does not apply any cover selection strategy, meaning that the cover images in the sequence of images are randomly selected according to the probability distribution of the cover images. In addition, they have simplified the problem of payload distribution by not considering the capacity of the cover images. Furthermore, the authors proposed the *uses-β* strategy which is a fusion of *greedy* and *linear* ones. This strategy is formulated in the following equation :

$$|\mathbf{m}_i| = \frac{|\mathbf{m}|}{\beta b}.$$

where $\beta \in]0, 1]$ is a fraction of the available cover images where Alice spreads the message evenly. This strategy is equivalent to the linear strategy for β =1. It is also similar to the greedy strategy as it spreads messages in a fraction of images.

4.3.4 Modern strategies: IMS, DeLS, DiLS, AdaBIM

[Cogranne *et al.*, 2017] proposed three practical spreading strategies that overcame all the strategies explained above. Those was proposed for the purpose of minimising the statistical detectability of the embedded images.

- The first one is the *Image Merging Sender* (*IMS*) strategy, where Alice generates a unique image from all the b images from a sequence B, and lets the embedding scheme (an adaptive embedding scheme is used) spread the payload all over this ”big” image. They eventually open images of the same size to create the big picture.

  After embedding, the stego images of the original images are recovered. The concatenation is illustrated in *Figure* 4.3. In the IMS strategy, a cost value is computed for each DCT coefficient of the ”big” image (the adaptive embedding scheme defines the way the cost map is computed), and then the embedding is obtained with STC [Filler *et al.*, 2010, 2011].

- The second one is the *Detectability Limited Sender* (*DeLS*) strategy, which operates with statistical detectability. Here Alice adopts a cover model and

spreads payload over b images that communicates the required payload, so that each image from the sequence contributes with the same value as the Kullback-Leibler (KL) divergence (deflection coefficient) based on MiPOD [Sedighi *et al.*, 2016b] cover model[2]. Note that KL divergence is a fundamental measure of security to be applied to the domain of steganography and steganalysis [Ker *et al.*, 2013b]. The MiPOD cover model is suitable in this case as it is based on minimizing the statistical detectability. This scheme was created to work on spatial domain, given an image $\mathbf{x}_i$ of n pixels, Alice selects change rates $\kappa_i, i \in \{1, ..., n\}$ that minimize the deflection coefficient ϱ defined by:

$$\varrho = \sqrt{2 \sum_{i=1}^{n} \sigma_i^{-4} \kappa_i^2}. \tag{4.6}$$

The embedding change rates are first calculated by solving the following n+1 equations numerically (*Equation* (4.7)):

$$\kappa_i \sigma_i^{-4} = \frac{1}{2\lambda} \ln \frac{1 - 2\kappa_i}{\kappa_i} \tag{4.7}$$

and *Equation* (4.8):

$$R = \sum_{i=1}^{n} H(\kappa_i) \tag{4.8}$$

where σ_i^2 is the variance of the cover image pixel x_i, which is calculated using a variance estimator, λ is the Lagrange multiplier, and $H(x) = -2x \ln x - (1 - 2x) \ln(1 - 2x)$ is the ternary entropy function expressed in *nats*[3]. Once the variance rates are calculated, using the Lagrange multiplier method, they are converted into costs $\rho_i, i \in \{1, ..., n\}$ using the following equation:

$$\rho_i = \ln(\frac{1}{\kappa_i} - 2).$$

In MiPOD, these costs are then used by the Syndrome Trellis Codes [Filler *et al.*, 2010, 2011] to embed the R payload during the embedding process.

[2]The deflection coefficient is computed on dequantized images.

[3]A *nat* is a logarithmic unit of measurement of information or entropy, based on the Neperian logarithm and the powers of e rather than the base 2 logarithm that defines the bit.

It is worth noting that recently in [Cogranne *et al.*, 2020b], a new approach has been proposed to extend the use of MiPOD to the JPEG images. The new algorithm is named J-MiPOD and it is based on extracting the image to spatial domain to estimate the σ_i^2 values, applying linear transformation to compute variance of DCT coefficients $\sigma_i'^2$ which allows to continue the MiPOD embedding procedure on the JPEG image coefficients.

The principle of the DeLS spreading strategy is that, given a sequence of cover images $\{\mathbf{x}_i\}_{i=1}^{i=b}$, to fix the same ϱ for all the $\{\mathbf{x}_i\}_{i=1}^{i=b}$. This allows to deduce for each image the change rate $\{\kappa_i\}_{i=1}^{i=n}$ leading to ϱ using equation *Equation* (4.6). Then, Alice takes the calculated payloads for the images of the bag and uses them to build the bag of images using a chosen embedding scheme.

- The last one is the *Distortion Limited Sender* (*DiLS*) strategy. Here, Alice spreads payload over images so that each image from the bag contributes with the same value of distortion. The distortion of each image is calculated using the distortion function of the used adaptive embedding scheme.

Recently in [Sharifzadeh *et al.*, 2020], a novel spreading strategy has been proposed. The strategy is close to the IMS one and it is called ***Ada****ptive* ***B****atch size* ***I****mage* ***M****erging steganographer* (AdaBIM). It differs from the IMS strategy in that the AdaBIM spreads the payload non-uniformly among all the images according to their suitability measure of the image for steganography defined as

$$\sqrt[n]{\prod_{i=1}^{n}\sigma_i^2}$$

where n is the number of pixels in image and σ_i^2 is the variance of the i^{th} pixel.

The authors formalise the spreading (embedding in the big image in this case) as follows: Given a dataset of images $\mathcal{X}$ and a bag of size b. Assuming that the l^{th} bag contains images with indexes $(l-1)b, ..., lb-1$. The IMS strategy is used for spreading $n \times b \times \alpha$ nats among b images in each bag. $\forall j \in (l-1)b, ..., lb-1$, the

payload of the j^{th} image is calculated using the following equation:

$$\alpha_j = n\alpha + \frac{\alpha}{2} ln \left(\frac{\sqrt[n]{\prod_{i=1}^{n} \sigma_{ij}^2}}{\sqrt[b]{\prod_{k=(l-1)b}^{lb-1} \sqrt[n]{\prod_{i=1}^{n} \sigma_{ik}^2}}} \right),$$

where σ_{ij} is the variance of the i^{th} pixel of j^{th} image.

4.3.5 Two other variants

In [Liao and Yin, 2018], the authors propose two spreading strategies based on image texture complexity (SS-ITC) and distortion distribution (SS-DD) to perform the payload spreading.

The *Spreading Strategy Based on Image Texture Complexity* (SS-ITC) is similar to the Greedy strategy, since Alice iteratively selects the cover images with the highest *capacity* still to be used, and embeds a portion of the message proportional to the estimated capacity of the image.

The difference is how the capacity is estimated. While in the greedy strategy the capacity is related to the embedding scheme, in the SS-ITC strategy, the capacity c_i of an image $\mathbf{x}_i$ of size $l_i \times w_i$ is a function of its size, and the entropy of the image which could well represent the complexity of the image texture. Alice first calculates the image entropy h_i on the basis of the co-occurrence matrix $P(u, v, d, \theta)$

$$h_i = -\sum_u \sum_v P(u, v, d, \theta)\ log_2\ P(u, v, d, \theta).$$

where $P(u, v, d, \theta)$ is applied to count the number of times the pixel values u and v appear at the same time, d is the distance between the positions of u and v, and θ is the angle between the two pixels positions and the abscissa axis. By setting $d = 1$, $\theta = 0°, 45°, 90°, 135°$ the authors get four image entropies h_i and compute the average $\bar{h_i}$ for image $\mathbf{x}_i$. Therefore, the image entropy of the cover bag is $H = \{\bar{h_1}, \bar{h_2}, ..., \bar{h_b}\}$. Finally, the capacity c_i of image $\mathbf{x}_i$ is calculated by using the

following equation:

$$c_i = \frac{l_i w_i (\bar{h}_i - \bar{h}_{min})}{\bar{h}_{max} - \bar{h}_{min}}.$$

where $\bar{h}_{min}$ and $\bar{h}_{max}$ denote the minimum and maximum value in H, respectively. The embedding is done the same way as *Equation* (4.4).

the Spreading Strategy Based on Distortion Distribution (SS-DD) mainly spreads the payload according to the distribution of embedding distortion values. Alice calculates the distortion values ρ_i of the cover image $\mathbf{x}_i$ by applying the cost function of existing embedding schemes. All pixels of the cover image are sorted in ascending order in a list, according to their distortion values. The payload distribution in each image, is determined by the origin of the first $|\mathbf{m}|$ pixels of the list. $|\mathbf{m}_i|$ will be equal to n_i, the number of pixels belonging to image $\mathbf{x}_i$. The authors give a simple example of this strategy, given two different cover images $\mathbf{x}$ and $\mathbf{y}$ with the size of 3×3. First Alice calculates the embedding distortion values ρ by applying the cost function from the embedding scheme.

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{1,1} & \mathbf{x}_{1,2} & \mathbf{x}_{1,3} \\ \mathbf{x}_{2,1} & \mathbf{x}_{2,2} & \mathbf{x}_{2,3} \\ \mathbf{x}_{3,1} & \mathbf{x}_{3,2} & \mathbf{x}_{3,3} \end{bmatrix} \rightarrow \begin{bmatrix} \rho_{\mathbf{x}_{1,1}} & \rho_{\mathbf{x}_{1,2}} & \rho_{\mathbf{x}_{1,3}} \\ \rho_{\mathbf{x}_{2,1}} & \rho_{\mathbf{x}_{2,2}} & \rho_{\mathbf{x}_{2,3}} \\ \rho_{\mathbf{x}_{3,1}} & \rho_{\mathbf{x}_{3,2}} & \rho_{\mathbf{x}_{3,3}} \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{1,1} & \mathbf{y}_{1,2} & \mathbf{y}_{1,3} \\ \mathbf{y}_{2,1} & \mathbf{y}_{2,2} & \mathbf{y}_{2,3} \\ \mathbf{y}_{3,1} & \mathbf{y}_{3,2} & \mathbf{y}_{3,3} \end{bmatrix} \rightarrow \begin{bmatrix} \rho_{\mathbf{y}_{1,1}} & \rho_{\mathbf{y}_{1,2}} & \rho_{\mathbf{y}_{1,3}} \\ \rho_{\mathbf{y}_{2,1}} & \rho_{\mathbf{y}_{2,2}} & \rho_{\mathbf{y}_{2,3}} \\ \rho_{\mathbf{y}_{3,1}} & \rho_{\mathbf{y}_{3,2}} & \rho_{\mathbf{y}_{3,3}} \end{bmatrix}$$

Next, Alice sorts all the distortion values in an ascending order:

$$\rho_{\mathbf{x}_{1,1}} < \rho_{\mathbf{y}_{1,1}} < \rho_{\mathbf{x}_{2,1}} < \rho_{\mathbf{x}_{3,1}} < \rho_{\mathbf{y}_{1,2}} < \rho_{\mathbf{y}_{2,1}} < \rho_{\mathbf{x}_{1,2}} \ \ldots$$

The message distribution is determined by the distribution of distortion values in the first $|\mathbf{m}|$ pixels. For $|\mathbf{m}| = 7$ bits, following the above ordering, then $|\mathbf{m}_\mathbf{x}| = 4$ bits and $|\mathbf{m}_\mathbf{y}| = 3$ bits, meaning that 4 bits will be embedded in $\mathbf{x}$, and 3 bits will be embedded in $\mathbf{y}$. Note that SS-DD is not very far from IMS except that the cost

computation in SS-DD is done independently and the embedding too, whereas in IMS the process is global and the optimisation is global in the "big" image.

Figure 4.3 illustrates multiple examples of batch spreading strategies. Not all the strategies can be visually illustrated. Note that for the greedy strategy, it is not necessary for the message to be concentrated in the upper left corner due to the random choice of images when spreading, The image is just an illustration.

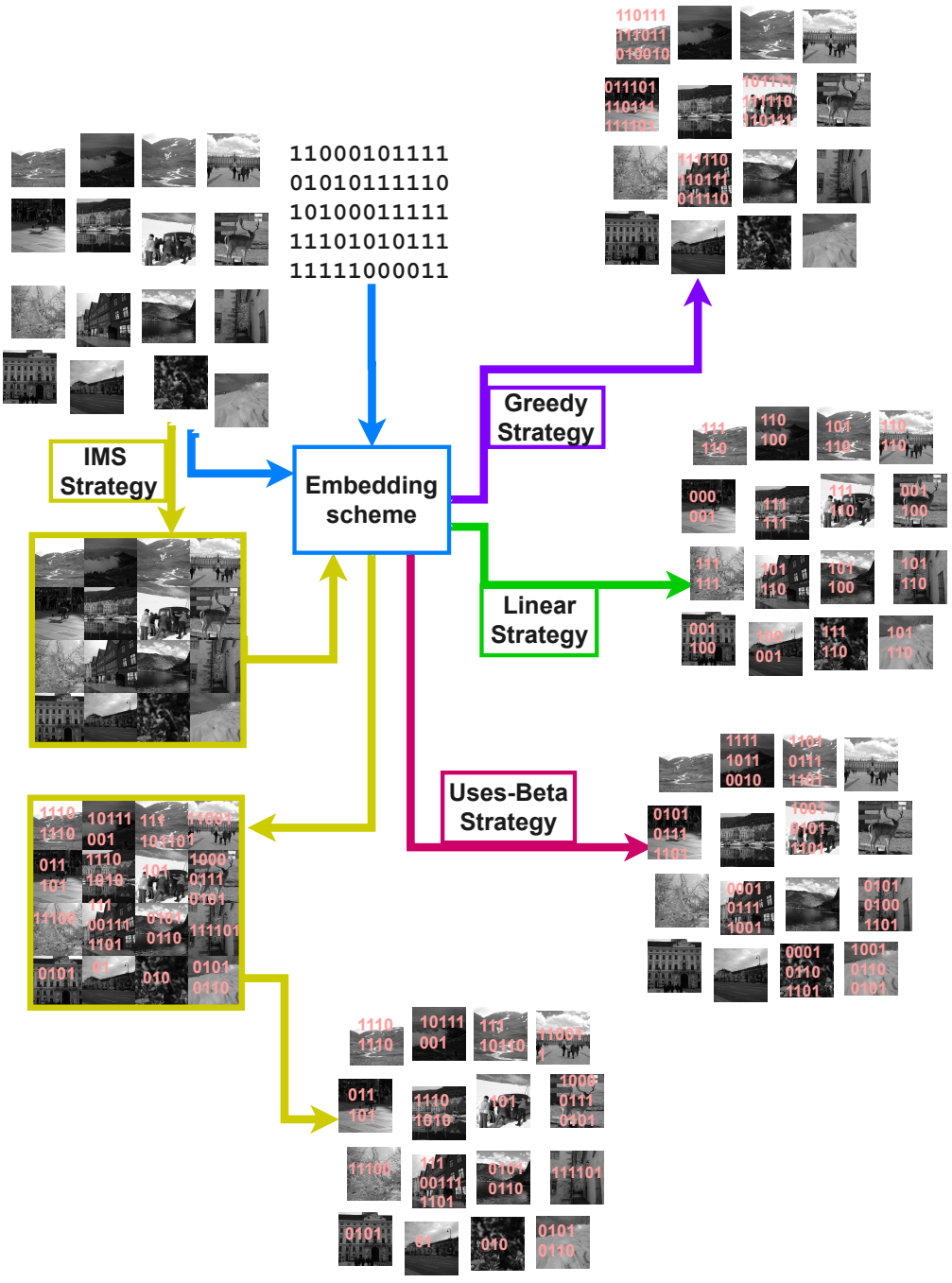

FIGURE 4.3: A scheme that illustrates four of batch spreading strategies.

4.4 Conclusion

In this chapter, we have discussed the problem of batch steganography. We began by defining the problem and then briefly presented the historical perspective. Then we discussed the most modern batch spreading strategies. Among all the strategies explained above, in the following experiments section, we use the $DeLS$, $DiLS$, IMS, *greedy*, *linear* and *uses-*β strategies. The $SS - ITC$ and the $SS - DD$ have not been used in our experiments and are postponed to future work.

Chapter 5

Pooled Steganalysis

Contents

In this chapter we discuss the opposite of batch steganography, that is pooled steganalysis, where Eve deals with several images that Alice sent. Eve extracts evidence from each image by predicting the presence/absence of a hidden message or its length, and then combines these shreds of evidence to make a final decision on whether or not there is hidden information in a whole set of images. The scenario is illustrated in *Figure* 5.1.

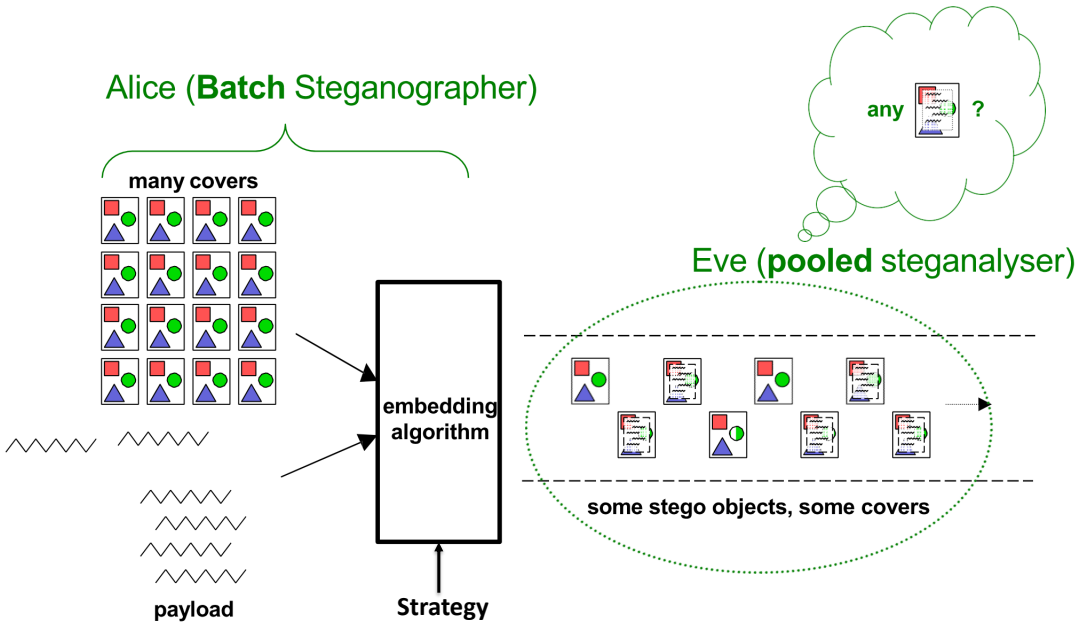

FIGURE 5.1: An illustration of batch steganography and pooled steganalysis.

5.1 Historical perspective

In 2007, in addition to batch steganography, [Ker, 2006] proposed a theory for the pooled steganalysis. Recalling the example of a criminal who wishes to hide information on his computer using batch steganography (see *Section* 4.2). The author defines the way the authorities deal with this situation as follows, *"When the authorities impound his computer, they are faced with a dilemma: how do they know which pictures to examine? Even possessing state-of-the-art steganalysis, they still observe fairly large false positive rates, and so if they test every picture on his computer they will inevitably turn up a lot of positive diagnoses – even if he is not a steganographer at all. They must run their statistical detector on every*

picture individually, and then find some way to combine the detection statistics into an overall "pooled" steganalysis for the presence of data, possibly spread across all the images.". This example led to the formulation of the problem in *Section* 5.2.

5.2 Some definitions

5.2.1 Pooling function

[Ker, 2006] has formulated the problem of pooled steganalysis as follows: The batch steganographer Alice uses a bag B of b images $\{x_i\}_{i=1}^{b} \in \mathcal{X}^b$, with $\mathcal{X}$ the images learning set. Each one of the bags may be cover or stego. Eve applies a Single Image Detector (SID), denoted by the function

$$f : \mathcal{X} \mapsto \mathbb{R}, \tag{5.1}$$

with the assumptions that f is an unbiased quantitative steganalysis detector (a detector which estimates the message length, see *Section* 3.1.2). Eve then applies f on the bag of images in order to get scores $\{f(x_i)\}_{i=1}^{b}$. She then aggregates these scores by using the function

$$g : \mathbb{R}^b \mapsto \mathbb{R}, \tag{5.2}$$

in order to get a single real output which allows to classify the bag as *cover* or *stego*. The function g is named the *pooling function*, see [Pevný and Nikolaev, 2015].

In what follows, we consider what is explained in this section for performing pooled steganalysis, i.e. the choice of f and g. This is illustrated in *Figure* 5.2.

5.2.2 Square root law of steganographic capacity

In *Chapter* 4, we discussed the notion of steganographic capacity, and its inter-relation to spreading strategies. In this section, we discuss the relationship between

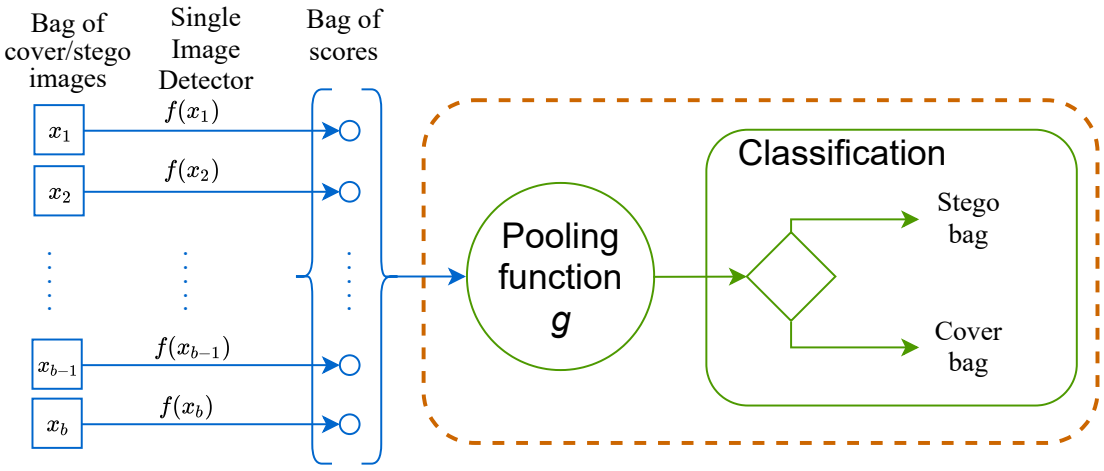

FIGURE 5.2: A scheme that illustrates how Eve uses a pooling function g to aggregate evidence from multiple images in order to make a final decision about the presence/abcence of a hidden message.

payload size and steganographic capacity for both batch steganography and single image (conventional) steganography.

In [Ker, 2004], in his experiments on non-adaptive embedding schemes, Andrew D. Ker observed a relation between the payload size and the steganographic capacity of an image. In [Ker, 2006], Ker applied non-adaptive embedding schemes to perform batch steganography. Based on the results obtained by applying some simple pooling functions (will be discussed in *Section* 5.3) on the bags of images with uniform capacity, Ker conjectured that the steganographic capacity increases only as the square root of the number of images in the bag. He proved this conjecture in [Ker, 2007b] with a mathematical theorem which states that pooling evidence from multiple images together should improve the accuracy of detection, provided that the payload grows quicker than the square root of the total steganographic capacity of the bag of images. This conclusion is named the *Square root law*.

Since then, many studies has been performed on the topic of the square root low. A study of this law on single images has been performed in [Ker *et al.*, 2008]. The authors tested contemporary methods of steganography and steganalysis at that time (using non-adaptive embedding schemes) to explain how square root law may be adapted to the steganographic capacity of a single image.

Furthermore, the robustness of this law in real life has been discussed in the survey paper [Ker *et al.*, 2013b], the authors said: *"What is remarkable about the square root law is that, although both asymptotic and proved only for artificial sources, it is robust and manifests in real life. This is despite the fact that practitioners detect steganography using empirical classifiers which are unlikely to approach the bound given by KL divergence, and the fact that empirical sources do not match artificial models."*

In all the studies cited above, the square root law was applied on non-adaptive embedding schemes. Recently, in [Ker, 2017], the author studied the applicability of the square root law to adaptive embedding schemes, where the embedding is not done uniformly throw the image, but using coding techniques. The author has made a theoretical conjecture on a square root law for adaptive embedding, which needs further study and experimental proof.

5.2.3 Sequential steganalysis

Later in [Cogranne, 2015], the author studied a scenario where Alice starts embedding messages after a while into a conveyed image stream, that is, a flow of images that are transmitted sequentially, one after the other. This approach is called sequential steganalysis or online steganalysis. The author theoretically formalized the problem of sequential analysis from a flow of transmitted images within the framework of hypothesis testing theory. The sequential detection problem is modeled as follows: after having sent an unknown number $\mathcal{V}-1$ of cover images, Alice starts embedding hidden message within the following images, with, by Kerckhoff's principle, a known embedding algorithm. Hence the sequence $x_1, ..., x_b$, $b \to \infty$, can be modeled as:

$$\begin{cases} x_i \sim \mathcal{P}_0 \ , \ \forall \ i \ < \ \mathcal{V} \\ x_i \sim \mathcal{P}_{\theta_i} \ , \ \forall \ i \ \geq \ \mathcal{V}, \ \theta_i \ \geq \ 0, \end{cases} \tag{5.3}$$

with $\mathcal{P}_0$ the distribution of the steganalysis results x_n before the $\mathcal{V}^{th}$ image (x_n are assumed by the author to be i.i.d (independent and identically distributed)), and $\mathcal{P}_{\theta_n}$ the distribution of the steganalysis results x_n at the $\mathcal{V}^{th}$ image where the embedding starts. θ_n represents a parameter of the distribution which is increased by the use of steganography.

Sequential steganalysis is considered as a particular case of pooled steganalysis, but it differ in assumptions ($b \to \infty$) and the way of construction of the pooling function (the author used the average as a pooling function g to aggregate scores from the function f).

5.2.4 Parzen-window for pooled steganalysis

Let us comment the Parzen window which is the most important ingredient of the steganalysis architecture proposed in [Pevný and Nikolaev, 2015]. The bag of SID scores, i.e. the vector $\mathbf{z} = \{f(x_1), ..., f(x_b)\}$, is transferred into a histogram representation thanks to the estimation by Parzen window. Given the Gaussian kernel function $k : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ with $k(x, y) = exp(-\gamma||x - y||^2))$, the Parzen window computation is such that for a bag $\mathbf{z}$, the resulting histogram is:

$$\mathbf{h} = \left[\frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_1),\ \ldots\ ,\ \frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_p)\right] \tag{5.4}$$

with $\{c_i\}_{i=1}^{p}$ a set of equally spaced real positive values belonging to the range $min_{x\in\mathcal{X}} f(x)$ and $max_{x\in\mathcal{X}} f(x)$. Each bin of the histogram $\mathbf{h}$, from *Equation* (5.4), is the result of the cumulative Gaussian distance between each component of $\mathbf{z}$ and a scalar from the set of predefined centers $\{c_i\}_{i=1}^{p}$.

Note that the histogram representation, $\mathbf{h}$, is of finite dimension p, whatever the dimension b of the bag, and that this representation is invariant to the sequential order in the bag.

We conclude this section by discussing the binary classification hypothesises applied on a bag of images, as the binary steganalysis is considered as a hypothesis test. Let us recall the definition of the steganographic channel explained in *Section* 4.2. In the case of multi-images, Eve has to decide if the set of images contains a hidden message. We have seen that in batch steganography, the message is spread among multiple cover images. Hence, in the case of pooled steganalysis, Eve tries to know whether there is a hidden message in the bag of images $\{x_i\}_{i=1}^b$ or not, and the test becomes:

$$H_0 : \{x_i\}_{i=1}^b \sim P_c^B,$$
$$H_1 : \{x_i\}_{i=1}^b \sim P_s^B$$

where P_c^B is a probability distribution of bags with only cover images, and P_s^B is the distribution of bags where at least one image x_i is stego. H_0 is then the null hypothesis ($\{x_i\}_{i=1}^b$ is a cover bag), and H_1 is the alternative hypothesis ($\{x_i\}_{i=1}^b$ is a stego bag). The probability distribution of images within P_s^B depends then on the cover source, the message length, the embedding scheme, and the *spreading strategy*.

Moreover, we conclude that the pooling is a key-point of the pooled steganalysis process.

5.3 Earlier pooling functions and assumptions

In [Ker, 2006], the author proposed some simple pooling functions. The author made some hypotheses to make the problem of pooled steganalysis easier to solve. These hypotheses have also been assumed by most of the papers on pooled steganalysis.

Hypotheses are based on the clairvoyant scenario, where the warden Eve has a perfect knowledge on the steganographic channel. This gives the following assumptions:

1. Eve *does know* the spreading strategy. Eve can then examine only one single strategy in each experiment.
2. Eve knows that each image has a fixed capacity and a fixed size.
3. The size of the bags is also known to Eve.
4. The relative total payload is fixed for all bags, i.e. the bag's payload is always the same.
5. All stego images have the same payload size.
6. The embedding scheme is known to the warden Eve.

In order to imitate the case where Alice can access a large quantity of images, the author used a set of 14,000 images, out of 20,000 on an online stock photo. The selection criterion was based on a hypothesis made by the author that *"each image should have macroscopic characteristics which indicate similar sensitivity to steganography"*. The selected images were all 640×416 pixels and had been stored as color JPEG images, then converted to grayscale. These images are embedded with the LSB replacement embedding scheme (see *Section* 2.1.2) to create stego bags.

The aggregation of the scores was performed using the SPA (Sample Pairs Analysis) detector (see *Section* 3.1.5). The author justifies his choice by saying that the SPA detector is simple in terms of computational complexity, and that the SPA makes it possible to approximately validate the shift hypothesis (see *Appendix* A).

Performing his experiments on bags of size $b \in \{10, 100, 1,000, 4,000\}$, the author proved that if Alice uses the Linear Strategy (that is the message is evenly spread into all the cover images), then the optimal pooling function is the average function

$$g_{mean} = \frac{1}{b}\sum_{i=1}^{b} f(x_i). \tag{5.5}$$

And when Alice applies the Greedy Strategy (where the message is spread into a few cover images as possible), then the optimal pooling function is the maximum

function:

$$g_{max} = \max_{i \in \{1,\dots,b\}} f(x_i). \tag{5.6}$$

These assumptions allow the optimal strategies to be found analytically but, in fact, they are hardly used in practice. Furthermore, due to the increased complexity of embedding schemes and spreading strategies, it is not evident that the shift hypothesis is sufficiently accurate.

5.4 Pooling functions and spreading strategies

Since this first paper in 2015 [Cogranne, 2015], three papers have addressed the problem of pooled steganalysis with a pooling function which aggregates SID score of each image individually [Cogranne, 2015], [Pevný and Nikolaev, 2015], and [Cogranne *et al.*, 2017].

- In the article [Cogranne, 2015], in his work on sequential steganalysis (see *Section* 5.2.3), the author makes the following assumptions:

  1. Eve *does not knows* the spreading strategy. He proposed a general framework, regardless of the spreading strategy.
  2. Eve knows that each image has a fixed size.
  3. The size of the bags is unknown to Eve. which is evident in the case of sequential steganalysis, see *Section* 5.2.3.
  4. The relative total payload is fixed for all bags, i.e. the bag's payload is always the same.
  5. All stego images have a fixed payload size, i.e. when Alice starts sending stego images, those will have a fixed payload size.
  6. The embedding scheme is known to the warden Eve.

  In this article, the author shows that, in this case where *Eve does not know the spreading strategy*, the best pooling function consists of *averaging* the individual scores.

- In the article [Pevný and Nikolaev, 2015], the authors used a machine learning approach (a linear classifier) to show how a pooling function combining the scores of a single image detector (SID) on a bag of images can be learned. The authors makes the following assumptions:

  1. Eve *does know* the spreading strategy. He examined only one single strategy in each experiment.
  2. Eve knows that each image has a fixed size.
  3. They assumed that the size of the bags is unknown to Eve.
  4. The bag payload size is fixed.
  5. All stego images have a fixed payload size.
  6. The embedding scheme is known to the warden Eve.

  The author proposed a pooled steganalysis architecture which is able to deal with case where Eve does not know the size of the bag. The general concept is illustrated in *Figure* 5.3. In the operational phase (i.e. when the general architecture is deployed), from this bag of b SID scores $\{f(x_1), ..., f(x_b)\}$, a *Parzen window* (see *Section* 5.2.4) is computed and lead to a histogram of of p bins, noted $\mathbf{h}$., which is then fed to the pooling function.

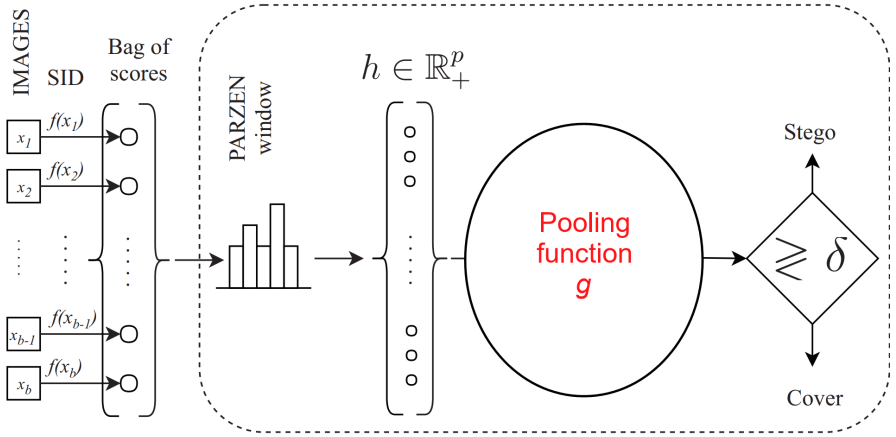

FIGURE 5.3: The general pooled steganalysis architecture from [Pevný and Nikolaev, 2015].

The pooling function is a linear classifier proposed in [Pevný and Ker, 2015], which directly minimizing the FP-50 measure, which is the false positive

rate at the 50% detection rate, see *Section* 3.1.4.2. This classifier finds a projection w minimizing the number of cover bags with a score higher than the average score (noted δ threshold) of stego bags:

$$arg \min_{w \in \mathbb{R}^m} \frac{\lambda}{2}||w||_2^2 + \frac{1}{l}\sum_{i=1}^{l} log(1 + e^{-w^T(\tilde{\mu}_s - \tilde{z}_i^c)}). \tag{5.7}$$

where $\{\tilde{z}_i^c\}_{i=1}^l$ is the set of cover bags, $\tilde{\mu}_s$ is the empirical mean of stego bags, and λ is the regularization constant. In that case, Pevny and Nikolaev observe that the knowledge of the strategy allows improving the steganalysis results since the knowledge of the strategy allows constructing an optimal pooling function, directly related the spreading strategy.

- In the article [Cogranne *et al.*, 2017], the author studies pooled steganalysis for clairvoyant scenario[1] . He also proposed the three modern spreading strategies explained in *Section* 4.3.4. The author makes the following assumptions:

  1. Eve *does know* the spreading strategy. He examined only one single strategy in each experiment.
  2. Eve knows that each image has a fixed size.
  3. The size of the bags is also known to Eve.
  4. The bag payload size is fixed.
  5. The payload of each image is unknown to Eve.
  6. The embedding scheme is known to the warden Eve.

  In the case where Eve *does know* the spreading strategy used by Alice, the author shows that this allows to better aggregate the individual SID scores, and thus to obtain better steganalysis results.

The conclusion shared by these three papers is that the optimal pooling function g applied on the SID scores depends on the steganographer's strategy used to spread

[1] In the clairvoyant scenario, Eve is assumed to have a perfect knowledge on the steganographic channel, this why she is also called "omniscient".

the messages among multiple images. Notice that this was already expressed in the seminal paper [Ker, 2006] in the case of non-adaptive embedding, see *Section* 5.3.

5.5 Assumptions and limits of a general pooled steganalysis architecture

The different assumptions for pooled steganalysis are whether the steganalyst knows or not:

- the embedding scheme,
- the test distribution (this is considered as known for the steganalyst in all of the articles cited in this manuscript).
- the single image payload size,
- the bag payload size,
- the bag size,
- the images size,
- the spreading strategy,

In order to construct a general pooled steganalysis architecture, we must define a pooled steganalysis system, integrating a pooling function, which would be *general*. To do so, we have to address many questions.

To be the more realist possible, we should choose a SID invariant to image size (see some preliminary work in [Tsang and Fridrich, 2018]) for a given detectability and should be robust to cover-source mismatch (CSM) and to the stego-source mismatch. The CSM problem is described in [Cancelli *et al.*, 2008]; a holistic solution is proposed in [Lubenko and Ker, 2012], and an atomistic solution proposed in [Pasquet *et al.*, 2014]. An example of a stego-source mismatch algorithm can be found in [Yousfi *et al.*, 2019]. In this thesis, we assume that the size of the

images is fixed and that there is no cover-source mismatch, neither stego-source mismatch.

We should choose a generic SID. It could be a quantitative detector such as [Kodovský and Fridrich, 2013b; Pevný and Nikolaev, 2015; Zakaria *et al.*, 2018; Chen *et al.*, 2018], or a detector outputting a score such as [Cogranne and Fridrich, 2015]. In this thesis, we will use the state-of-the-art quantitative detector described in [Kodovský and Fridrich, 2013b] (we made this choice according to the experiments explained in *Chapter* 6).

A general pooled steganalysis architecture must also be able to process a bag of any number of images. This is the case of the pooling functions proposed in [Cogranne, 2015] or [Pevný and Nikolaev, 2015] even if in those papers, experiments are performed by using bags of only one size. On the contrary, this is not possible in the algorithm described in [Cogranne *et al.*, 2017] where the hypothesis is that the steganalyst knows the number of images in the bag to analyze. In this thesis, we study an architecture dealing with any number of images in the bag, even if in the experiments, we have trained our model on bags of fixed sizes. We postpone the experiment with bags of various size for future work.

Lastly, such architecture must be able to process a bag of any payload size, which is not done in any of the papers [Cogranne, 2015; Pevný and Nikolaev, 2015; Cogranne *et al.*, 2017]. In this thesis, we make the same assumption as [Cogranne *et al.*, 2017] where the steganographer knows the mean payload of each bag and we postpone the experiments with any payload size in the bag for future work.

5.6 Conclusion

We conclude from this chapter that it is important to know the spreading strategy, as the best pooling functions in all approaches are for a given and known spreading strategy. Using an optimal pooling function for each existing spreading strategy would require to be able to estimate first the strategy used by Alice.

Another solution consists in defining a pooled steganalysis system, integrating a pooling function, which would be *general*, that is, could deal with all the spreading strategies, which is the purpose of this thesis.

In this thesis, the objective is to study the scenario where *Eve does not know* the spreading strategy used by Alice, while trying to work with less assumptions than the related works. Hence we will make the following assumptions:

1. **Eve *does not know* the spreading strategy.**
2. **Eve knows that each image has a fixed size.**
3. **The size of the bags is also known to Eve.**
4. **The bag payload size is fixed.**
5. **The payload of each image is unknown to Eve.**
6. **The embedding scheme is known to the warden Eve.**

And we propose to evaluate Eve's capability to obtain better results than those she would obtain by averaging the SID scores. In *Table* 5.1 we compare the difference in assumptions between this thesis and the related works.

Our proposed general architecture can be used independently to all the assumptions discussed in *Section* 5.5 except the cover-source mismatch and the stego-source mismatch (SIDs are not today able to solve those problems). We do the experiments on images with a fixed size, embedded with the same embedding scheme (J-UNIWARD) and we make a reasonable assumption that over time the steganographers maintain an average communicated payload $\bar{R} = 0.1$ bptc.

Our choice of the best SID is based on a study discussed in *Chapter* 6. In this chapter, we investigated two state-of-the-art steganalysis algorithms, QS and GLRT, discussed in *Chapter* 3. The goal was to compare them and find the best to use in our work for pooled steganalysis in *Chapter* 7.

TABLE 5.1: A comparison between hypotheses in [Ker, 2006], [Cogranne, 2015], [Pevný and Nikolaev, 2015], [Cogranne *et al.*, 2017] and this thesis. The hypotheses in boldface together made the difference between this thesis and the prior work.

hypothesizes / Article	Embedding scheme	single image payload size	bag payload size	bag size	images size	spreading strategy
[Ker, 2006]	known	unknown	known	known	fixed	known
[Cogranne, 2015]	known	known	known	unknown	fixed	unknown
[Pevný and Nikolaev, 2015]	known	unknown	known	unknown	fixed	known
[Cogranne *et al.*, 2017]	known	known	known	known	fixed	known
This thesis	known	**unknown**	known	**unknown**	fixed	**unknown**

Part III

Contributions

Chapter 6

Quantitative And Binary Steganalysis In JPEG: A Comparative Study

Contents

6.1 Motivation

Let us recall the purpose of this thesis, which is to study the case where Eve uses pooled steganalysis without knowing the spreading strategy. Let us also recall the theoretical approach of the pooled steganalysis architecture which uses the scores $f(x)$ calculated using a single image detector (SID) which is normally a quantitative steganalysis algorithm, see *Figure* 5.2.

An important question concerns the method of calculating the $f(x)$ scores and the quality of these scores. At the beginning of the thesis, there was a state-of-the-art quantitative detector (QS algorithm [Kodovský and Fridrich, 2013b]). Another interesting algorithm was the GLRT algorithm [Cogranne and Fridrich, 2015] which gave good results in binary steganalysis. Since the GLRT was more recent, we thought it was worth adapting it to extract the $f(x)$ scores.

This chapter compares the QS algorithm and the GLRT algorithm in the quantitative domain. We also extend this comparison to make a comparison in the binary domain. *Figure* 6.1 provides a general illustration of the binary and quantitative steganalysis scenarios.

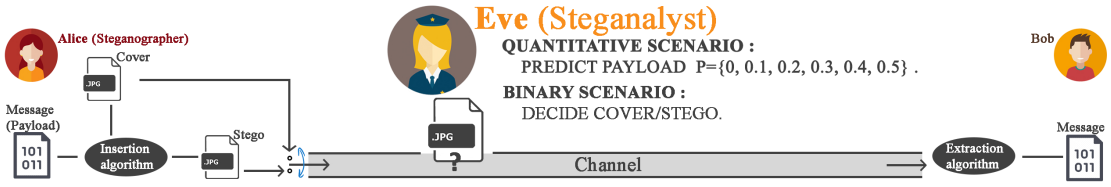

FIGURE 6.1: A general illustration of the binary and quantitative steganalysis scenarios.

6.2 Presentation of the algorithms

In the first scenario, we will study the binary steganalysis which is based on the GLRT-ensemble Classifier (GLRT), see *Section* 3.2.1.2. In the second scenario, we will study the quantitative steganalysis which is based on the QS algorithm, see *Section* 3.3.1. In all cases, payload estimation p can be continuous or discrete.

It is worth mentioning that in our experiments, both GLRT and QS algorithms use the same feature vectors for training, as depicted in *Figure* 6.2.

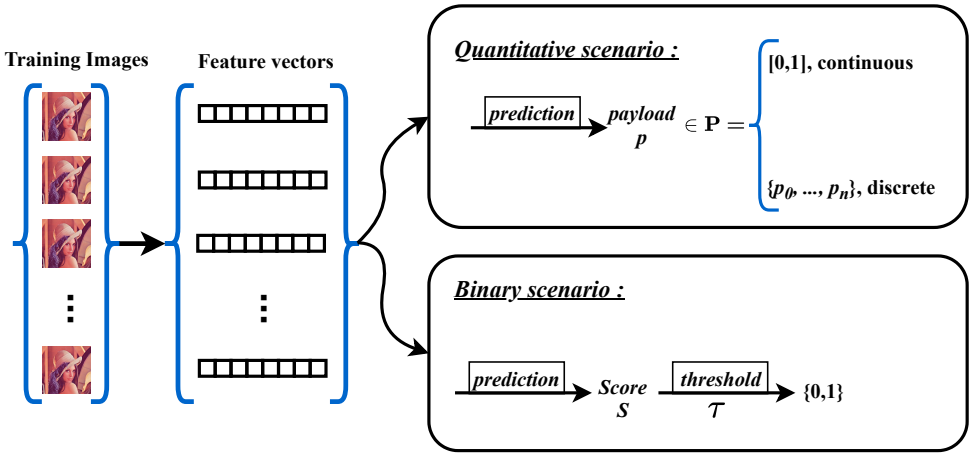

FIGURE 6.2: Schematic representation summarising the binary and quantitative steganalysis scenarios.

6.3 Comparison procedure

The results of the two algorithms are in different forms which obliges us to adapt them for comparison:

- In the **binary scenario**, we construct a binary steganalysis algorithm from the QS regressor to compare its results with the original GLRT classifier. We name the QS regressor the *QS-binary*.
- In the **quantitative scenario**, we construct two quantitative algorithms, the GLRT-multiclass and the GLRT-regression, to compare their results to the original QS algorithm.

6.3.1 Binary scenario

In this scenario, we have to transform estimated payloads given by the QS algorithm into a binary decision. For this, we propose to construct the QS-binary algorithm.

QS-binary algorithm

The QS-binary algorithm uses thresholding to transform the estimated payload given by the QS algorithm into a binary decision (0=cover / 1=stego):

$$\begin{cases} 1 & p_i > p_\tau, \\ 0 & p_i \leq p_\tau, \end{cases} \tag{6.1}$$

where for the i^{th} image vector, $p_i \in P = [0, 1]$ is the payload predicted from the original QS regressor. p_τ is a fixed threshold over P calculated in the validation phase and optimized to minimize the probability of error Pe (*Equation* (5.7)), when there is the same number of cover and stego image vectors.

To compare GLRT and QS-binary, we calculate the probability of error Pe, and we draw the Receiver Operating Characteristic (ROC) curves for both.

6.3.2 Quantitative scenario

To obtain quantitative results from the GLRT classifier, we construct two quantitative algorithms, the GLRT-multiclass and the GLRT-regression.

GLRT-regression algorithm

The GLRT-regression algorithm is a piecewise linear regression model, trained on a set of scores $S \in \mathbb{R}^L$ given from the GLRT classifier, to estimate the payloads $p \in P$, with $P = [p_0, p_n]$:

$$p = \begin{cases} p_0 & s \leq p_0 \\ a \times s & p_0 < s \leq p_n \\ p_n & p_n < s \end{cases}, \tag{6.2}$$

where $s \in S$ is the score obtained by the GLRT classifier before thresholding. All predictions belongs to the interval $[p_0, p_n]$. $a \in \mathbb{R}^*$ is a slope of a linear function, $p = a \times s$, whose construction is based on the assumption that the scores follow a standardised Gaussian distribution in the cover image dataset as in the stego image dataset whatever is the payload, and the "shift hypothesis" (the shift is proportional to the payloads). The regression function goes through $p_0 = 0$. We calculate the scores from the testing data, the same way as we did for the training data, to use them for predictions of payloads using our regression model of *Equation* (6.2).

GLRT-multiclass algorithm

The GLRT-multiclass algorithm can be created, in the case we have a discrete range of payloads $P = \{p_0, \dots, p_n\}$. We thus use a one-vs-one multi-class classifier which predicts a class by calculating the maximum of votes given by applying the GLRT between each couple of classes, the algorithm makes the final decision using a simple majority technique. We formalize it as follows:

- For n classes of payloads, let $i, j, k \in [0, n]$. Let I be an image vector. Let c_i be a class for payload p_i. Let $\zeta_{i,j}$ be a binary classifier between c_i and c_j such that $i < j$. These are $(n-1)n/2$ classifiers.
- Let V be the vector of votes where $V[k]$ contains the votes for c_k. We train all $\zeta_{i,j}$ then we test them on our testing data. The final decision for I is $c_k[I]$. It is calculated as follows: For all $\zeta_{i,j}[I]$, if $\zeta_{i,j}[I]$ is equal to c_i then $V[i] = V[i] + 1$, else $V[j] = V[j] + 1$.
- Finally, the value of k is calculated by

$$k = \underset{k}{argmax}\ V[k].$$

- Note that in case of equality of votes, the algorithm votes for the higher payload, as the higher payload is more detectable according to the square root law (see *Section* 5.2.2).

6.4 Experimental protocol

6.4.1 Algorithm parameters and implementation

To precisely examine the steganalysis algorithms, we use them with their optimal parameters. In the training phase, the GLRT classifier searches for the optimal value of feature space dimensionality d_{sub} and automatically determines the number of base learners L. For the QS algorithm, the hyper-parameters are set manually, with the same values as in the original paper [Kodovský and Fridrich, 2013b]

The two steganalysis algorithms, the J-UNIWARD embedding algorithm, and the GFR feature extractor, are implemented in MATLAB. Their implementations are available for download on the Web page of the Binghamton University[1].

In the quantitative scenario, we use the Scikit-learn Python package [Pedregosa *et al.*, 2011] to calculate the Root Mean Squared Error and the Mean Absolute Error. The Matplotlib Python package is used to plot the regression function.

6.4.2 JPEG feature vector construction

The first step in our experimental protocol is the preparation of the images data. We convert 10,000 512 × 512 grey-scale spatial images from BOSSbase into JPEG images, using the MATLAB's command imwrite, with quality factors 75 and 95. Then we use the advanced adaptive steganographic scheme J-UNIWARD to generate stego images with different embedding rates {0.1, 0.2, 0.3, 0.4, 0.5} bits

[1] `HTTP://dde.binghamton.edu/download/`

per nonzero AC DCT coefficient (bpnzAC). We restrict our work to this range following the same scale used in [Cogranne and Fridrich, 2015].

We use the 17,000-dimensional JPEG domain Rich Model (GFR), to extract the feature vectors from the cover and stego images, see *Section* 3.2.1.1.

We clean the feature vectors from NaN values (it occurs when the feature values are constant over images) and from constant values, to obtain 16750-dimensional feature vectors. Finally, we normalise the data using the normalization algorithm proposed in [Boroumand and Fridrich, 2017]. There are 10,000 feature vectors for cover images and 10,000 feature vectors for each payload which gives a total of 60,000 feature vectors.

6.4.3 Database construction

For GLRT, GLRT-regression and GLRT-multiclass, we use 10,000 covers and 10,000 stegos such that each five different payload sizes are equally distributed. A ratio of 1/5 is respected when selecting stegos; we choose the first 10% of stegos with payload 0.1 bpnzAC that corresponds to the first 10% of covers, the second 10% of stegos with payload 0.2 bpnzAC that corresponds to the second 10% of covers, and so on... The proportions are illustrated in *Figure* 6.3. Each time we randomly split the data into two equal parts, 50% for the training and validation phase, and 50% for the testing phase. There are thus 10,000 vectors for training and validation, and 10,000 vectors for testing.

For QS and QS-binary, the image vectors are with payloads 0, 0.1, 0.2, 0.3, 0.4, 0.5 bpnzAC such that the total number of features vectors is 20,000. We prepare them respecting a ratio of 1/6 for each payload. Additionally, we split the input data between training, validation or testing phases, with 8400 vectors for training, 2100 for validation and 9500 for testing. This is illustrated in *Figure* 6.4.

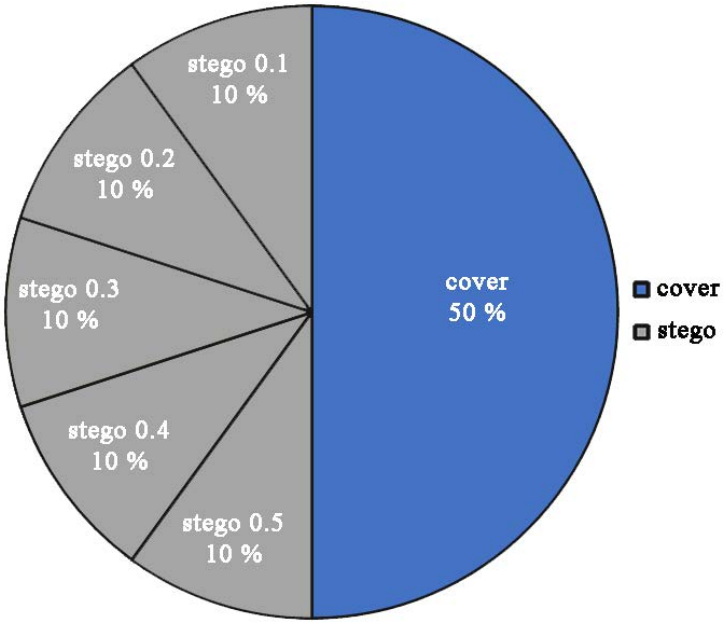

FIGURE 6.3: Images proportions for the GLRT algorithm. These proportions are respected in all training, validation and testing processes.

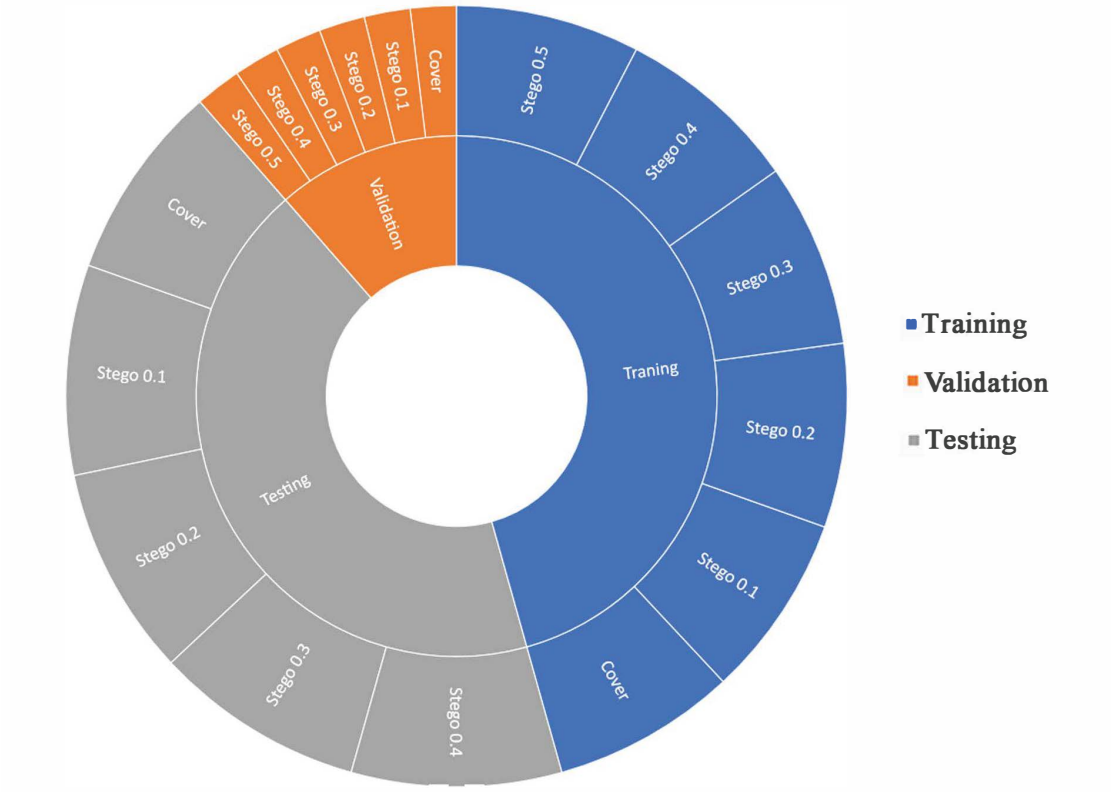

FIGURE 6.4: Images proportions for the QS algorithm. These proportions are respected in all training, validation and testing processes.

6.4.4 Comparison procedures

We explained in *Section* 6.3, the binary scenario and the quantitative scenario, and how we construct the algorithms. In this section, we explain how we apply these algorithms to our data.

6.4.4.1 Binary scenario

Below we explain how to use the QS-binary algorithm to compare its results with the original GLRT classifier.

QS-binary First, we apply the QS algorithm (training) on the data that we already prepared as in *Section* 6.4.3 and obtain the predicted payloads. Next, as explained in *Section* 6.3, we calculate p_τ that minimize the probability of error Pe in the validation phase. Finally, we classify the predicted payloads in the testing phase using p_τ and *Equation* (6.1). This way, from the QS prediction, we create a binary (cover/stego) classification which can be used for comparison with GLRT results.

6.4.4.2 Quantitative scenario

In the quantitative scenario, we compare the algorithms on $P = \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ bpnzAC, by applying the GLRT-regression and the GLRT-multiclass algorithms as explained in *Section* 6.3.

GLRT-regression We train the regression model explained in *Section* 6.3 on scores obtained from the GLRT classifier to predict a payload $p \in P$, as in *Equation* (6.2). The parameters p_0 is 0 and p_n is 0.5 as shown in *Figure* 6.5. Note that, in *Figure* 6.5 the regression is given for QF 75, and that the regression slope is a little bit more steep for QF 95 as shown in *Figure* 6.6. To train the model, we use the following procedure:

First, we apply the GLRT training algorithm which finds the optimal values of d_{sub} and L parameter and trains each FLD base learner.

Next, we compute the projection onto all base learners for training samples themselves (the regular use of the algorithm is to calculate the projection onto all base learners for testing samples then to continue into the testing phase). The projections are under $H0$ for training covers, and under $H1$ for training stegos, they all will be normalised by the covariance under $H0$ and by subtracting the mean value under $H0$.

Next, we compute the Generalised Likelihood Ratio (GLR) test which is given by the projection onto the vector of the mean projections under $H1$ normalised by the norm to ensure that the GLR follows a standardised Gaussian distribution under $H0$. Further details are available in *Section* 3.2.1.2. Next, we train a regression model, *Equation* (6.2) on the obtained training GLRs to predict payloads.

Finally, we calculate the GLRs from the testing data, the same way as we did for the training data, then we use them for predicting the payload using our regression model.

GLRT-multiclass Our numerical range of payloads to be predicted is discrete, which is close to the multi-class classification problem, so we apply the GLRT-multiclass classifier explained in *Section* 6.3:

A one-vs-one classifier uses the GLRT classifier to do the binary classification between each couple of classes. We represent the classes by numbers between 0 and 5 instead of using their real values for a simpler use, hence we get a list of votes $V = \{ V_{0,1}, V_{0,2}, V_{0,3}, V_{0,4}, V_{0,5}, V_{1,2}, V_{1,3}, V_{1,4}, V_{1,5}, V_{2,3}, V_{2,4}, V_{2,5}, V_{3,4}, V_{3,5}, V_{4,5} \}$ of binary decisions calculated from the binary classifiers for each image. These will be used to calculate the final decision as explained in *Section* 6.3.

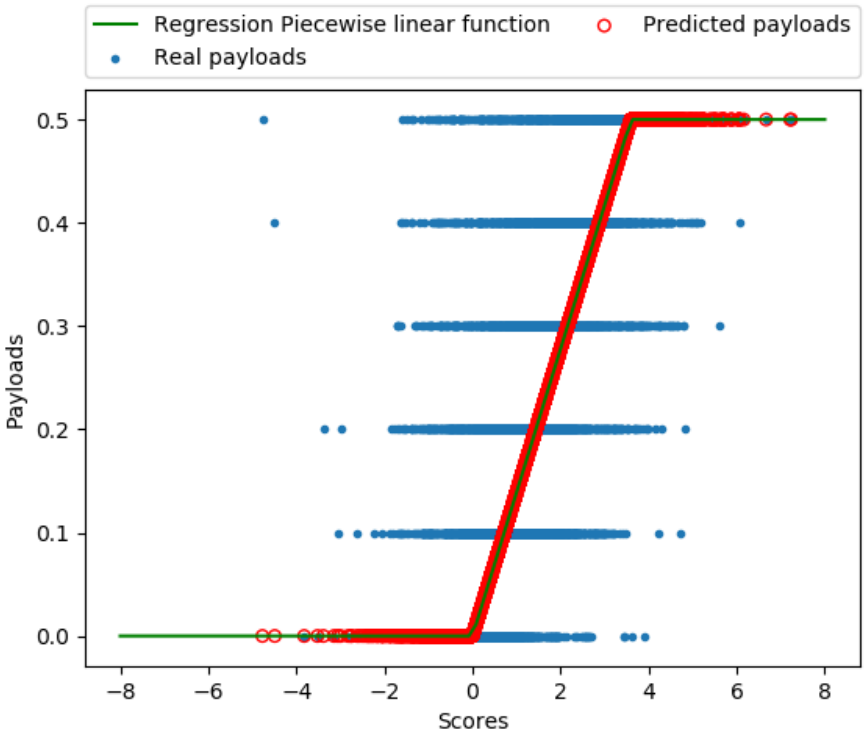

FIGURE 6.5: Quantitative scenario: schematic description for the regression piecewise linear function for quality factor 75.

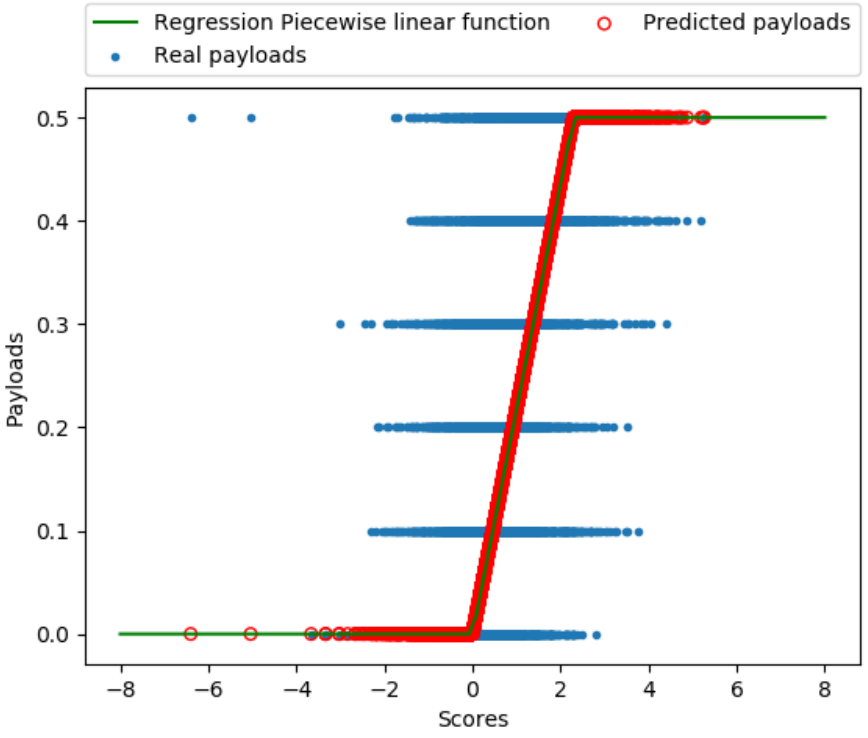

FIGURE 6.6: Quantitative scenario: schematic description for the regression piecewise linear function for quality factor 95.

TABLE 6.1: Binary scenario: prob. of error Pe, of GLRT and QS-binary approaches, for QF 75 ($p_\tau = 0.1482$) and 95 ($p_\tau = 0.2647$) .

	QS-binary	GLRT
QF 75	0.2479	0.2275
QF 95	0.3795	0.3438

TABLE 6.2: Binary scenario: detection power of GLRT and QS-binary appraoches for $\alpha_0 = 0.055$, for QF 75 and 95.

Quality factor	Payload	Clairvoyant, GLRT	Payload Mixture, GLRT	Trained for R=0.5, GLRT	Payload Mixture, QS-binary
75	0.5	0.9367	0.9151	0.9348	0.8829
	0.4	0.8091	0.7806	0.7878	0.6853
	0.3	0.5467	0.5258	0.5015	0.3739
	0.2	0.2665	0.2524	0.2165	0.1691
	0.1	0.1068	0.1032	0.0925	0.0823
95	0.5	0.5526	0.5487	0.5583	0.4364
	0.4	0.3741	0.3635	0.3303	0.2779
	0.3	0.2158	0.1974	0.1746	0.1629
	0.2	0.1188	0.1071	0.0981	0.0943
	0.1	0.0710	0.0679	0.0649	0.0655

6.5 Results & discussion

6.5.1 Binary scenario

We obtain results for the GLRT approach according to 3 different training scenarios:

- in the clairvoyant test, the embedding rate is known, i.e. training and testing are performed with the same payload.

TABLE 6.3: Binary scenario: Probability of error Pe, for GLRT and QS-binary approaches, for QF 75 and 95 in the case of different training scenarios.

Quality factor	Payload	Clairvoyant, GLRT	Payload Mixture, GLRT	Trained for R=0.5, GLRT	Payload Mixture, QS-binary
75	0.5	0.0585	0.0684	0.0596	0.2128
	0.4	0.1059	0.1198	0.1137	0.2203
	0.3	0.1842	0.1975	0.2006	0.2502
	0.2	0.2932	0.3075	0.3180	0.3253
	0.1	0.4059	0.4188	0.4277	0.4333
95	0.5	0.1975	0.2115	0.1954	0.2511
	0.4	0.2707	0.2802	0.2774	0.3305
	0.3	0.3490	0.3555	0.3544	0.4017
	0.2	0.4185	0.4272	0.4247	0.4565
	0.1	0.4714	0.4740	0.4740	0.4849

- training is performed on a uniform mixture of payloads.
- training is performed with a fixed payload $R = 0.5$.

Results for the QS-binary approach are obtained by training on a uniform mixture of payloads.

In the case of binary scenario, *Table* 6.1 shows a small superiority of the GLRT classifier with a difference in Pe of about 2% for quality factor 75 and about 4% for quality factor 95.

Table 6.2 and *Table* 6.3 present respectively, the detection power (i.e. the probability of detection of a stego image within all the examined stego images) for a probability of false alarm of $\alpha_0 = 0.055$ and the minimal total probability of error Pe.

TABLE 6.4: Quantitative scenario: Average Predicted Error (AVG), Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) for GLRT-regression, QS and GLRT-multiclass approaches, for QF 75 and 95.

Payload	**GLRT-regression**			**GLRT-multiclass**			**QS**		
	QF 75								
	AVG	**RMSE**	**MAE**	**AVG**	**RMSE**	**MAE**	**AVG**	**RMSE**	**MAE**
0	0.0541	0.0960	0.0541	0.0692	0.1298	0.0692	0.1312	0.1568	0.1312
0.1	0.1334	0.1229	0.0989	0.1197	0.1309	0.1017	0.1645	0.1094	0.0812
0.2	0.1614	0.1355	0.1141	0.1876	0.1359	0.1070	0.2182	0.0919	0.0749
0.3	0.2292	0.1544	0.1290	0.2868	0.1331	0.0980	0.2883	0.0909	0.0745
0.4	0.2826	0.1858	0.1495	0.3797	0.1148	0.0809	0.3623	0.0919	0.0704
0.5	0.3524	0.2103	0.1477	0.4548	0.0949	0.0452	0.4251	0.1021	0.0759
All		0.1508			0.1232			0.1071	
	QF 95								
	AVG	**RMSE**	**MAE**	**AVG**	**RMSE**	**MAE**	**AVG**	**RMSE**	**MAE**
0	0.0908	0.1498	0.0908	0.1494	0.2362	0.1494	0.2413	0.2506	0.2413
0.1	0.1431	0.1566	0.1224	0.1627	0.1925	0.1527	0.2478	0.1625	0.1478
0.2	0.1393	0.1466	0.1266	0.2084	0.1886	0.1646	0.2613	0.0916	0.0736
0.3	0.1826	0.1967	0.1703	0.2619	0.1896	0.1589	0.2816	0.0731	0.0599
0.4	0.2700	0.2200	0.1796	0.3420	0.1838	0.1368	0.3096	0.1166	0.0986
0.5	0.2821	0.2795	0.2180	0.3993	0.1874	0.1007	0.3422	0.1747	0.1580
All		0.1915			0.1963			0.1448	

From *Table* 6.2, one can notice that the greater the payload, the higher the performance for all algorithms and scenarios. Moreover, in the payload mixture scenario, the GLRT outperform the QS-binary in terms of detection power by $\sim 7.6\%$ for quality factor 75, and by $\sim 4.9\%$ for quality factor 95.

Furthermore, in *Table* 6.3, in the payload mixture scenario, the GLRT is better then the QS-binary in terms of probability of error Pe. The difference is by $\sim 6.5\%$ for quality factor 75, and by $\sim 3.5\%$ for quality factor 95.

We can conclude that the detection power is better for GLRT approach whatever the training scenario (clairvoyant, payload mixture or fixed payload) compared to the QS-binary approach.

6.5.2 Quantitative scenario

In *Table* 6.4, we compare the performance of the QS, the GLRT-regression, and the GLRT-multiclass approaches, all implemented with GFR features, see *Section* 3.2.1.1.

Unlike the binary scenario, here, the QS approach provides better results than the GLRT-regression and the GLRT-multiclass ones, with in average about 4% smaller RMSE than the GLRT-regression and about 2% lower RMSE than the GLRT-multiclass for quality factor 75. For quality factor 95, QS approach gives about 4% smaller RMSE than the others. But this is only true for high payloads. For small payloads, the QS approach gives less good results.

Note from *Figure* 6.7, *Figure* 6.8 and from *Table* 6.4 that in both quantitative and binary scenarios, the results are better for quality factor 75 than quality factor 95, especially for small payloads. This is due to image compression that makes the embedding changes more straightforward to detect [Song *et al.*, 2015].

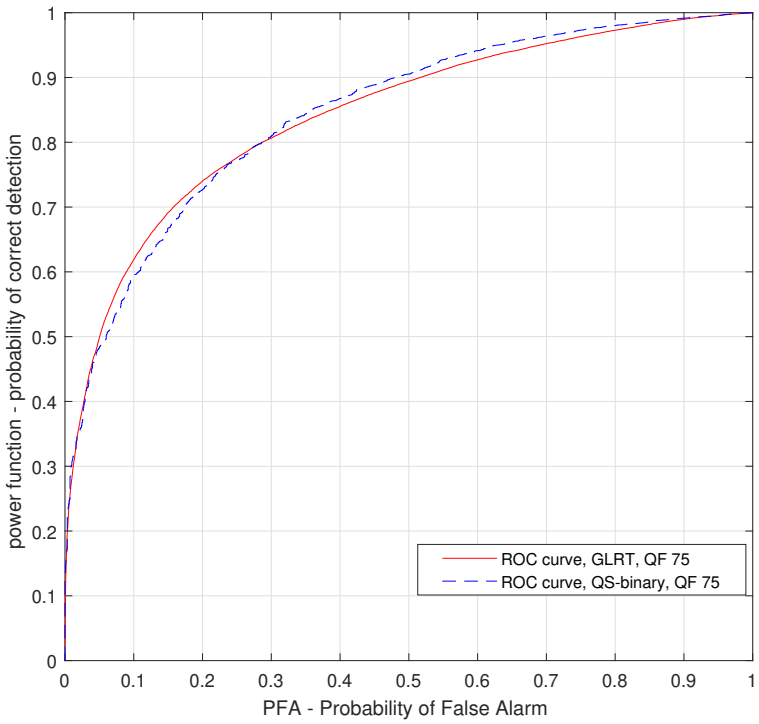

FIGURE 6.7: Binary scenario: empirical ROC curves for the QS-binary and the GLRT algorithms, for quality factor 75.

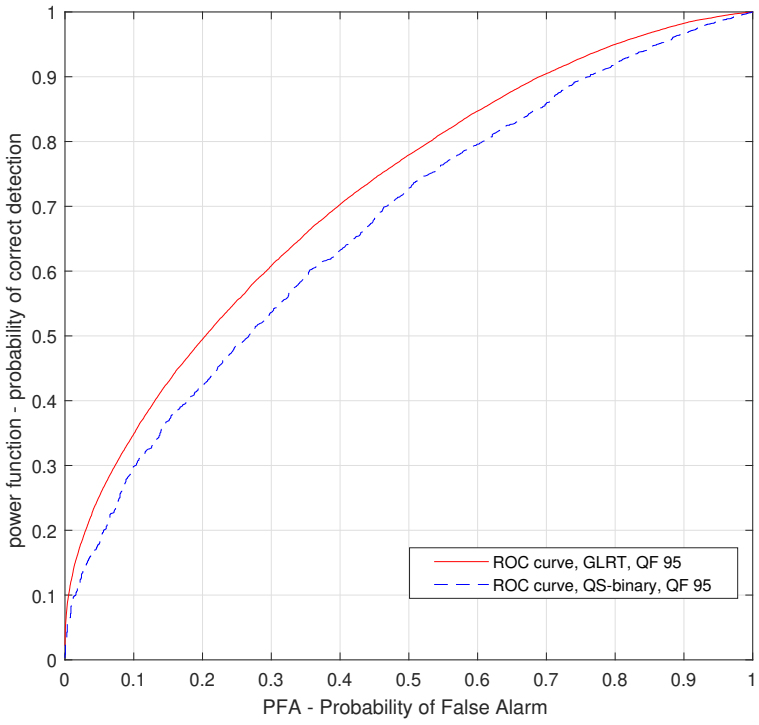

FIGURE 6.8: Binary scenario: empirical ROC curves for the QS-binary and the GLRT algorithms, for quality factor 95.

6.6 Conclusions

In this chapter, we investigated two state-of-the-art steganalysis algorithms, QS and GLRT. The goal was to compare them and find the best to use in our future work in pooled steganalysis.

Numerical results based on Content Adaptive Embedding Scheme and Rich Model show that the GLRT approach is slightly better than the QS one when doing binary steganalysis and that GLRT approach is marginally worse than the QS one when doing the quantitative steganalysis. GLRT approach seems more accurate to estimate payload when it is small. This may be due to the accuracy of the original GLRT classifier which is good for small payload [Cogranne and Fridrich, 2015].

Despite the broad difference between the binary and the quantitative scenarios, using an algorithm specially developed for one scenario in the other scenario context gives interesting results since the difference is between 2-4% in Pe or $RMSE$. It may be interesting to use this way of comparing algorithms developed for different scenarios to analyse new algorithms as [Chen *et al.*, 2018] where scores given by a classifier are used to train a regression model for payload estimation.

In conclusion, the QS algorithm seems better for the construction of a pooled steganalysis architecture, as it gives better results on average. This conclusion needs to be verified in practice, since the payloads spread over a set of images are normally very small in most cases (depending on the used spreading strategy).

Chapter 7

A Method To Deal With The Spreading Strategy For Pooled Steganalysis In JPEG

Contents

7.1 Motivation

This chapter is the core of this thesis. After studying the possible approaches to obtain the scores of the SID $f(x)$ (see *Chapter* 6), we are going to use these scores in pooled steganalysis. We remind that we study the scenario where *Eve does not know* the spreading strategy used by Alice, and the objective is to get better results than those Eve would obtain by averaging the SID scores $f(x)$ (*Section* 5.3).

For this, we propose that, first, Eve *learns* a set of well-known spreading strategies, and secondly we optimize a weighted sum of scores, using a SVM model, based on the different strategies in order to take a final decision. Notice that this approach does not guess the spreading strategy used by Alice, even if, thanks to the learning process, Eve is able to better *discriminate* between *cover* bag and *stego* bag, by distinguishing better the various statistics associated to each strategy.

The reader must understand that compared to the assumptions in the previous papers and especially to those in [Pevný and Nikolaev, 2015], which are:

1. Eve *does know* the spreading strategy. He examined only one single strategy in each experiment,
2. Eve knows that each image has a fixed size,
3. He assumed that the size of the bags is unknown to Eve,
4. The bag payload size is fixed,
5. All stego images have a fixed payload size,
6. The embedding scheme is known to the warden Eve,

the question which is raised in our thesis is first, about the ability for Eve to use some knowledge, gained from available spreading strategies, about the spreading strategies (without knowing exactly which one is used by Alice), and second, to do it in a more realistic scenario which implies to minimize the assessments: no

knowledge about the individual payload sizes, no knowledge about the spreading strategy, and eventually no knowledge of the bag size.

In *Section* 7.2 we briefly recall the list of spreading strategies used in the experiments, and we describe the general pooled steganalysis framework we have used to integrate (or to fuse) scores based on different strategies and to pool them in order to make the final decision (cover/stego bag).

In *Section* 7.3 we discuss the technical details of the question raised in this thesis. Finally, we present in *Section* 7.5 the results and discuss them. We conclude and give some perspectives in *Section* 7.6.

7.2 Batch spreading strategies and a general pooled steganalysis architecture

7.2.1 Batch spreading strategies ($\mathcal{S}$)

Recall that the embedding in a bag is done at a given payload size which is expressed in bit per total coefficients (bptc). The bptc is thus the size of the message in bits divided per the total number of pixels (i.e. the ACs and DCs coefficients) of all the images in the bag. We have compiled the batch spreading strategies (detailed in *Chapter* 4) in the following list:

1. **Greedy strategy,**
2. **Linear strategy,**
3. $Uses$-β **strategy,**
4. **Image Merging Sender (IMS) strategy,**
5. **Detectability Limited Sender (DeLS) strategy,**
6. **Distortion Limited Sender (DiLS) strategy.**

7.2.2 A general pooled steganalysis architecture

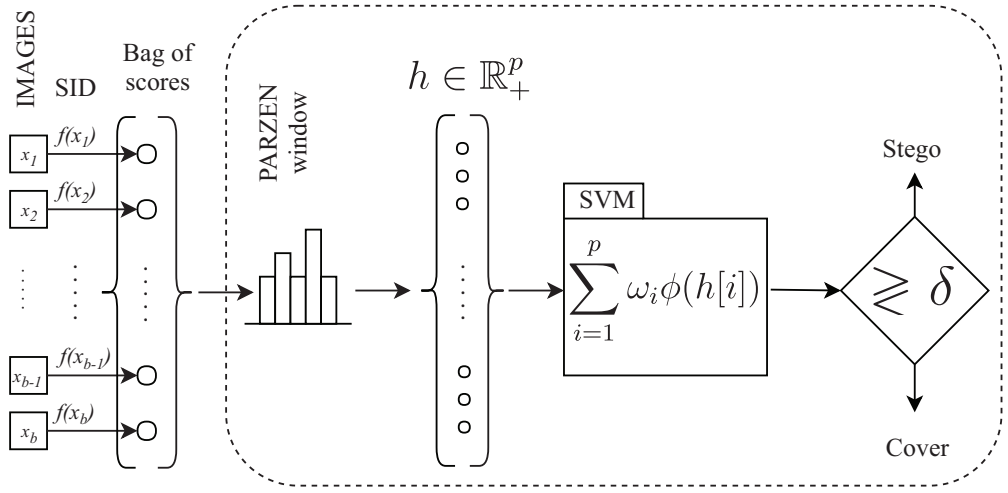

FIGURE 7.1: The general pooled steganalysis architecture. We have a bag of images $x_1 ... x_b$. A SID gives scores for each image $f(x_i)$ which can be seen as a bag of scores. The Parzen window allows to approximate a histogram of the scores of a given number of bins. This forms a vector $\mathbf{h}$ of dimension p. Then a SVM model trained on the $\mathbf{h}$ vectors. The SVM results a value which is compared to a threshold δ.ϕ is the re-description transformation function.

In this section, we recall the general pooled steganalysis architecture that were proposed in [Pevný and Nikolaev, 2015] for evaluating a given pooled steganalysis facing a given spreading strategy. *Figure* 7.1 resumes the different steps applied by Eve when she analyzes a bag of images.

In the operational phase (i.e. when the general architecture is deployed), the pooled steganalysis algorithm takes as input a bag made of b images $\{x_1, ..., x_b\}$ which may be cover or stego, and computes for each image the SID score, which is a real positive number, through the f function (see *Section* 7.1 and *Section* 7.3). It thus gives a bag of real numbers $\{f(x_1), ..., f(x_b)\}$. From this bag made of b values, where b is variable because it is not known in advance, a *Parzen window* (detailed below) is computed and lead to an approximation of the histogram of the distribution of scores, noted $\mathbf{h}$. $\mathbf{h}$ is made of p bins, independently of b. Finally, the pooling function, which is a SVM model, aggregates the p values of the histogram via a weighted sum. The resulting weighted sum is then compared to a threshold, noted δ in order to decide if the bag is cover or stego.

Let us comment the Parzen window which is an important ingredient of the architecture proposed in [Pevný and Nikolaev, 2015]. The bag of SID scores, i.e. the vector $\mathbf{z} = \{f(x_1), ..., f(x_b)\}$, is transferred into a histogram representation thanks to the estimation by Parzen window. Given the Gaussian kernel function $k : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ with

$$k(x, y) = exp(-\gamma||x - y||^2), \tag{7.1}$$

with an empirically calculated γ parameter to optimize the performance of steganalysis. The Parzen window computation is such that for a bag $\mathbf{z}$, the resulting histogram is:

$$\mathbf{h} = \left[\frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_1),\ \ldots\ , \frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_p)\right] \tag{7.2}$$

with $\{c_i\}_{i=1}^{p}$ a set of equally spaced real positive values belonging to the range $min_{x\in\mathcal{X}} f(x)$ and $max_{x\in\mathcal{X}} f(x)$, with $\mathcal{X}$ the images learning set. Each bin of the histogram $\mathbf{h}$, from *Equation* (7.2), is the result of the cumulative Gaussian distance between each component of $\mathbf{z}$ and a scalar from the set of predefined centers $\{c_i\}_{i=1}^{p}$.

Note that the histogram representation, $\mathbf{h}$, is of finite dimension p, whatever the dimension b of the bag, and that this representation is invariant to the sequential order in the bag.

Once the Parzen window is applied, the vector $\mathbf{h}$ of fixed dimension p is given to an SVM which pools the vector component in the re-description space for classification:

$$\sum_{i=1}^{p} \omega_i\ \phi(h[i]), \tag{7.3}$$

with ϕ the function redefining the feature space. Note that ϕ is, in practice, never computed because of the "kernel trick". We can see when looking to *Equation* (7.3), that the pooling function is a weighted sum where the weights ω_i are learnt during the SVM training. It is clearly more subtle to pool the set of features $\{\phi(h[i])\}_{i=1}^{p}$ of the bag than using the straightforward average or maximum.

7.3 Technical details

7.3.1 Assumptions and limits of the proposed pooled steganalysis architecture

In *Section* 5.5, we discussed the assumptions and limits of a general pooled steganalysis architecture. We presented the different assumptions that the steganalyst may or may not know when performing pooled steganalysis. We also discussed the questions that need to be asked in order to build a general pooled steganalysis architecture. In this chapter, we make the assumptions previously detailed in *Table* 5.1, where we made a comparison between the assumptions of [Ker, 2006], [Cogranne, 2015], [Pevný and Nikolaev, 2015], [Cogranne *et al.*, 2017] and this thesis.

7.3.2 Single Image Detector (SID)

We choose as Single Image Detector (SID), which is referred in this chapter as the function f, the feature-based Quantitative steganalysis algorithm proposed in [Kodovský and Fridrich, 2013b], applied on the 17,000-dimensional JPEG domain Rich Model — the Gabor features residuals (GFR) [Song *et al.*, 2015]. The Quantitative steganalysis algorithm is a machine learning regression framework that assembles, via the process of gradient boosting, a large number of simpler base learners built on random subspaces of the original high-dimensional feature space. See *Chapter* 6 for more details on the SID.

7.3.3 Some technical details about the strategy implementation

For the *DeLS* strategy, we use the MiPOD embedding scheme to calculate the deflection coefficient. The fact that MiPOD is a spatial domain embedding scheme

obliges us to make some additional adaptations for our experiments when working in the JPEG domain. In addition, to ensure the accuracy of measuring detectability in the JPEG domain, it was necessary to ensure that the size of the message payload is the same when measuring detectability in the spatial domain and when spreading in the JPEG domain. We therefore measured the payload on all AC and DCT coefficients in JPEG images (bpc), which gives the same ratio as the bit-per-pixel (bpp) measurement in the spatial domain, this is valid in this thesis as we use J-UNIWARD, which has no restriction in term of message size comparing to spatial domain. Also, we applied the MiPOD on de-quantified rounded images from our JPEG database.

7.3.4 Pooling functions

As explained in *Figure* 7.1, the general pooled steganalysis architecture provided in [Pevný and Nikolaev, 2015] produces a feature space which is represented by a Parzen histogram, and proposes to pool the values of this histogram thanks to a linear SVM classifier. In the article [Pevný and Nikolaev, 2015], only *one pooling* function (the SVM) is learnt for *one spreading strategy* since the study was on the comparison with the historical average and maximum pooling functions, depending on the embedded payload size in a bag. Additionally, the experiment done in [Pevný and Nikolaev, 2015] only uses old spreading strategies (greedy and linear; see *Section* 7.2.1).

In our study, we look at the behaviour of the architecture when it has learned to recognize *various spreading strategy*. Our approach is thus a pooling function which is able to face multiple spreading strategies. The experiment objective is to show that even if Eve does not have any information on the spreading strategy used by Alice, she can obtain better steganalysis results than when using a simple average or maximum, and she can be close to the results that she would obtain if she was clairvoyant i.e. if she knows the spreading strategy. The difference compared with the paper [Pevný and Nikolaev, 2015] is in the addressed question, and our chapter is thus in the natural continuity of the three state-of-the-art

papers [Pevný and Nikolaev, 2015; Cogranne *et al.*, 2017; Ker and Pevný, 2014]. The experiments are in agreement with various spreading strategies (the modern and recent ones) and with state-of-the-art two-step machine learning in order to build the Rich Model and the SID.

In order to study the efficiency of our *discriminative* pooling function in order to *discriminate* s among a set of strategies $\mathcal{S}$, we defined various pooling functions:

1. g_{mean}**:** This function is only an average applied on a Parzen histogram. The threshold τ_{min} is obtained by minimizing, on all the strategies, the total classification error probability under equal priors $P_e = \frac{1}{2}(P_{fa} + P_{md})$, where P_{fa} and P_{md} are the false-alarm and missed-detection probabilities.

2. g_{max}**:** This function is only a maximum applied on a Parzen histogram. The threshold τ_{max} is obtained by minimizing, on all the strategies, the total classification error probability under equal priors $P_e = \frac{1}{2}(P_{fa} + P_{md})$, where P_{fa} and P_{md} are the false-alarm and missed-detection probabilities.

3. g_{clair}**:** This function is the *clairvoyant* one. The training and the test are done with the knowledge of the used spreading strategy. The pooling function is obtained thanks to the use of an SVM, similarly as the g_{disc} function.

4. g_{disc}**:** This function is our *discriminative* pooling function and we train it on the Parzen histograms of all the strategies from $\mathcal{S}$. During the test, only one spreading strategy s will be tested at a time. g_{disc} is obtained through the learning of the various patterns from all the strategies and the minimization of the classification error between a cover bag and a stego bag (whatever the spreading strategy) in order to obtain the threshold. Thanks to the Parzen representation, the SVM re-description space, and the weighted sum, the general architecture learns with different spreading strategies, and it should be able to classify better than applying an average or a maximum.

7.4 Assessment of discriminative framework

The assessment of the discriminative framework is based on an experimental evaluation based on a large database of various JPEG images. The whole process is presented in *Figure* 7.2.

7.4.1 Data preparation

Our image database is built from the BOSSbase 1.01 [Bas *et al.*, 2011b]. We convert those 10,000 512 × 512 grey-scale **spatial** images into JPEG images, using the MATLAB's command *imwrite*, with quality factors 75.

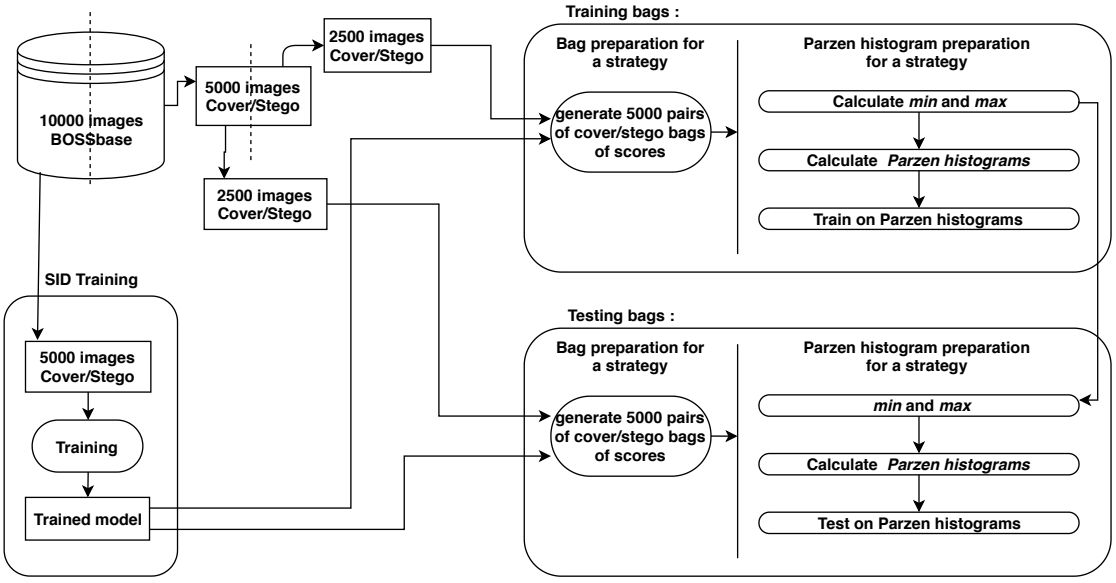

FIGURE 7.2: Protocol for the data-sets creation, the learning and the test.

The 10,000 cover JPEG images are split in two equal sizes sets. Set 1 (5,000 cover images) is used for the learning of the SID. Set 2 (5,000 cover images) is used to create bags, which will be used to learn the pooling functions and to test the various pooling. All the embedding are performed based on J-UNIWARD embedding scheme. The first step, consists in extracting a feature vector *Gabor Features Residuals* (GFR) [Song *et al.*, 2015] of dimension 17,000. This feature vector is then cleaned from NaN values (it occurs when the feature values are constant over images) and from constant values, to obtain a 16 750-dimensional

feature vectors. Finally, we normalize this vector using the algorithm proposed in [Boroumand and Fridrich, 2017].

7.4.2 Training the SIDs

In order to train the quantitative Single Image Detector, we respect the following ratios between covers and stegos:

From one cover image (that is 1 feature vector), we generate 10 stego images with a payload which range is fixed to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1} bpc[1], this gives 11 images. As we have 5,000 cover images, this gives a total of 55,000 features vectors (11×5,000). Indeed, these ratios are chosen to avoid any bias in the results, such that the cover and each one of the ten different payload sizes are equally distributed. This way the resulted SID model will guarantee a fair scoring between each payload (payload 0 for covers).

7.4.3 Bags preparation

The remaining 5,000 cover images (set 2) are then used for batch steganography which will results in images bags. More precisely 2,500 cover images are used for training the pooling function, and the remaining 2,500 cover images will be used for the test.

We create 5,000 "cover" image bags by dispatching cover images of set 2 into bags of size $b \in \mathcal{B} = \{2, 4, 6, 10, 20, 50, 100, 200\}$ images. The cover images are chosen randomly. Then we create the corresponding 5,000 "stego" image bags for each of the 6 strategies IMS, $DeLS$, $DiLS$, $Greedy$, $Linear$ and $Uses\text{-}\beta$ (β is fixed to 0.5). For each bag, the SID scores are computed leading to a vector of SID scores. The whole process is presented in *Figure* 7.2. Given a bag size $b \in \mathcal{B}$, the set of spreading strategies IMS, $DeLS$, $DiLS$, $Greedy$, $Linear$ and $Uses\text{-}\beta$ (β is fixed

[1]We adapted the J-UNIWARD algorithm to insert an amount of data measured in bpc instead of bpnzAC.

to 0.5) leads to a total of 30,000 stego bags plus 5,000 cover bags, with inside each stego bag a bit-rate of $\bar{R} = 0.1$ bptc.

7.4.4 Learning the pooling functions g_{disc} and g_{clair}

The clairvoyant and discriminative scenarios are applied using the same architecture denoted $\tilde{g}$, the difference is in the training phase. For both scenarios, following the same principle as in [Pevný and Nikolaev, 2015], the minimum and the maximum of the SID scores are computed, which allows to define $p = 100$ centers, equally spaced in the range $[min_{x\in\mathcal{X}} f(x), max_{x\in\mathcal{X}} f(x)]$. The γ values of *Equation* (7.1) are calculated using a method inspired from [Ker and Pevný, 2014]. We investigated γ in a range $\in \{10^n | n \in \{-4, -2, ..., 5\}\}$, and obtained the best performance for $n = -1$ ($\gamma = 1/10$). For values of $n > -1$, performance decreases markedly, while for values of $n < -1$, performance decreases dramatically towards random guessing. A Parzen histogram $\mathbf{h}$ can then be computed for each of the 35,000 bags.

The classifier, used for learning g_{disc}, is a SVM with a linear kernel. We use the SVM package from the free software machine learning library for the Python programming language Scikit-Learn [Pedregosa *et al.*, 2011]. The parameters are set to default, but the value of the kernel is set to 'linear'.

Figure 7.3 illustrates the two training procedures for the $\tilde{g}$ function in order to obtain g_{clair} and g_{disc}, it also illustrates the testing phase for both g_{clair} and g_{disc} functions. In the following we discuss the difference between these two function:

- **Learning the pooling function g_{clair}:** The pooling function g_{clair} is trained on 5,000 cover bags and their corresponding 5,000 stego bags for one strategy. The trained model is then tested for each strategy, on 5,000 cover bags and their corresponding 5,000 stego bags.

- **Learning the pooling function g_{disc}:** The pooling function g_{disc}, uses for the learning, 5,000 cover bags (i.e 5,000 Parzen vectors $\mathbf{h}$) and their corresponding 5,000 stego bags (i.e 5,000 Parzen vectors $\mathbf{h}$) equally distributed on the six spreading strategies. The trained model is then tested for each strategy, on 5,000 cover bags and their corresponding 5,000 stego bags.

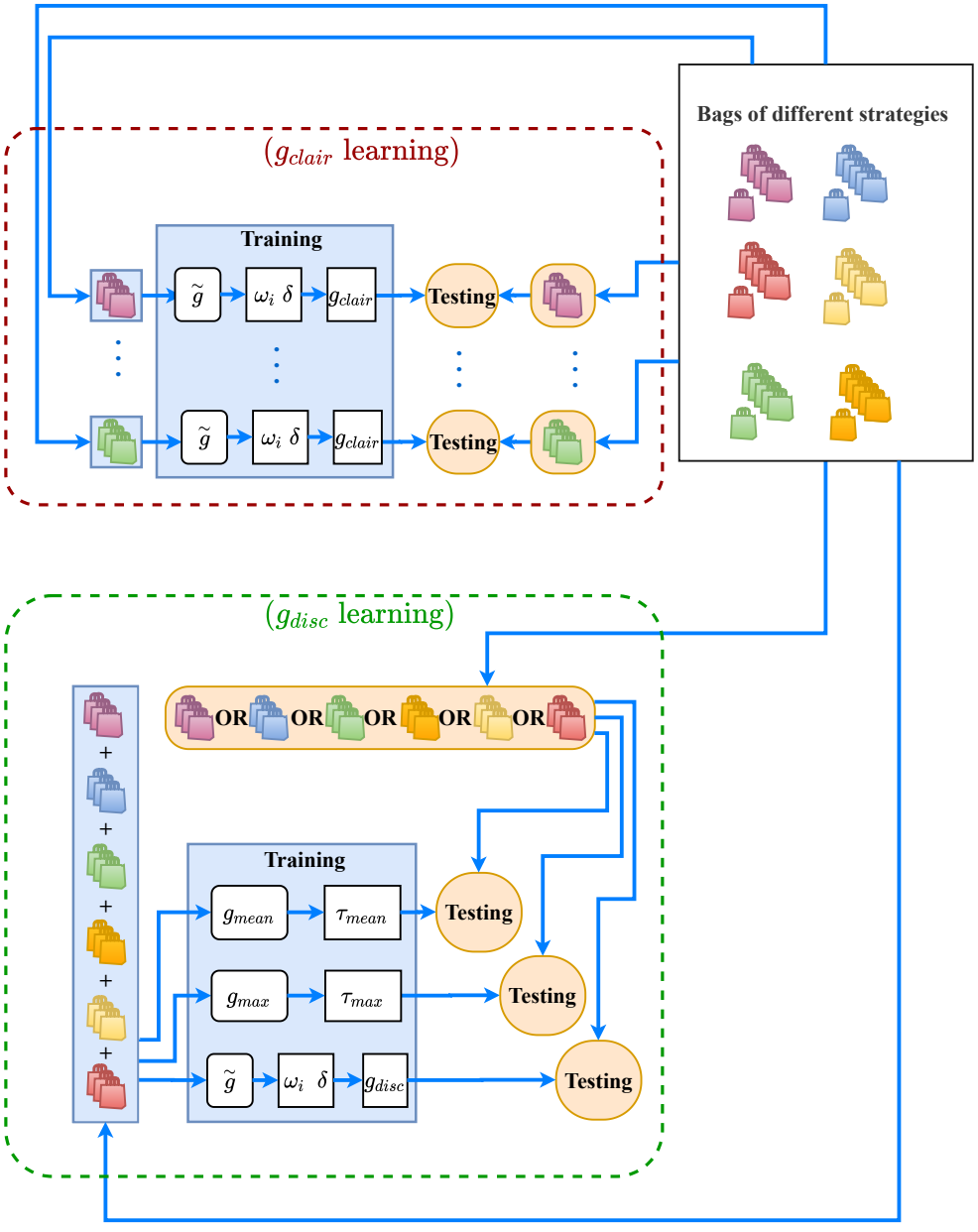

FIGURE 7.3: Learning the pooling functions g_{clair} and g_{disc}

7.5 Results

As shown in *Figure* 7.2, tests are done by using 2 500 cover image never seen, which allow to form a set of 5,000 pairs cover/stego bags for a fixed size b for a spreading strategy (IMS, $DeLS$, $DiLS$, $Greedy$, $Linear$ and $Uses\text{-}\beta$). The average probability of error, obtained by each pooling function, over 10 runs done each time with a different learning set (and testing set) of 5000 pairs of cover/stego bags, is then reported.

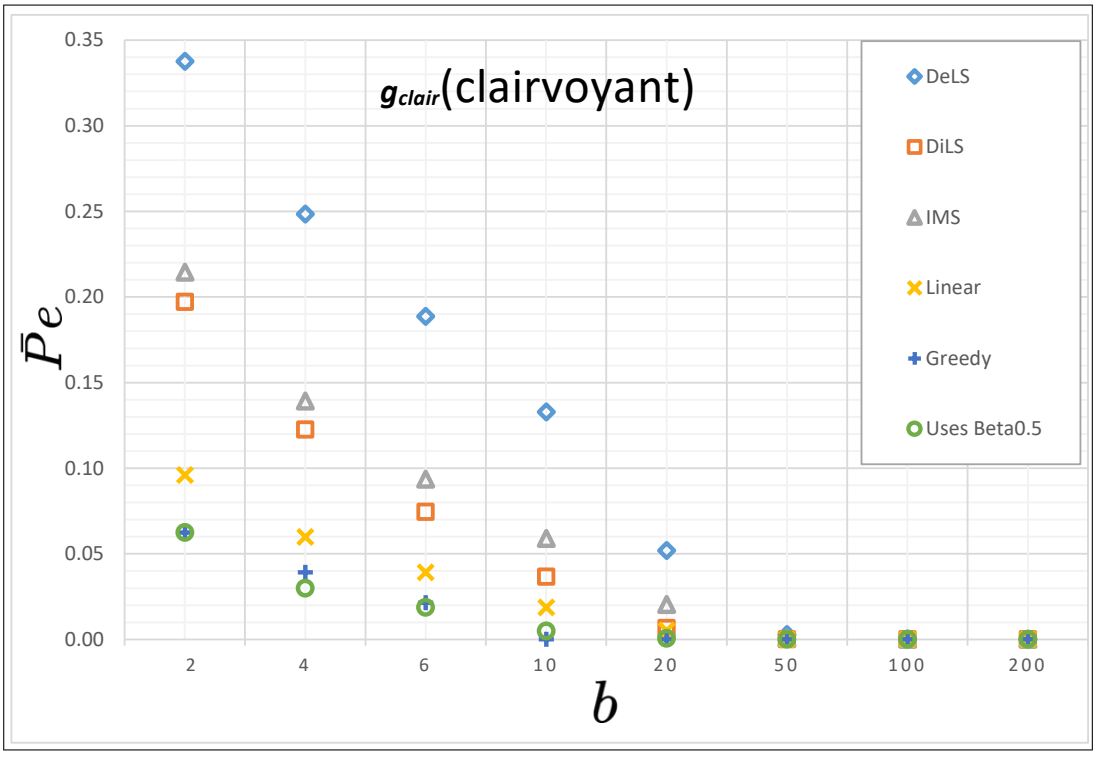

FIGURE 7.4: spreading strategies comparison in the *clairvoyant* case. Average probability of error under equal prior, $\overline{Pe}$, as a function of pooling bag size $b \in \mathcal{B}$ for an average payload 0.1 bptc for g_{clair} pooling function.

Figure 7.4 reports the results obtained with the *clairvoyant* steganalysis scenario i.e. with g_{clair}. One can notice that the DeLS is the best and it outperforms the IMS, which was more competitive to $DeLS$ in [Cogranne *et al.*, 2017], while the Greedy strategy is the worst. One can cluster the strategies into 3 groups: the $Greedy$ and $Uses\text{-}\beta$ which are highly detectable with a detectability which starts to coincide for bag sizes ≥ 10 with $\overline{Pe} \approx 0$, the strategies $DeLS$, IMS and $DiLS$ which are the more secure ones, and the $Linear$ strategy which falls between the

two other groups. The strategies $DeLS$ and IMS becomes totally detectable at $b = 100$ with $\overline{Pe} \approx 0$.

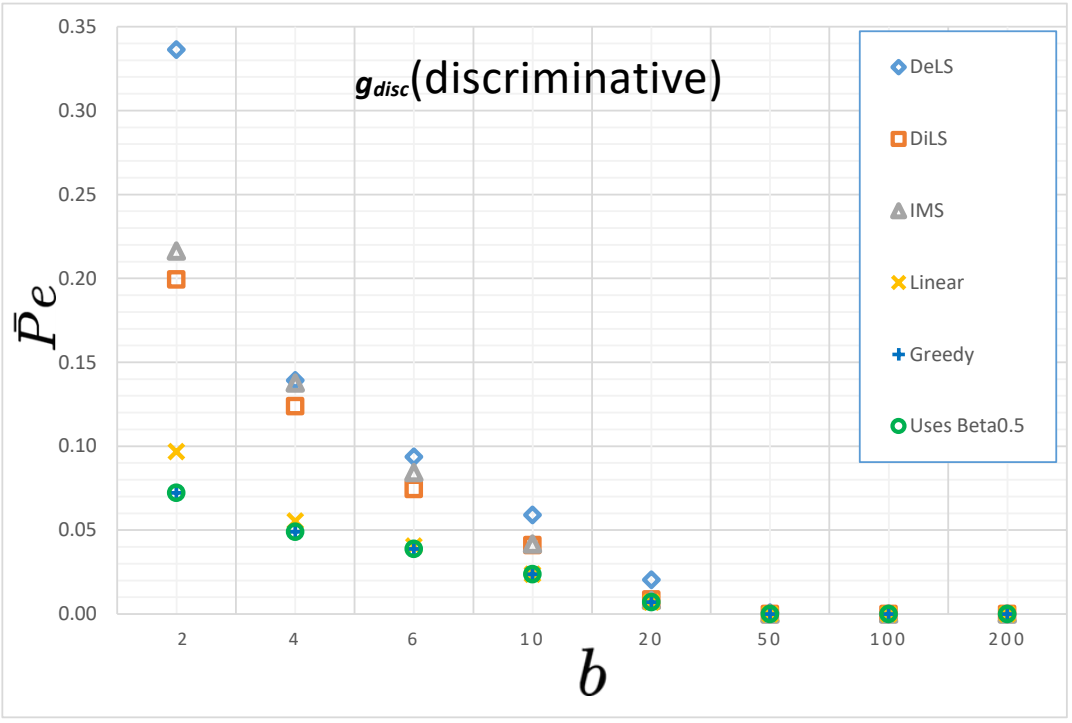

FIGURE 7.5: spreading strategies comparison in the *discriminative* case. Average probability of error under equal prior, $\overline{Pe}$, as a function of pooling bag size $b \in \mathcal{B}$ for an average payload 0.1 bptc for g_{disc} pooling function learnt over all the strategies.

In *Figure* 7.5, we report the detection of each spreading strategy with the *discriminative* g_{disc} pooling function. The best strategies are in the descending order, in the sense of its ability to resist the pooling function that tries to *discriminate* it, $DeLS$, IMS, $DiLS$, $Linear$, $Uses\text{-}\beta$, $Greedy$. $DeLS$ is again performing well in this *non-clairvoyant* approach, and it remains resistant until $b = 100$ where its average probability of error $\overline{Pe}$ start to coincide with those of the other strategies and becomes ≈ 0. $DeLS$ and IMS are more detected by the discriminative g_{disc} than the clairvoyant g_{clair}. Looking at the histograms of the SVM scores (*Figure* 7.7), we observe indeed a higher separation with g_{disc}. Looking to *Figure* 7.5, we also observe a smaller gap between all the $\overline{Pe}$ of each strategy, compared to *Figure* 7.4. Those behaviour are probably because the optimization (learning of the SVM) try to minimize the prediction error fairly for each of the strategies.

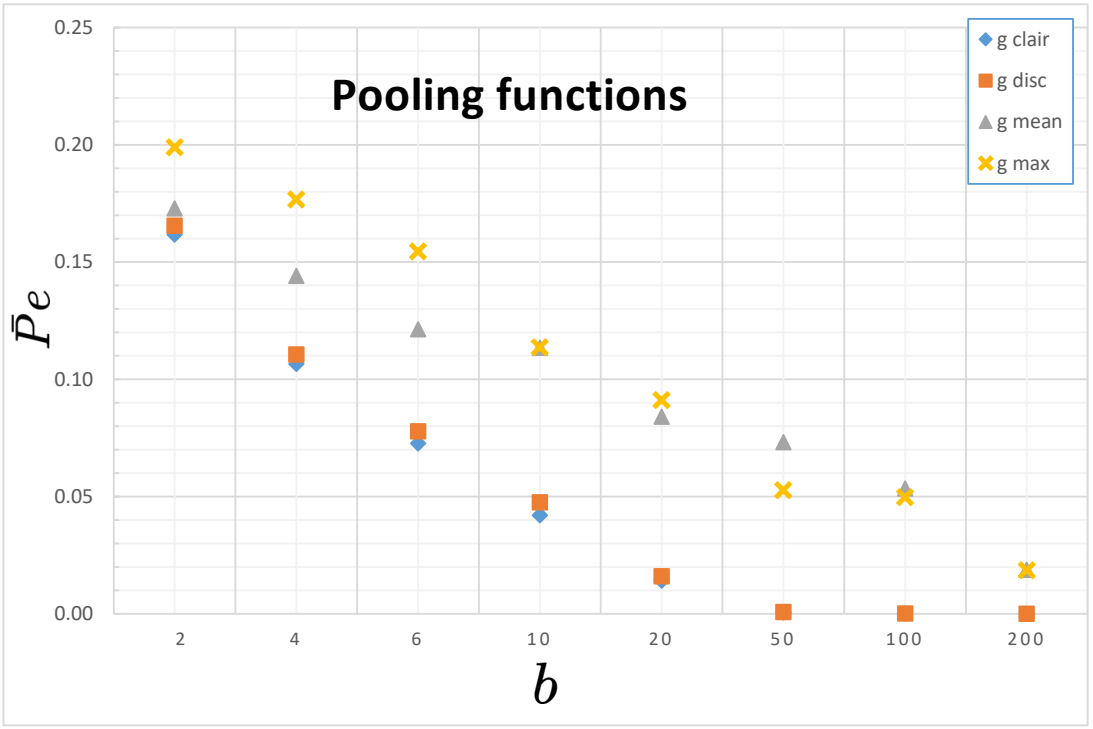

FIGURE 7.6: Pooled steganalysis comparison. Average probability of error under equal prior, $\overline{Pe}$, as a function of pooling bag size $b \in \mathcal{B}$ for an average payload 0.1 bptc. The average $\overline{Pe}$ is computed by testing each spreading strategy.

Finally, *Figure* 7.6 provides a useful insight regarding the accuracy of the pooling methods. This shows the *average* probability of error under equal prior, $\overline{Pe}$, as a function of pooling bag size $b \in \mathcal{B}$, with an average payload size $\bar{R} = 0.1$ bptc for each pooling function over all the strategies. We note that the g_{disc} pooling function outperforms g_{mean} and g_{max} functions with average difference of $\overline{Pe} \approx 2\%$ and is closer to g_{clair} with an average difference of $\overline{Pe} \approx 0.8\%$[2]. From *Figure* 7.6, we could observe that g_{disc} and g_{clair} are more stable than g_{mean} and g_{max}. This is in agreement with the hypothesis of this chapter which is that a *discriminative* pooling function allows obtaining better detection results compared to the mean and max pooling function.

[2]The variance on $\overline{Pe}$ for each pooling is around $\times 10^{-6}$.

7.6 Conclusion

In this chapter, we studied the problem of content-adaptive batch steganography and pooled steganalysis for a steganalyst unaware of the payload-spreading strategy and equipped with a single-image detector trained as a quantitative detector.

We studied the ability of the steganalyst to *discriminate* the spreading strategy, thanks to a pooling function that is able to recognize various stego patterns, and then able to pool the SID scores much cleverly than applying a simple average or maximum. Empirical results made with six different spreading strategies and a state-of-the-art Single Image Detector confirms that our *discriminative* pooling function can improve the accuracy of the pooled steganalysis. Our pooling function gets results close to a *clairvoyant* steganalyst which is assumed to know the spreading strategy.

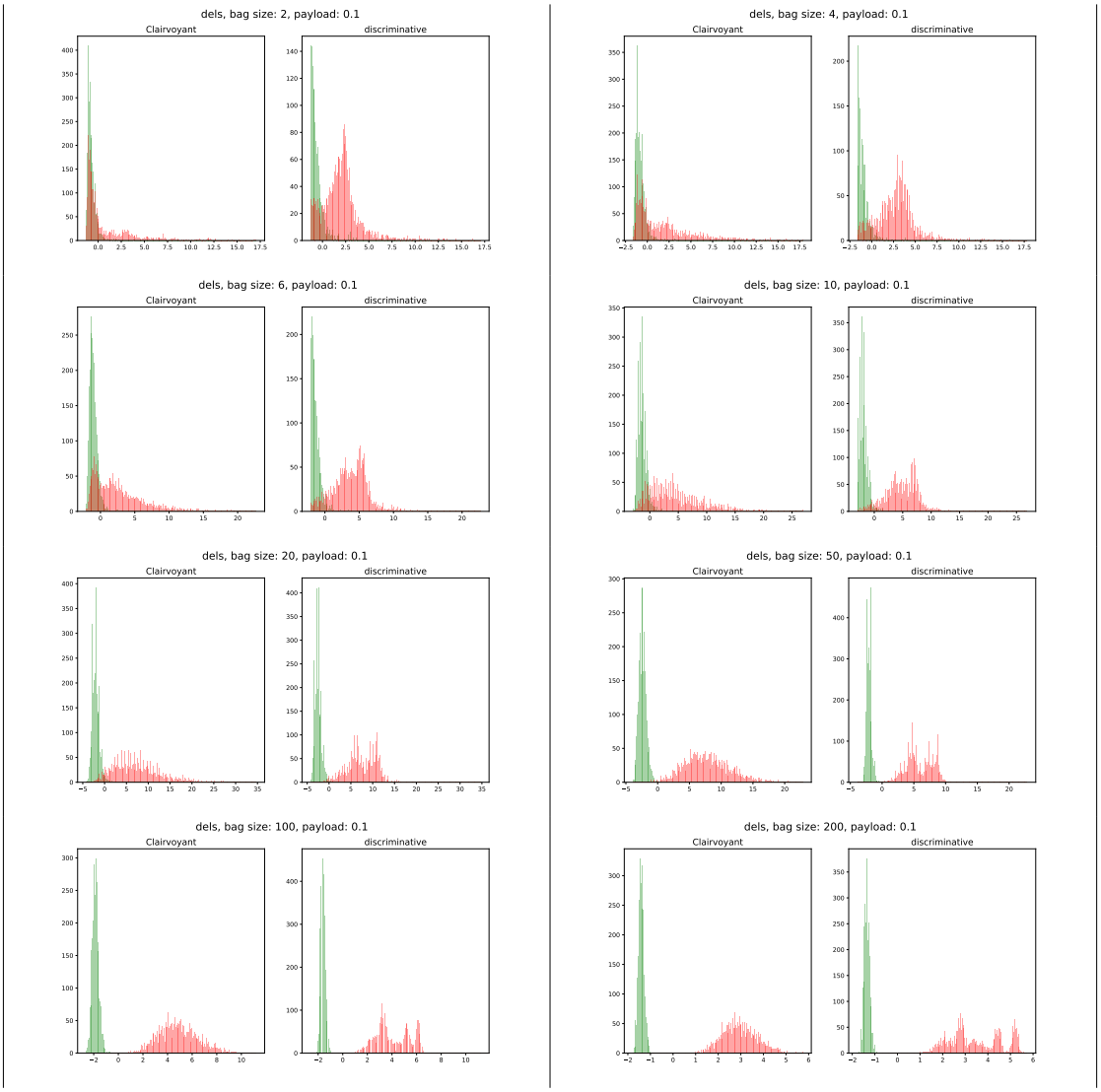

FIGURE 7.7: DeLS spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

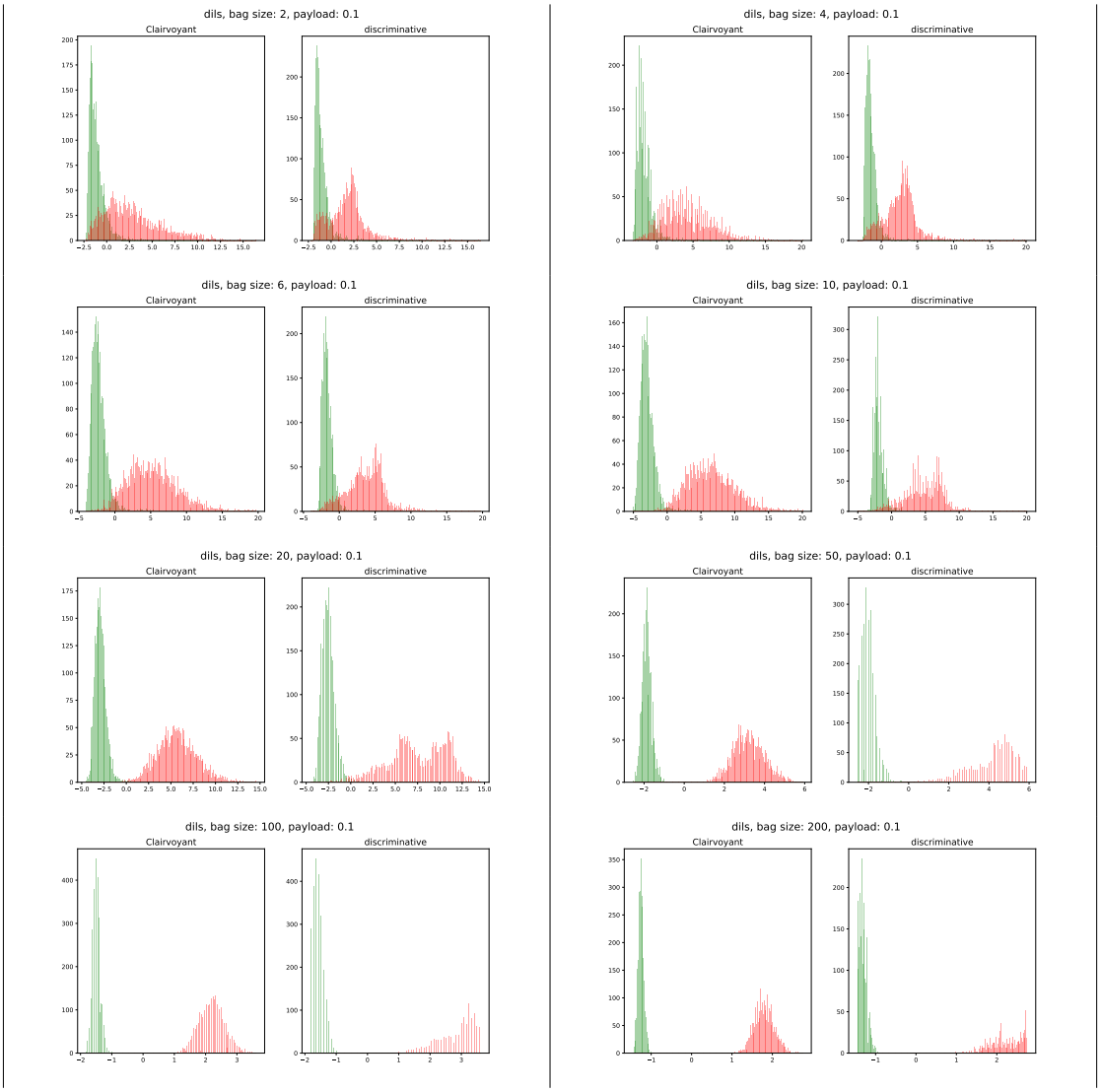

FIGURE 7.8: DiLS spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

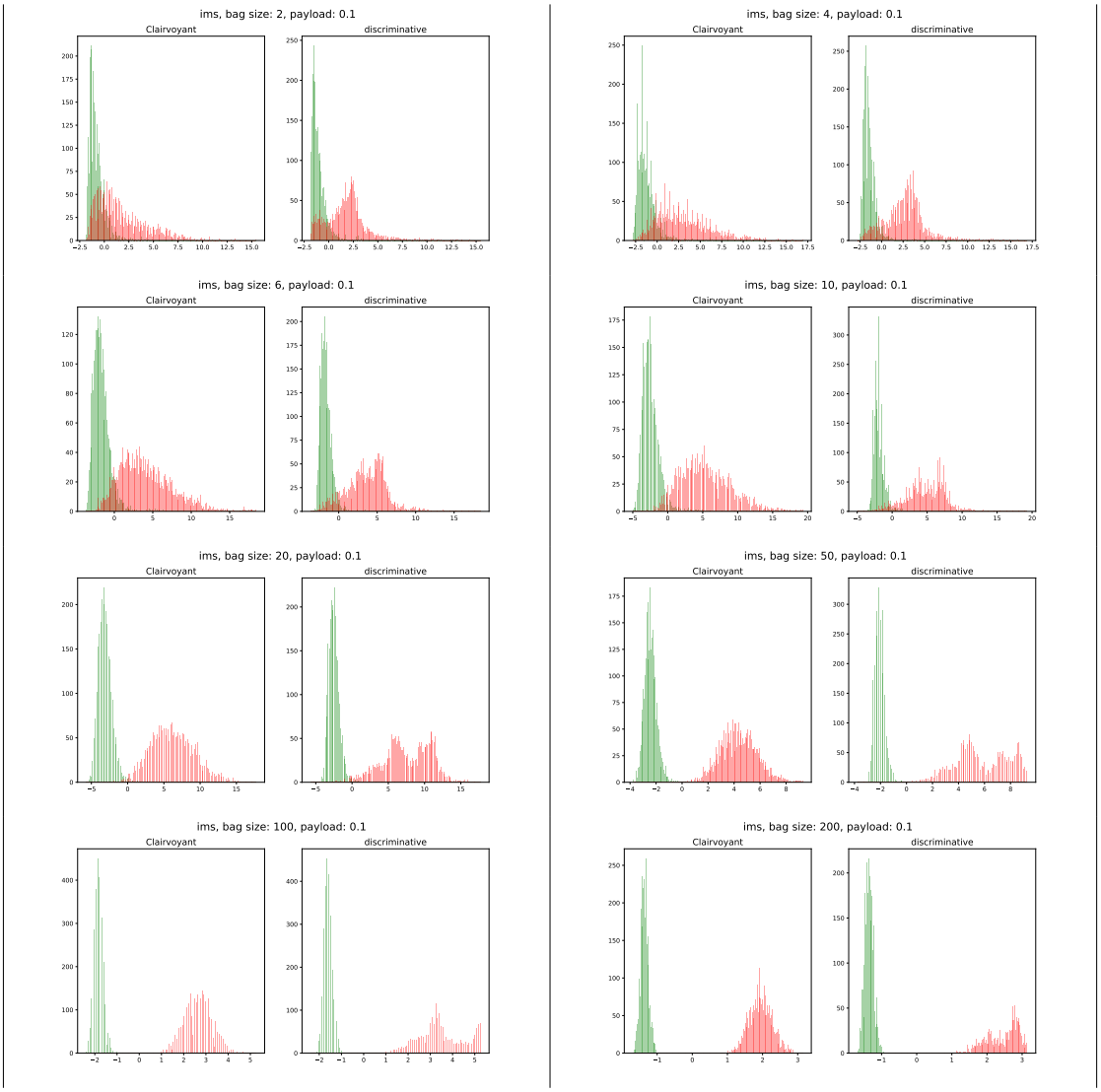

FIGURE 7.9: IMS spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

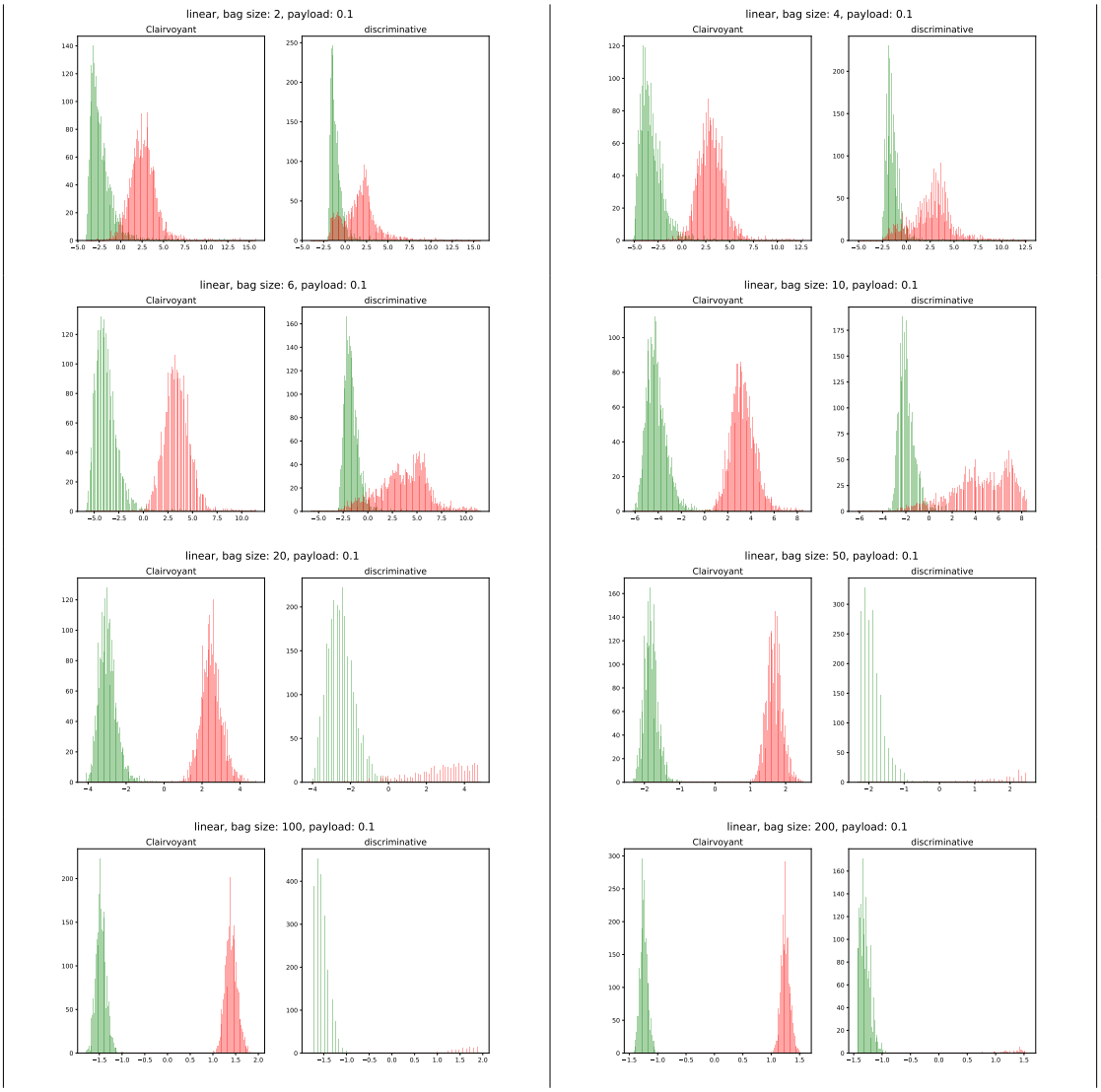

FIGURE 7.10: Linear spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

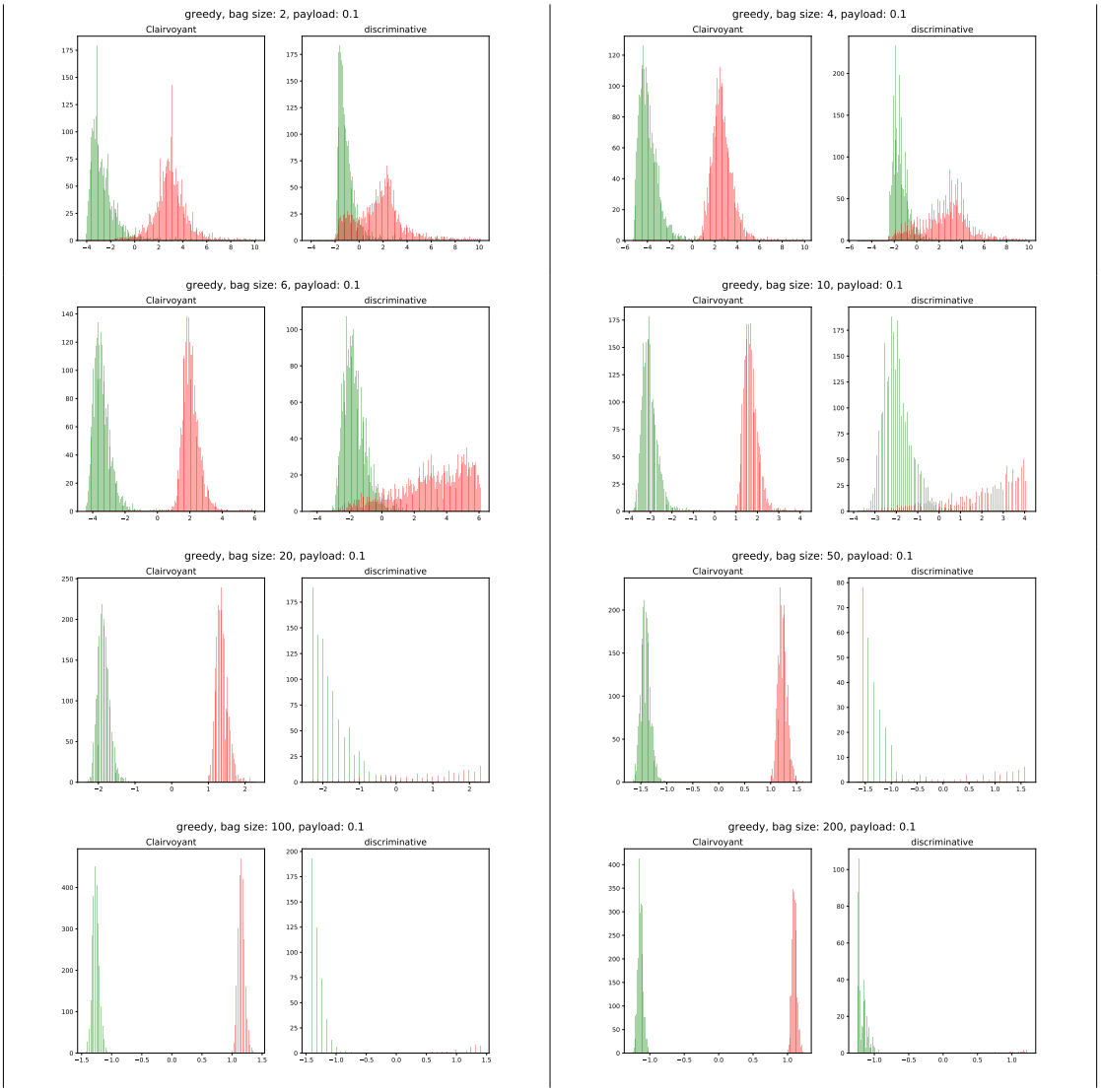

FIGURE 7.11: Greedy spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

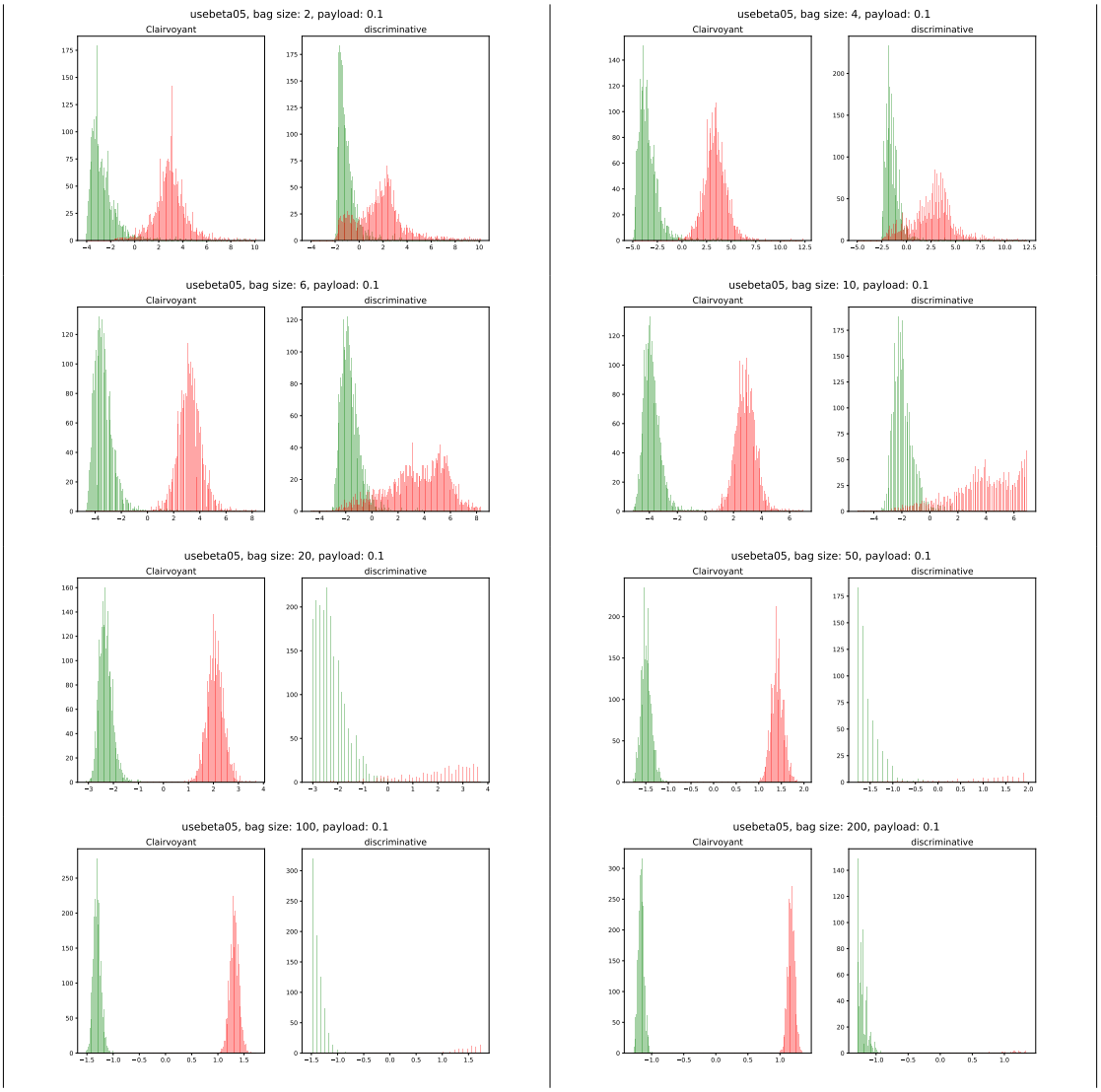

FIGURE 7.12: *Uses-β* spreading strategy: The histograms of the SVM scores (just before thresholding) for cover bags and stego bags, for clairvoyant pooling and for discriminative pooling.

Part IV

Conclusions And Perspectives

Conclusions

In this thesis, we studied the problem of batch steganography and pooled steganalysis in the case where the steganalyst is unaware of the payload-spreading strategy. This led to comparison of different single image detectors, spreading strategies and pooling functions. Furthermore, this led to a general architecture.

In our study on single image detectors, we made a comparison between quantitative steganalysis algorithms in order to decide which one is best suited for pooled steganalysis. We also proposed to extend this comparison to binary steganalysis algorithms. For this we propose a methodology to use both types of algorithms for both binary and quantitative steganalysis. We investigated two state-of-the-art steganalysis algorithms, QS and GLRT.

Numerical results based on J-UNIWARD embedding scheme, and the GFR Rich Model show that the GLRT approach is slightly better than the QS one when doing binary steganalysis, and that GLRT approach is marginally worse than the QS one, when doing the quantitative steganalysis. GLRT approach seems more accurate to estimate payload when it is small. This may be due to the accuracy of the original GLRT classifier which is good for small payload. Nevertheless, in average the QS algorithm was better then the GLRT algorithm.

Despite the broad difference between the binary and quantitative scenarios, the use of a specially developed algorithm for the binary scenario in the quantitative scenario, and vice versa, seems to be advantageous. Our results proved this, as the difference is 2% to 4% in Pe or $RMSE$.

In our study on pooled steganalysis, we use the most recent and generic approach in order to be closer to reality, i.e. to be operational. We therefore use the QS (quantitative steganalysis) algorithm in order to be insensitive to the size of the payload.

Our pooled steganalysis architecture is constructed to be able to *discriminate* the spreading strategy, thanks to a pooling function that is optimized on a set of spreading strategies. This pooling function is able to adapt to various stego patterns, and then able to aggregate the SID scores much cleverly than applying a simple average or maximum. Consequently, this allows to improve the accuracy of the pooled steganalysis.

Empirical results made with six different spreading strategies and a state-of-the-art Single Image Detector confirms that our *discriminative* pooling function can improve the accuracy of the pooled steganalysis. This pooling function gets results close to a *clairvoyant* steganalyst which is assumed to know the spreading strategy.

Perspectives And Open Issues

The results of this thesis open the door to future studies related to the single image detectors, the spreading strategies and the pooling functions.

The purpose of this thesis was to study scenarios of steganography, that are more realistic, less constraining, and closer to the real world.

More general and less constrained scenarios

Unknown bag's payload:

We have used our architecture only in the case of bags of a fixed payload size, but it could be generalised to deal with scenarios where the bag's payload is unknown. For this, we guess that training the architecture on bags with different bag's payload sizes may be an option since the architecture is not dependent on the number of images in a bag; it is in practice very easy to experiment.

A positive point is that this experiment would increase the number of learning samples. Thus, we would have **bags with multiple payloads.** To let the architecture deals with bags of different payload sizes, we could train on bags with different payload sizes instead of bags with 0.1 bptc. These payload sizes would be in the range of $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ bptc.

It is worth reflecting on the difficulty of this scenario. The lower the payload of the bag, the closer the Parzen histograms of the cover bags are to the Parzen histograms of the stego bags (regardless of the spreading strategy).

Unknown bag size:

We used our architecture with a fixed bag size, but thanks to Parzen's windows, the architecture is theoretically not related to the size of the bag. Thus we could have in this experiment, **bags with multiple sizes at a time**. In this scenario, the bag size is assumed to be unknown from the steganalyst. We could train our architecture on bags of sizes $\in \{2, 4, 6, 10, 20, 50, 100, 200\}$ images.

It is worth noting that this scenario will probably work well because the distribution of cover images and stego images will not change with bag size (perhaps these results will be better because more examples will lead to more accurate distributions and thresholds).

Further study on the single image detector

The experiments of *Chapter* 6 open doors for multiple questions and the possible use of machine learning algorithms for binary and quantitative steganalysis. The number of algorithms to compare was limited. It may be interesting to use this way of comparing algorithms developed for different scenarios to analyse new algorithms, especially the deep learning architectures as [Chen *et al.*, 2018] where scores given by the *VNet* [Chen *et al.*, 2017] classifier are used to train a regression model for payload estimation. More efficient algorithms have since been proposed such as, ReST-Net [Li *et al.*, 2018], Xu-Net-JPEG [Xu, 2017], SRNet [Boroumand *et al.*, 2019] and Low-Complexity-Net [Huang *et al.*, 2019], hence it is worthy to study them and find a way to improve the SID block of the pooled steganalysis architecture.

All integrated solution using Deep Learning

After publishing our work on *Chapter* 7, where we used classical machine learning approaches to build a pooled steganalysis architecture, we thought of rebuilding the same architecture but with blocks based on deep learning. We started some tests but we did not finish this work at the end of this thesis. It is postponed to future work.

Regarding the SID, we have worked on three possible architectures with methodologies close to the methodology of [Chen *et al.*, 2018], see *Section* 3.3.2 :

- In the first architecture, we built 10 pretrained CNN (Low-Complexity-Net [Huang *et al.*, 2019]) binary detectors LC_α trained on cover and stego images with a fixed payload $\alpha \in I$, $I = \{0.1, ..., 1\}$ as indicated in *Figure* 7.13. Next, we connect the 10×256 feature map to 3 fully connected layer, and a payload regressor, as shown in *Figure* 7.13. Finally, the regressor is trained on the concatenated features of stego images embedded with payloads α chosen uniformly randomly from some fixed interval I. The regression part is a three-layer fully connected neural network (FNN) with 10×256 neurons in each layer and an output neuron, which output the final payload estimation.

- In the second architecture, the same pretrained CNNs are used, but this time we extracted the features from the fourth layer and then added them to what what we call a diminution block. As shown in *Figure* 7.14, the subsampling block leads to obtain the same 10×256 concatenated features as in the first architecture. The resulted features are then connected to the FNN which is the same as the first architecture.

- The third architecture is the same as the first one but we used a reduced version of the Low-Complexity-Net (LC) CNN to build the 10 preformed CNNs, as shown in *Figure* 7.15.

Once the SID is built, we build the pooling architecture. For a bag of size b, we used b SIDs so that each image of the bag is given to a SID that will produce a

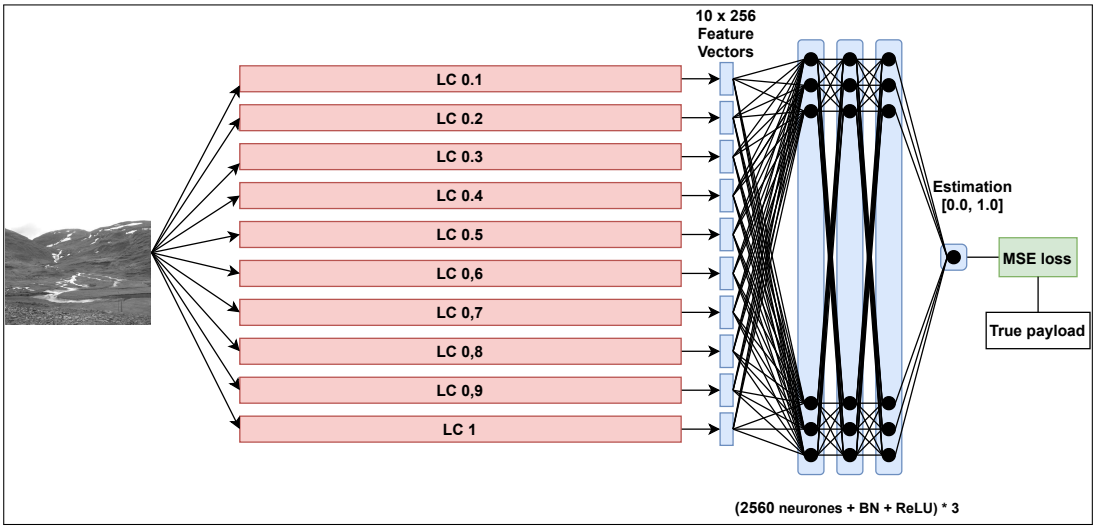

FIGURE 7.13: The first SID architecture.

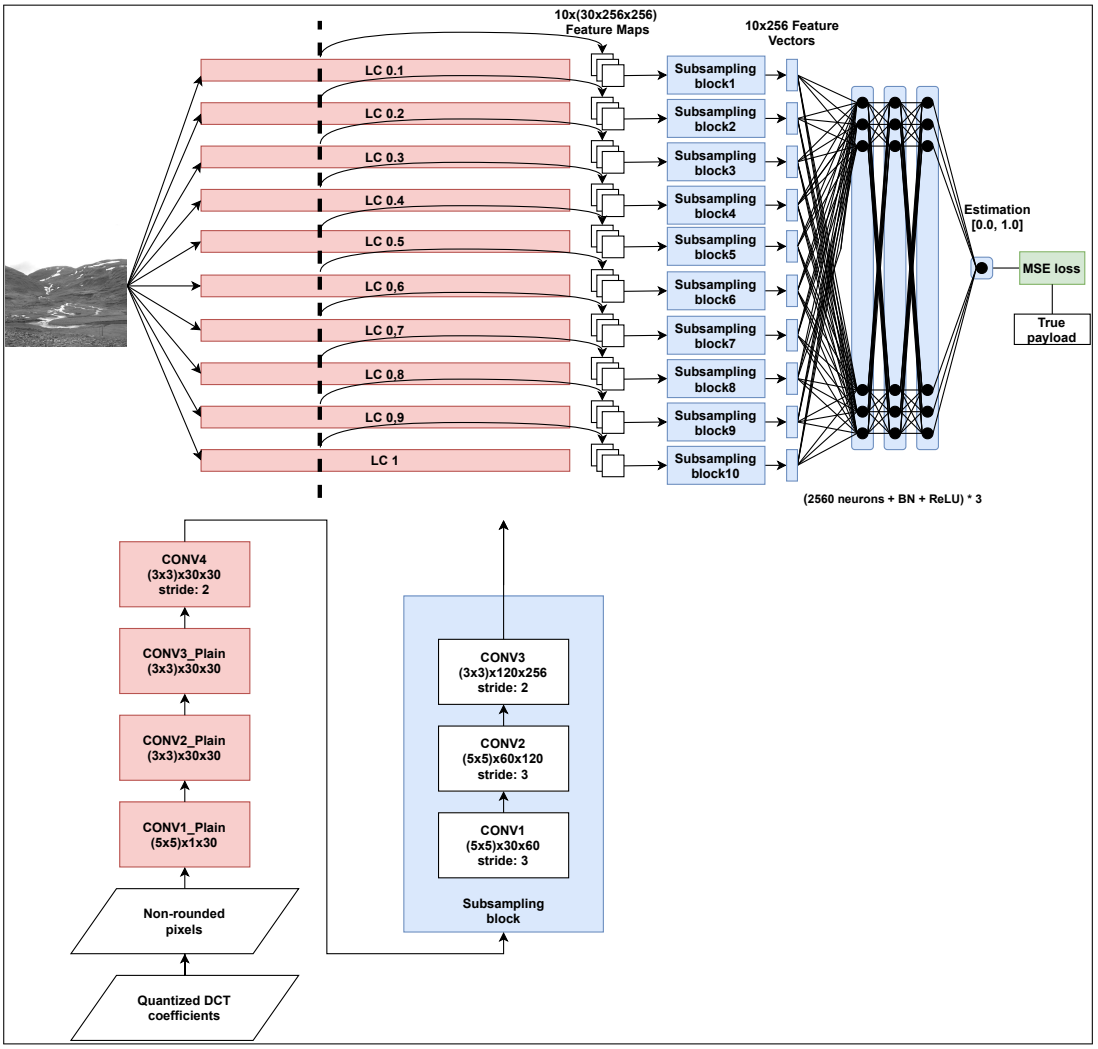

FIGURE 7.14: The second SID architecture.

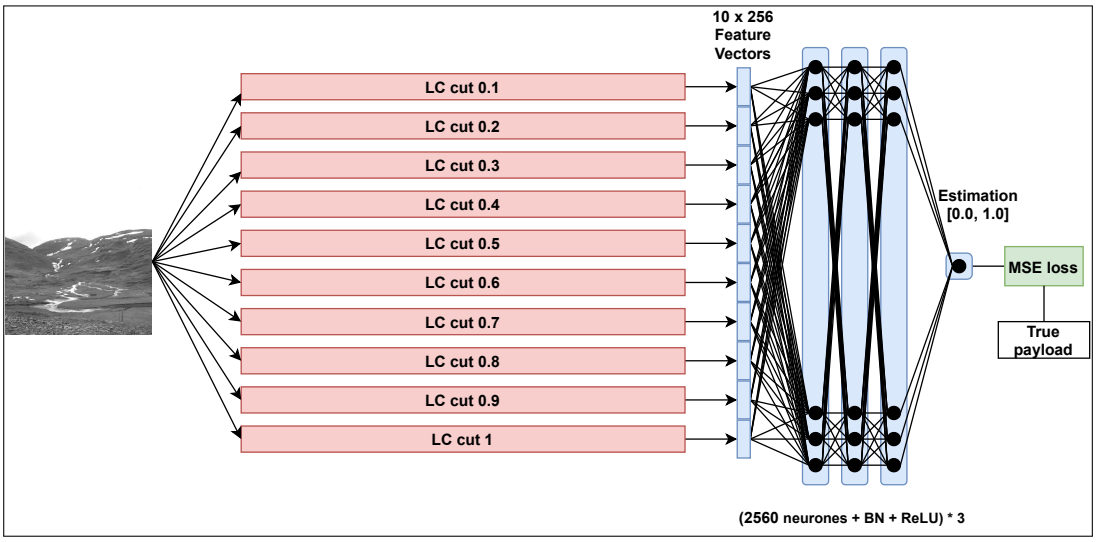

FIGURE 7.15: The third SID architecture.

score, as shown in *Figure* 7.16. The bag will give a vector of scores of the size b similar to the classical method explained in *Chapter* 5. The resulting vector is then given to a single-layer fully connected neural network (FNN) with 1024 neurons and two neurons corresponding to the number of classes of the network output. Finally, a softmax activation function is used to produce a distribution on the two class labels.

Developing SID using accurate models to obtain precise scores is a major problem, but extracting a score from an image causes limitations, as it causes a huge loss of information. We need to think about going further, instead of aggregating the scores of images (thus a scalar value for an image), it might be preferable to aggregate the feature vectors describing the images, thus a set of scalar values. Deep learning makes it possible. However, we need a sufficient large training sets, and use the whole image as input in a general deep learning architecture.

In general, an all integrated solution using Deep Learning benefits from the improvements in conventional steganalysis. One can think about using the most efficient state-of-the-art steganalysis networks. In addition, we could evaluate the transfer learning technique which can be used to handle stego images with a small payload and thus, stego bags with small payloads.

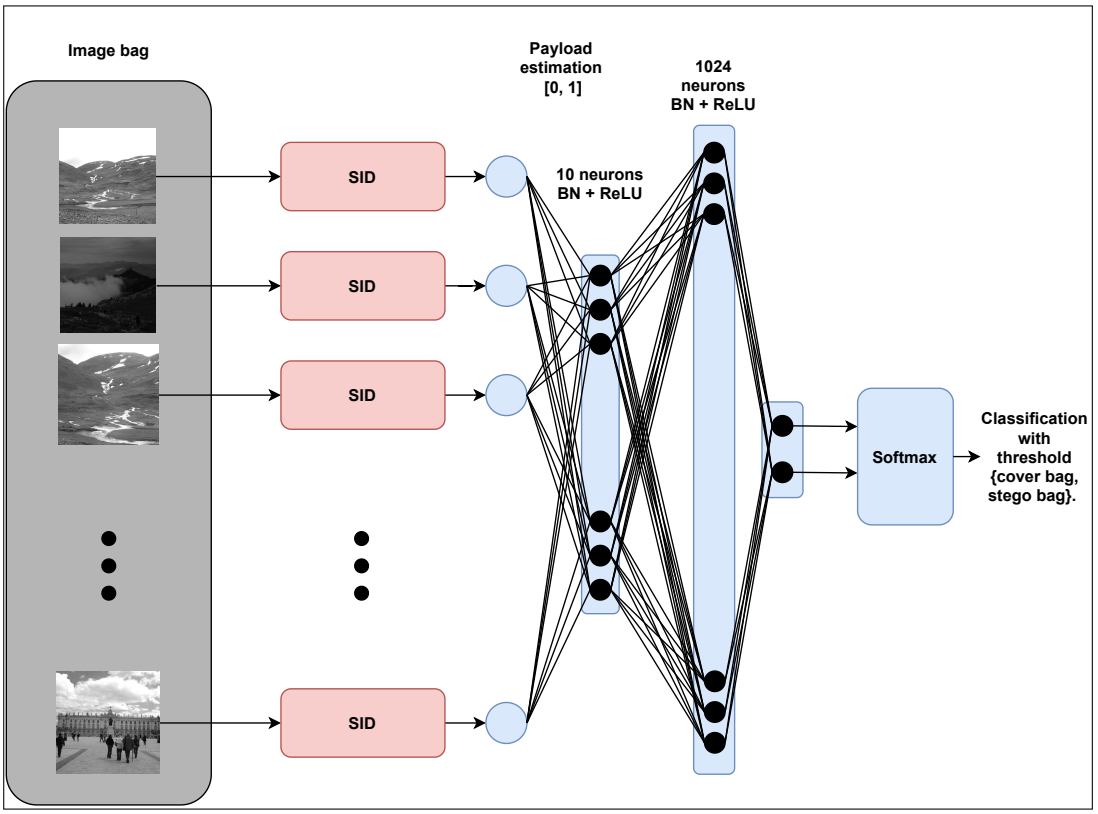

FIGURE 7.16: The overall pooled steganalysis architecture.

This type of approach poses many practical challenges, such as the computational power and memory resource, because this type of experiment requires many images to build the bags. Moreover, it is complicated for the moment to use a large sized bag of images as input of a neural network to process it.

Batch steganography using a 3 player game

With the advent of deep learning and more specifically Generative Adversarial Networks (GANs), new techniques have emerged. Among these techniques the three-player game approach [Yedroudj *et al.*, 2019], where three networks compete against each other.

The first attempt to hide data using the three-player game approach is in the architecture GSIVAT [Hayes and Danezis, 2017] and then in HiDDEeN [Zhu *et al.*, 2018]. These two articles used 3 CNNs, iteratively updated, and playing the roles

of Alice, Bob, and Eve. Unfortunately, the security notions and their evaluation are not treated correctly. When Eve is clairvoyant, these two approaches are very detectable. Recently, in [Yedroudj *et al.*, 2019], the authors discussed the GSIVAT and HiDDEeN architectures and defined the general concept and how to correctly use it for steganography. Moreover, the authors proposed three architectures based on the 3 player game approach.

One could think of extending the approach of batch steganography to a game theory, or practical GAN simulation using the 3 player game approach, which has never been studied before. In this approach we could simulate the operations of Alice doing the batch steganography, Bob extracting the messages without any errors, and Eve the steganalyst trying to prevent communication between Alice and Bob.

Dealing with the spreading-strategy mismatch issue

One of the most challenging issue in the steganalysis domain is the *source* mismatch, whether the *cover-source mismatch* or the *stego-source mismatch*, which lead to inaccurate predictions. These notions are defined as follows:

- **A source [Giboulot *et al.*, 2020]:** "*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*".
- **Cover-source mismatch (CSM) [Giboulot *et al.*, 2020]:** "*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."

– **Stego-source mismatch (SSM) [Lerch-Hostalot and Megías, 2019]:** "*stego source mismatch occurs when some embedding parameter, such as the exact payload, differ between the training and the testing datasets.*"

In the case of batch steganography and pooled steganalysis, the message is spread into a bag of images. In this case, more questions can be asked about a strategy never seen before by Eve, assuming that Alice uses images from the same distribution, i.e. there is no cover-source mismatch. We call this problem the spreading-strategy mismatch.

Dealing with the spreading-strategy mismatch issue:

In this scenario, our pooled steganalysis architecture should deal with a spreading strategy never seen before, considering that Alice uses images from the same distribution, i.e. there is no cover-source mismatch. This is closely related to stego-source mismatch.

One possible solution is to generate random strategies that are a mixture of the basic strategies. This way, a stego bag is constructed using multiple strategies and then fed to the pooled steganalysis architecture to be learned. This possible, since we currently have a large number of different families of strategies and it is possible that our approach is already working well when tested on strategies never seen before.

Extension of batch steganography and pooled steganalysis to other applications

- **Color image steganography:** Batch steganography consists of spreading a message over several images. This approach can be extended to conventional steganography in color images. Note that in this thesis, we worked on grayscale images only. One can think about considering chrominance and luminance channels as separate images, apply spreading strategy and an embedding scheme to spread the message over the channels. As in the beginning of this thesis, we have worked on the problem of steganography in color images [Ndiaye *et al.*, 2017], we can think about extending this work and applying some spreading strategies in the problem of color image steganography. The benefits of this approach is that we already examined multiple spreading strategies, and we know which one is the best to use for JPEG images (see *Chapter* 7). Moreover, a new study [Cogranne *et al.*, 2020b] has been performed to address the use of batch spreading strategies for embedding in color channels. The results of this study opens the door to several questions and improvements that would be very interesting to be addressed in future works.

- **Video steganalysis:** Another interesting extension is the generalization of steganalysis applied to videos through the use of convolutional 3d networks that takes the timeline aspect. This poses an architectural challenge to generalize the algorithms used until now in 2D (Images) to move to 3D (Videos). In addition, this type of architecture requires a very powerful computing capacity.

Further studies on the spreading strategies:

- **The DeLS strategy:** As mentioned in *Chapter* 7, *Section* 7.3.3, we used the MiPOD embedding scheme to calculate the deflection coefficient, but

the underlying hypothesis was considering a spatial domain. Hence, we were obliged to make some additional adaptations for our experiments when working in the JPEG (frequential) domain. We applied the MiPOD on de-quantified rounded images from our JPEG database, and we recalculated the payload size in bits per coefficient (bpc) instead of bits per non-zero AC DCT coefficient (bpnzAC) in order to guarantee the same message length to be embedded.

What about using the MiPOD embedding scheme to calculate the deflection coefficient directly from JPEG images, as in the JPEG domain? The quality factor is a crucial constraint that cannot be avoided and does not exist in the spatial domain, hence it may affect the the deflection coefficient. Furthermore, as we work on content-adaptive embedding scheme (J-UNIWARD), due to coding techniques, the change rate of embedding may differ between the spatial domain and the JPEG domain for a given payload size.

Now that a version of MiPOD applicable to JPEG images (J-MiPOD) is available [Cogranne *et al.*, 2020b], it would be very interesting to work on it and to study the difference in efficiency between the DeLS spreading strategy with deflection coefficients calculated in the spatial domain (de-quantified rounded images, our approach) and those calculated in the DCT domain (rounded quantized DCT coefficients) proposed by the authors.

Part V

List Of Publications

"Pooled Steganalysis In JPEG: How To Deal With The Spreading Strategy?"

Citation:

[Zakaria *et al.*, 2019a] Zakaria, Ahmad; Chaumont, Marc; Subsol, Gérard: Pooled Steganalysis in JPEG: how to deal with the spreading strategy? In: *IEEE International Workshop on Information Forensics and Security, WIFS2019*, Delft, The Netherlands, December 9-12, 2019, IEEE, 2019, S. 1–6. Acceptance rate = 30% – DOI https://doi.org/10.1109/WIFS47025.2019.9035096.

Abstract:

In image pooled steganalysis, a steganalyst, Eve, aims to detect if *a set* of images sent by a steganographer, Alice, to a receiver, Bob, contains a hidden message. We can reasonably assess that the steganalyst does not know the strategy used to spread the payload across images. To the best of our knowledge, in this case, the most appropriate solution for pooled steganalysis is to use a Single-Image Detector (SID) to estimate/quantify if an image is cover or stego, and to average the scores obtained on the *set* of images.

In such a scenario, where Eve does not know the embedding strategies, we experimentally show that if Eve can *discriminate* among few well-known embedding strategies, she can improve her steganalysis performances compared to a simple averaging or maximum pooled approach. Our *discriminative* approach allows obtaining steganalysis efficiencies comparable to those obtained by a clairvoyant, Eve, who knows the Alice embedding strategy. Another interesting observation is that DeLS embedding strategy behaves really better than all the other embedding strategies.

Those observations results in the experimentation with six different embedding strategies made on Jpeg images with J-UNIWARD, a state-of-the-art Single-Image-Detector, and a *discriminative* architecture that is invariant to the individual payload in each image, invariant to the size of the analyzed set of images,

and build on a binary detector (for the pooling) that is able to deal with various embedding strategies.

"Quantitative And Binary Steganalysis In JPEG: A Comparative Study"

Citation:

[Zakaria *et al.*, 2018] Zakaria, Ahmad ; Chaumont, Marc ; Subsol, Gérard: Quantitative and Binary Steganalysis in JPEG: A Comparative Study. In: *26th European Signal Processing Conference, EUSIPCO 2018, Roma, Italy, September 3-7*, 2018, URL https://doi.org/10.23919/EUSIPCO.2018. 8553580, 2018, S. 1422–1426

Abstract:

We consider the problem of steganalysis, in which Eve (the steganalyst) aims to identify a steganographer, Alice who sends images through a network. We can also hypothesise that Eve does not know how many bits Alice embed in an image. In this paper, we investigate two different steganalysis scenarios: Binary Steganalysis and Quantitative Steganalysis. We compare two classical steganalysis algorithms from the state-of-the-art: the QS algorithm and the GLRT-Ensemble Classifier, with features extracted from JPEG images obtained from BOSSbase 1.01. As their outputs are different, we propose a methodology to compare them. Numerical results with a state-of-the-art Content Adaptive Embedding Scheme and a Rich Model show that the approach of the GLRT-ensemble is better than the QS approach when doing Binary Steganalysis but worse when doing Quantitative Steganalysis.

"Stéganalyse Groupée En JPEG : Comment Gérer La Stratégie d'Étalement?"

Citation:

[Zakaria *et al.*, 2019b] Zakaria, Ahmad ; Chaumont, Marc; Subsol, Gérard: Stéganalyse groupée en JPEG: comment gérer la stratégie d'étalement? In: *GRETSI'2019, XXVIIème Colloque francophone de traitement du signal et des images* (2019). – URL http://www.lirmm.fr/~chaumont/publications/GRETSI_2019_ZAKARIA_CHAUMONT_SUBSOL-Pooled_Steganalysis.pdf

Résumé:

Dans la Stéganalyse Groupée d'images (SG), une stéganalyste (Eve) vise à détecter si un ensemble d'images, envoyées par une stéganographe (Alice) à un pair (Bob) via un réseau, contient un message caché. Nous pouvons raisonnablement supposer qu'Eve ne connaît pas la stratégie exacte utilisée pour insérer et étaler le message à travers les images, mais elle devine qu'Alice et Bob utilisent un certain algorithme stéganographique. Au meilleur de notre connaissance, dans ce cas, la solution la plus appropriée pour la SG est d'utiliser un détecteur d'image unique (SID) pour estimer si une image est stéganographiée ou non et de faire la moyenne des scores obtenus sur un ensemble d'images. Et si Eve pouvait prendre en compte la stratégie d'étalement? Pourrait-elle alors utiliser un algorithme de SG meilleur que la moyenne des scores? Dans cet article, nous proposons une architecture générale de SG prenant en compte différentes stratégies d'étalement. Les résultats expérimentaux obtenus avec six stratégies et un détecteur d'image unique montrent que, si Eve discrimine la stratégie d'étalement, elle peut améliorer la précision de la SG.

"Stéganographie Et Stéganalyse Des Images JPEG Couleur"

Citation:

[Ndiaye *et al.*, 2017] Ndiaye, Papa; Chaumont, Marc; Yedroudj, Mehdi ; Zakaria, Ahmad: Stéganographie et Stéganalyse des images JPEG Couleur. In: *CORESA'2017, COmpression et REprésentation des Signaux Audiovisuels, Caen, France*, 20-21 septembre (2017). http://www.lirmm.fr/~chaumont/publications/CORESA-2017-NDIAYE-CHAUMONT-YEDROUDJ-ZAKARIA-SteganographieCouleur.pdf

Résumé:

JPEG est aujourd'hui le format d'image le plus couramment utilisé pour l'échange d'images. Bien que cela en fasse un standard naturel pour la stéganographie moderne, il n'en demeure pas moins qu'il n'y a pas de contributions pour l'insertion dans des images JPEG en couleur. Les approches d'insertion considèrent en effet uniquement l'insertion dans une image JPEG en niveaux de gris, principalement parce que l'insertion dans des images en niveau de gris est déjà un problème difficile. Dans cet article, nous étudions, de manière pratique, la question de l'insertion dans une image JPEG couleur. La question principale consiste à déterminer comment doit être effectuée la répartition du message, c'est-à-dire des bits à insérer, entre les composantes de couleurs (Y, C_r, C_b) qui ont été quantifiées. Après avoir rappelé l'état de l'art, nous donnons de premiers résultats expérimentaux indiquant que l'insertion doit principalement être effectuée dans la composante de luminance.

Appendix

A Appendix: shift hypothesis

The shift hypothesis was first assumed by [Ker, 2006] in order to reduce the analyses of pooled steganalysis to tractable problems. The author assumes that Eve possesses a quantitative detector that estimates the payload in an individual stego image. Eve aims to detect the presence of a message in a set of images by combining the payloads statistics, i.e. the results of the quantitative detector on each image in the set.

The author writes ψ_p for the density function of the quantitative detector, when the payload p is embedded in a cover image; Eve expects that despite random errors, if the detector is an accurate one, if she averages the outputs of the detector, she gets the right p.

The shift hypothesis is that $\psi_p(\mathbf{x}) = \psi_0(\mathbf{x} - p)$, i.e. the shape of the density function is the same whatever is the payload. For a cover image (0), it will be centered in 0 whereas for a payload p, it will be shifted and centered in p.

In *Figure* 17, two histograms from [Ker, 2006] are displayed for two quantitative detectors, the Sample Pair Analysis (SPA) detector from [Dumitrescu *et al.*, 2002] and the WS detector from [Fridrich and Goljan, 2004]. Those detectors are mentioned in *Section* 3.1.5. Ker has built the histograms to represent the observed estimate of the detector for a set of 3,000 images, with the experiments being repeated at 10 payloads. The shift hypothesis is approximately valid for the SPA detector, for payloads less than 0.8, but there is both a shape change and a negative bias for larger payloads. For the WS detector, the offset assumption seems less relevant, but for average values of p, there is still evidence of a constant distribution shape. Ker considers these histograms as an evidence that one should be able to develop detectors that are not far from satisfying the shift hypothesis.

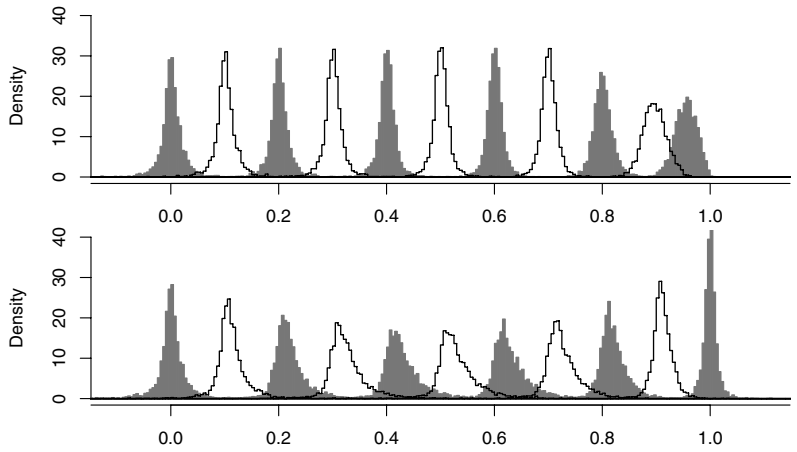

FIGURE 17: Two histograms for two quantitative detectors, the Sample Pair Analysis (SPA) detector from [Dumitrescu *et al.*, 2002] and the WS detector from [Fridrich and Goljan, 2004]. The first histogram is for the covers (on the zero dot) and the others are for the stego images. The black and white colors are just for the visual purposes [Ker, 2006].

B Appendix: ROC curve algorithm

In our experiments, the *Receiver Operating Characteristic* (ROC) curve is built by using the algorithm presented in [Fridrich, 2009]. If we take back all the framework of *Chapter* 3, we get a set of real values f_c and f_s, which are a sample of two one-dimension distributions, over the set of real numbers, P_c and P_s which are modeling the distributions of cover and stego images.

In practice, the underlying distributions of cover images P_c, and stego images P_s, (see *Section* 3.1.3) are usually obtained experimentally in a sampled form. For one-dimensional distributions over the set of real numbers, and based on the shift hypothesis (see *Appendix* A), P_s is assumed to be a shifted version of P_c and both distributions are unimodal. In this case, the critical region is determined by a scalar threshold τ. Hence, one method to draw a ROC curve is to directly moving the threshold τ from $-\infty$ to $+\infty$ and computing the probability of false alarm P_{fa} and the probability of correct detection P_{cd} of the cover/stego detection for each threshold τ. This ROC curve will allow one to define a classification threshold according to some objectives like minimizing the probability of error P_e explained in *Section* 3.1.4.

Moreover, the ROC can be drawn by fitting a parametric model through the sample distribution [Fridrich, 2009]. This procedure is applied in Algorithm 2, it will allow drawing the ROC curve and make comparison with another without focusing on the values of the threshold.

Algorithm 2 Drawing a ROC curve for one-dimensional features $f_s[i]$ (stego) and $f_c[i]$ (cover), f can be an output of a binary detector before thresholding or payload estimation from a quantitative detector. $i = 1, ..., k$, computed from k cover and k stego images.

f = **sort(**$f_s \cup f_c$**);**
// f is the set of all features sorted to form a non-decreasing sequence
$P_{fa}[0] = 1; \quad P_{cd}[0] = 1$
for $i = 1$ **to** $2k$ **do**
    if $f[i] \in f_c$ **then**
        $P_{fa}[i] = P_{fa}[i-1] - 1/k; P_{cd}[i] = P_{cd}[i-1]$;
    else
        $P_{cd}[i] = P_{cd}[i-1] - 1/k; P_{fa}[i] = P_{fa}[i-1]$;
    end if
    DrawLine($(P_{fa}[i-1], P_{cd}[i-1]), (P_{fa}[i], P_{cd}[i])$**);**
end for

C Appendix: Gabor Filters

[Gabor, 1946] first defined elementary signals as optimal signal carriers in communications. Those signals, called 1D Gabor signals, are represented as a modulation product of a sinusoidal wave of frequency ω_x and a Gaussian envelope of duration σ occurring at epoch x_o:

$$s(x) = exp[-\frac{1}{2}(\frac{x - x_0}{\sigma})^2 + j\omega_x x]. \tag{4}$$

The 2D Gabor filter technique was proposed by [Daugman, 1985]. The main purpose is to make an elliptic generalization of 1D elementary Gabor functions to obtain 2D spatial weighting functions generated by second order exponential bi-variate polynomials with complex coefficients.

Among many Gabor functions used to construct 2D Gabor filters, we cite those related to the use of rich model based steganalysis. In [Grigorescu *et al.*, 2002; Song *et al.*, 2015], the authors use the following bank of Gabor functions

$$f_{\lambda,\Theta,\varphi}(x,y) = e^{-((x'^2+\gamma^2 y'^2)/2\sigma^2)} \cos\left(2\pi\frac{x'}{\lambda}+\varphi\right) \tag{5}$$

where

$$x' = x\cos\Theta + y\sin\Theta,$$

$$y' = -x\sin\Theta + y\cos\Theta.$$

The parameters of *Equation* (5) are explained below:

- σ represents the scale parameter, the small value of σ means high spatial resolution, the image filtering coefficients reflect local properties in fine scale, while the large value of σ means low spatial resolution, the coefficients reflect local properties in coarse scale. σ is fixed to 0.56λ in [Grigorescu *et al.*, 2002; Song *et al.*, 2015].
- Θ specifies the orientations of 2D Gabor filters.
- λ denotes the wavelength of the cosine factor.
- γ is the spatial aspect ratio and specifies the ellipticity of Gaussian factor. γ is fixed to 0.5 in [Grigorescu *et al.*, 2002; Song *et al.*, 2015].
- φ specifies the phase offset of the cosine factor ($\varphi = 0$, π correspond to symmetric "center-on" functions, while $\varphi = \pi/2$, $\pi/2$ correspond to anti-symmetric functions). Note that [Song *et al.*, 2015] use only symmetric functions, i.e. $\varphi = 0$.

These spatial weighting functions are used to produce a family of optimal 2D filters, the 2D Gabor filters, which allow to reach the theoretical lower limit for the joint entropy of 2D spatial orientation, spatial frequency and spatial position. The 2D Gabor filtering are described as follows:

Given an image I, a set of image points Ω, $(x, y) \in \Omega$ and a 2D Gabor function f. The Gabor feature image $u(x, y)$ is obtained by convolving the $I(x, y)$ with $f(x, y)$ in the following equation:

$$u(x, y) = \int\int_{\Omega} I(\xi, \eta) f(x - \xi, y - \eta) d\xi d\eta \tag{6}$$

Bibliography

Bibliography

[Anderson and Petitcolas 1998] Anderson, Ross J. ; Petitcolas, Fabien A. P.: On the limits of steganography. In: *IEEE Journal on Selected Areas in Communications* 16 (1998), Nr. 4, S. 474–481. – URL https://doi.org/10.1109/49.668971

[Augot *et al.* 2011] Augot, Daniel ; Barbier, Morgan ; Fontaine, Caroline: Ensuring Message Embedding in Wet Paper Steganography. In: *Cryptography and Coding - 13th IMA International Conference, IMACC 2011, Oxford, UK, December 12-15, 2011. Proceedings*, URL https://doi.org/10.1007/978-3-642-25516-8_15, 2011, S. 244–258

[Babington 1996] Babington, Peter: *Herodotus : the histories*. London, Eng. : Penguin Books, 1996. – New edition. – ISBN 999819905002121

[Bamatraf *et al.* 2011] Bamatraf, Abdullah ; Ibrahim, Rosziati ; Salleh, Mohd. Najib M.: A New Digital Watermarking Algorithm Using Combination of Least Significant Bit (LSB) and Inverse Bit. In: *Journal of Computing (CoRR)* volume 3 (2011). – URL http://arxiv.org/abs/1111.6727

[Bas *et al.* 2011a] Bas, Patrick ; Filler, Tomás ; Pevný, Tomás: "Break Our Steganographic System": The Ins and Outs of Organizing BOSS. In: *Information Hiding - 13th International Conference, IH 2011, Prague, Czech Republic, May 18-20, 2011, Revised Selected Papers*, URL https://doi.org/10.1007/978-3-642-24178-9_5, 2011, S. 59–70

[Bas *et al.* 2011b] Bas, Patrick ; Filler, Tomás ; Pevný, Tomás: "Break Our Steganographic System": The Ins and Outs of Organizing BOSS. In: *Information Hiding - 13th International Conference, IH 2011, Prague, Czech Republic, May 18-20, 2011, Revised Selected Papers*, URL https://doi.org/10.1007/978-3-642-24178-9_5, 2011, S. 59–70

[Bierbrauer 2001] Bierbrauer, Jürgen: Crandall's problem. In: *Department of Mathematical Sciences, Michigan Technological University, HOUGHTON (MI)* (2001). – URL http://www.ws.binghamton.edu/fridrich/covcodes.pdf

[Boroumand *et al.* 2019] Boroumand, Mehdi ; Chen, Mo ; Fridrich, Jessica J.: Deep Residual Network for Steganalysis of Digital Images. In: *IEEE Trans. Information Forensics and Security* 14 (2019), Nr. 5, S. 1181–1193. – URL https://doi.org/10.1109/TIFS.2018.2871749

[Boroumand and Fridrich 2017] Boroumand, Mehdi ; Fridrich, Jessica J.: Nonlinear Feature Normalization in Steganalysis. In: *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2017, Philadelphia, PA, USA, June 20-22, 2017*, URL https://doi.org/10.1145/3082031.3083239, 2017, S. 45–54

[Brassil *et al.* 1995] Brassil, Jack ; Low, Steven ; Maxemchuk, Nicholas ; O'Gorman, Larry: Hiding information in document images. In: *Proc. Conf. Information Sciences and Systems (CISS-95)*, 1995, S. 482–489

[Butora *et al.* 2020] Butora, Jan ; Yousfi, Yassine ; Fridrich, Jessica J.: Turning Cost-Based Steganography into Model-Based. In: Riess, Christian (Hrsg.) ; Schirrmacher, Franziska (Hrsg.) ; Amerini, Irene (Hrsg.) ; Bestagini, Paolo (Hrsg.) ; Pevný, Tomás (Hrsg.): *IH&MMSec '20: ACM Workshop on Information Hiding and Multimedia Security, Denver, CO, USA, June 22-24, 2020*, ACM, 2020, S. 151–159. – URL https://doi.org/10.1145/3369412.3395065

[Cachin 1998] Cachin, Christian: An Information-Theoretic Model for Steganography. In: *Information Hiding, Second International Workshop, Portland, Oregon, USA, April 14-17, 1998, Proceedings*, URL https://doi.org/10.1007/3-540-49380-8_21, 1998, S. 306–318

[Cancelli *et al.* 2008] Cancelli, Giacomo ; Doërr, Gwenaël J. ; Barni, Mauro ; Cox, Ingemar J.: A comparative study of +/- steganalyzers. In: *International Workshop on Multimedia Signal Processing, MMSP 2008, October 8-10, 2008, Shangri-la Hotel, Cairns, Queensland, Australia*, IEEE Signal Processing Society, 2008, S. 791–796. – URL https://doi.org/10.1109/MMSP.2008.4665182

[Chang *et al.* 2002] Chang, Chin-Chen ; Chen, Tung-Shou ; Chung, Lou-Zo: A steganographic method based upon JPEG and quantization table modification. In: *Inf. Sci.* 141 (2002), Nr. 1-2, S. 123–138. – URL https://doi.org/10.1016/S0020-0255(01)00194-3

[Chaumont 2020] Chaumont, Marc: Deep learning in steganography and steganalysis. In: *Digital Media Steganography: Principles, Algorithms, Advances, M. Hassaballah (Ed.). Vol. abs/1904.01444. Elsevier* (2020), S. 46

[Chen *et al.* 2018] Chen, Mo ; Boroumand, Mehdi ; Fridrich, Jessica J.: Deep Learning Regressors for Quantitative Steganalysis. In: *Media Watermarking, Security, and Forensics 2018, Burlingame, CA, USA, 28 January 2018 - 1 February 2018*, URL https://doi.org/10.2352/ISSN.2470-1173.2018.07.MWSF-160, 2018

[Chen *et al.* 2017] Chen, Mo ; Sedighi, Vahid ; Boroumand, Mehdi ; Fridrich, Jessica J.: JPEG-Phase-Aware Convolutional Neural Network for Steganalysis of JPEG Images. In: *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2017, Philadelphia, PA, USA, June 20-22, 2017*, URL https://doi.org/10.1145/3082031.3083248, 2017, S. 75–84

[Cogranne 2015] Cogranne, Rémi: A sequential method for online steganalysis. In: *2015 IEEE International Workshop on Information Forensics and Security, WIFS 2015, Roma, Italy, November 16-19, 2015*, URL https://doi.org/10.1109/WIFS.2015.7368596, 2015, S. 1–6

[Cogranne *et al.* 2020a] Cogranne, Rémi ; Bas, Patrick ; Chaumont, Marc: STÉGANALYSE : Détection d'information cachée dans des contenus multimédias. In: *Sécurité Multimédia - Partie 1 : Authentification et Insertion de Données Cachées*. URL https://hal-utt.archives-ouvertes.fr/hal-02470070, 2020

[Cogranne and Fridrich 2015] Cogranne, Rémi ; Fridrich, Jessica J.: Modeling and Extending the Ensemble Classifier for Steganalysis of Digital Images Using Hypothesis Testing Theory. In: *IEEE Trans. Information Forensics and Security* 10 (2015), Nr. 12, S. 2627–2642. – URL https://doi.org/10.1109/TIFS.2015.2470220

[Cogranne *et al.* 2019] Cogranne, Rémi ; Giboulot, Quentin ; Bas, Patrick: The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis. In: Cogranne, Rémi (Hrsg.) ; Verdoliva, Luisa (Hrsg.) ; Lyu, Siwei (Hrsg.) ; Troncoso-Pastoriza, Juan R. (Hrsg.) ; Zhang, Xinpeng (Hrsg.): *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2019, Paris, France, July 3-5, 2019*, ACM, 2019, S. 125–137. – URL https://doi.org/10.1145/3335203.3335726

[Cogranne *et al.* 2020b] Cogranne, Rémi ; Giboulot, Quentin ; Bas, Patrick: Steganography by Minimizing Statistical Detectability: The Cases of JPEG and Color Images. In: *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*. New York, NY, USA : Association for Computing Machinery, 2020 (IH&MMSec '20), S. 161–167. – URL https://doi.org/10.1145/3369412.3395075. – ISBN 9781450370509

[Cogranne *et al.* 2017] Cogranne, Rémi ; Sedighi, Vahid ; Fridrich, Jessica J.: Practical strategies for content-adaptive batch steganography and

pooled steganalysis. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, URL https://doi.org/10.1109/ICASSP.2017.7952531, 2017, S. 2122–2126

[Cogranne *et al.* 2015] Cogranne, Rémi ; Sedighi, Vahid ; Fridrich, Jessica J. ; Pevný, Tomás: Is ensemble classifier needed for steganalysis in high-dimensional feature spaces? In: *2015 IEEE International Workshop on Information Forensics and Security, WIFS 2015, Roma, Italy, November 16-19, 2015*, URL https://doi.org/10.1109/WIFS.2015.7368597, 2015, S. 1–6

[Cogranne *et al.* 2011] Cogranne, Rémi ; Zitzmann, Cathel ; Fillatre, Lionel ; Retraint, Florent ; Nikiforov, Igor V. ; Cornu, Philippe: A Cover Image Model For Reliable Steganalysis. In: *Information Hiding - 13th International Conference, IH 2011, Prague, Czech Republic, May 18-20, 2011, Revised Selected Papers*, URL https://doi.org/10.1007/978-3-642-24178-9_13, 2011, S. 178–192

[Cole 2003] Cole, Eric: *Hiding in plain sight: steganography and the art of covert communication*. New York : Wiley Pub, 2003. – ISBN 9780471444497

[Crandall 1998] Crandall, Ron: Some notes on steganography. Posted on steganography mailing list (1998). In: *Source: http://dde.binghamton.edu/download/Crandall$_m$atrix.pdf* (1998)

[Daugman 1985] Daugman, John G.: Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. In: *Journal of the Optical Society of America A (JOSA A)* 2 (1985), Nr. 7, S. 1160–1169

[DELL 2014] DELL: Malware Analysis of the Lurk Downloader. In: *secureworks* (2014). – URL https://www.secureworks.com/research/malware-analysis-of-the-lurk-downloader. – Access date: 2020-04-09

[Denemark and Fridrich 2015] Denemark, Tomás ; Fridrich, Jessica J.: Improving Steganographic Security by Synchronizing the Selection Channel. In:

Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2015, Portland, OR, USA, June 17 - 19, 2015, URL https://doi.org/10.1145/2756601.2756620, 2015, S. 5–14

[Desoky 2016] Desoky, Abdelrahman: *Noiseless Steganography: The Key to Covert Communications*. Auerbach Publications, April 2016. – URL https://www.taylorfrancis.com/books/9781439846223. – Access date: 2020-05-20. – ISBN 9780429063213

[Dhobale *et al.* 2010] Dhobale, DD ; Ghorpade, VR ; Patil, BS ; Patil, SB: Steganography by hiding data in TCP/IP headers. In: *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)* Bd. 4 IEEE (Organizer.), 2010, S. V4–61

[Dilpazir *et al.* 2012] Dilpazir, Hammad ; Mahmood, Hasan ; Shah, Tariq ; Malik, Hafiz: Source Coding and Channel Coding for Mobile Multimedia Communication. In: *Mobile Multimedia-User and Technology Perspectives*. IntechOpen, 2012

[Duda 2001] Duda, RO: Hart. P., E., and Stork, DG," Pattern Classification". In: *Nueva York: John Wiley and Sons* (2001)

[Dumitrescu *et al.* 2002] Dumitrescu, Sorina ; Wu, Xiaolin ; Wang, Zhe: Detection of LSB Steganography via Sample Pair Analysis. In: *Information Hiding, 5th International Workshop, IH 2002, Noordwijkerhout, The Netherlands, October 7-9, 2002, Revised Papers*, URL https://doi.org/10.1007/3-540-36415-3_23, 2002, S. 355–372

[Ettinger 1998] Ettinger, Mark: Steganalysis and Game Equilibria. In: *Information Hiding, Second International Workshop, Portland, Oregon, USA, April 14-17, 1998, Proceedings*, URL https://doi.org/10.1007/3-540-49380-8_22, 1998, S. 319–328

[Fard *et al.* 2006] Fard, Amin M. ; Akbarzadeh-T, M-R ; Varasteh-A, Farshad: A new genetic algorithm approach for secure JPEG steganography.

In: *2006 IEEE International Conference on Engineering of Intelligent Systems* IEEE (Organizer.), 2006, S. 1–6

[Farid 2002] Farid, Hany: Detecting hidden messages using higher-order statistical models. In: *Proceedings of the 2002 International Conference on Image Processing, ICIP 2002, Rochester, New York, USA, September 22-25, 2002*, URL https://doi.org/10.1109/ICIP.2002.1040098, 2002, S. 905–908

[Filler *et al.* 2010] Filler, Tomás ; Judas, Jan ; Fridrich, Jessica J.: Minimizing embedding impact in steganography using trellis-coded quantization. In: *Media Forensics and Security II, part of the IS&T-SPIE Electronic Imaging Symposium, San Jose, CA, USA, January 18-20, 2010, Proceedings*, URL https://doi.org/10.1117/12.838002, 2010, S. 754105

[Filler *et al.* 2011] Filler, Tomás ; Judas, Jan ; Fridrich, Jessica J.: Minimizing Additive Distortion in Steganography Using Syndrome-Trellis Codes. In: *IEEE Trans. Information Forensics and Security* 6 (2011), Nr. 3-2, S. 920–935. – URL https://doi.org/10.1109/TIFS.2011.2134094

[Fisher 1936] Fisher, Ronald A.: The use of multiple measurements in taxonomic problems. In: *Annals of Eugenics* 7 (1936), Nr. 2, S. 179–188

[Fox 2010] Fox, Stuart: FBI: Russian spies hid codes in online photos, Discovery marks the first real-life use of steganography in espionage. In: *nbcnews* (2010). – URL http://www.nbcnews.com/id/38028696/ns/technology_and_science-science/t/fbi-russian-spies-hid-codes-online-photos/#.Xo5wcoj7SUk. – Access date: 2020-04-09

[Fox-News 2010] Fox-News: Authorities Arrest Alleged Russian Spies in Massachusetts. In: *Fox News* (2010). – URL https://www.foxnews.com/us/authorities-arrest-alleged-russian-spies-in-massachusetts

[Francia and Gomez 2006] Francia, Guillermo A. ; Gomez, Tyler S.: Steganography obliterator: an attack on the least significant bits. In: *Proceedings of the 3rd Annual Conference on Information Security Curriculum Development, InfoSecCD 2006, Kennesaw, Georgia, USA, September 22-23, 2006*, URL https://doi.org/10.1145/1231047.1231066, 2006, S. 85–91

[Franz *et al.* 1996] Franz, Elke ; Jerichow, Anja ; Möller, Steffen ; Pfitzmann, Andreas ; Stierand, Ingo: Computer Based Steganography: How It Works and Why Therefore Any Restrictions on Cryptography Are Nonsense, at Best. In: *Information Hiding, First International Workshop, Cambridge, UK, May 30 - June 1, 1996, Proceedings*, URL https://doi.org/10.1007/3-540-61996-8_28, 1996, S. 7–21

[Fridrich 2009] Fridrich, Jessica: *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009

[Fridrich *et al.* 2002a] Fridrich, Jessica ; Goljan, Miroslav ; Hogea, Dorin: Attacking the outguess. In: *Proceedings of the ACM Workshop on Multimedia and Security* Bd. 2002 Juan-les-Pins, France (Organizer.), 2002

[Fridrich and Filler 2007] Fridrich, Jessica J. ; Filler, Tomás: Practical methods for minimizing embedding impact in steganography. In: *Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, USA, January 28, 2007*, URL https://doi.org/10.1117/12.697471, 2007, S. 650502

[Fridrich and Goljan 2004] Fridrich, Jessica J. ; Goljan, Miroslav: On estimation of secret message length in LSB steganography in spatial domain. In: *Security, Steganography, and Watermarking of Multimedia Contents VI, San Jose, California, USA, January 18-22, 2004, Proceedings*, URL https://doi.org/10.1117/12.521350, 2004, S. 23–34

[Fridrich *et al.* 2001] Fridrich, Jessica J. ; Goljan, Miroslav ; Du, Rui: Detecting LSB Steganography in Color and Gray-Scale Images. In: *IEEE MultiMedia* 8 (2001), Nr. 4, S. 22–28. – URL https://doi.org/10.1109/93.959097

[Fridrich *et al.* 2002b] Fridrich, Jessica J. ; Goljan, Miroslav ; Hogea, Dorin: Steganalysis of JPEG Images: Breaking the F5 Algorithm. In: *Information Hiding, 5th International Workshop, IH 2002, Noordwijkerhout, The Netherlands, October 7-9, 2002, Revised Papers*, URL https://doi.org/10.1007/3-540-36415-3_20, 2002, S. 310–323

[Fridrich *et al.* 2003] Fridrich, Jessica J. ; Goljan, Miroslav ; Hogea, Dorin ; Soukal, David: Quantitative steganalysis of digital images: estimating the secret message length. In: *Multimedia Syst.* 9 (2003), Nr. 3, S. 288–302. – URL https://doi.org/10.1007/s00530-003-0100-9

[Fridrich *et al.* 2005] Fridrich, Jessica J. ; Goljan, Miroslav ; Lisonek, Petr ; Soukal, David: Writing on wet paper. In: *Security, Steganography, and Watermarking of Multimedia Contents VII, San Jose, California, USA, January 17-20, 2005, Proceedings*, URL https://doi.org/10.1117/12.583160, 2005, S. 328–340

[Fridrich and Kodovský 2013] Fridrich, Jessica J. ; Kodovský, Jan: Multivariate gaussian model for designing additive distortion for steganography. In: *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, URL https://doi.org/10.1109/ICASSP.2013.6638198, 2013, S. 2949–2953

[Fridrich *et al.* 2007] Fridrich, Jessica J. ; Pevný, Tomás ; Kodovský, Jan: Statistically undetectable jpeg steganography: dead ends challenges, and opportunities. In: *Proceedings of the 9th workshop on Multimedia & Security, MM&Sec 2007, Dallas, Texas, USA, September 20-21, 2007*, URL https://doi.org/10.1145/1288869.1288872, 2007, S. 3–14

[Friedman 2001] Friedman, Jerome H.: Greedy function approximation: a gradient boosting machine. In: *Annals of statistics* (2001), S. 1189–1232

[Gabor 1946] Gabor, D.: Theory of communication. Part 1: The analysis of information. In: *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering* 93 (1946), November, Nr. 26, S. 429–441

[Giboulot *et al.* 2020] Giboulot, Quentin ; Cogranne, Rémi ; Borghys, Dirk ; Bas, Patrick: Effects and solutions of Cover-Source Mismatch in image steganalysis. In: *Signal Process. Image Commun.* 86 (2020), S. 115888. – URL https://doi.org/10.1016/j.image.2020.115888

[Godse and Godse 2009] Godse, A.P. ; Godse, D.A.: *Computer Graphics And Multimedia*. Technical Publications, 2009. – URL https://books.google.fr/books?id=iJUvWG59zvkC. – ISBN 9788184317305

[Grigorescu *et al.* 2002] Grigorescu, Simona E. ; Petkov, Nicolai ; Kruizinga, Peter: Comparison of texture features based on Gabor filters. In: *IEEE Trans. Image Processing* 11 (2002), Nr. 10, S. 1160–1167. – URL https://doi.org/10.1109/TIP.2002.804262

[Guo *et al.* 2014] Guo, Linjie ; Ni, Jiangqun ; Shi, Yun-Qing: Uniform Embedding for Efficient JPEG Steganography. In: *IEEE Trans. Information Forensics and Security* 9 (2014), Nr. 5, S. 814–825. – URL https://doi.org/10.1109/TIFS.2014.2312817

[Guo *et al.* 2015] Guo, Linjie ; Ni, Jiangqun ; Su, Wenkang ; Tang, Chengpei ; Shi, Yun-Qing: Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited. In: *IEEE Trans. Information Forensics and Security* 10 (2015), Nr. 12, S. 2669–2680. – URL https://doi.org/10.1109/TIFS.2015.2473815

[Hassaballah 2020] Hassaballah, Mahmoud (Hrsg.): *DIGITAL MEDIA STEGANOGRAPHY*. S.l. : Elsevier Academic Press, 2020. – OCLC: 1130904927. – ISBN 9780128194386

[Hayes and Danezis 2017] Hayes, Jamie ; Danezis, George: Generating steganographic images via adversarial training. In: Guyon, Isabelle (Hrsg.) ; Luxburg, Ulrike von (Hrsg.) ; Bengio, Samy (Hrsg.) ; Wallach, Hanna M. (Hrsg.) ; Fergus, Rob (Hrsg.) ; Vishwanathan, S. V. N. (Hrsg.) ; Garnett, Roman (Hrsg.): *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9*

December 2017, Long Beach, CA, USA, URL http://papers.nips.cc/paper/6791-generating-steganographic-images-via-adversarial-training, 2017, S. 1954–1963

[He *et al.* 2016] He, Kaiming ; Zhang, Xiangyu ; Ren, Shaoqing ; Sun, Jian: Deep Residual Learning for Image Recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, URL https://doi.org/10.1109/CVPR.2016.90, 2016, S. 770–778

[Hetzl and Mutzel 2005] Hetzl, Stefan ; Mutzel, Petra: A Graph-Theoretic Approach to Steganography. In: *Communications and Multimedia Security, 9th IFIP TC-6 TC-11 International Conference, CMS 2005, Salzburg, Austria, September 19-21, 2005, Proceedings*, URL https://doi.org/10.1007/11552055_12, 2005, S. 119–128

[Holub and Fridrich 2015a] Holub, Vojtech ; Fridrich, Jessica J.: Low-Complexity Features for JPEG Steganalysis Using Undecimated DCT. In: *IEEE Trans. Information Forensics and Security* 10 (2015), Nr. 2, S. 219–228. – URL https://doi.org/10.1109/TIFS.2014.2364918

[Holub and Fridrich 2015b] Holub, Vojtech ; Fridrich, Jessica J.: Phase-aware projection model for steganalysis of JPEG images. In: *Media Watermarking, Security, and Forensics 2015, San Francisco, CA, USA, February 9-11, 2015, Proceedings*, URL https://doi.org/10.1117/12.2075239, 2015, S. 94090T

[Holub *et al.* 2014] Holub, Vojtech ; Fridrich, Jessica J. ; Denemark, Tomás: Universal distortion function for steganography in an arbitrary domain. In: *EURASIP J. Information Security* 2014 (2014), S. 1. – URL https://doi.org/10.1186/1687-417X-2014-1

[Huang *et al.* 2007] Huang, Fangjun ; Li, Bin ; Huang, Jiwu: Attack LSB Matching Steganography by Counting Alteration Rate of the Number of Neighbourhood Gray Levels. In: *Proceedings of the International Conference on Image*

Processing, ICIP 2007, September 16-19, 2007, San Antonio, Texas, USA, URL https://doi.org/10.1109/ICIP.2007.4378976, 2007, S. 401–404

[Huang *et al.* 2019] Huang, Junwen ; Ni, Jiangqun ; Wan, Linhong ; Yan, Jingwen: A Customized Convolutional Neural Network with Low Model Complexity for JPEG Steganalysis. In: Cogranne, Rémi (Hrsg.) ; Verdoliva, Luisa (Hrsg.) ; Lyu, Siwei (Hrsg.) ; Troncoso-Pastoriza, Juan R. (Hrsg.) ; Zhang, Xinpeng (Hrsg.): *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2019, Paris, France, July 3-5, 2019*, ACM, 2019, S. 198–203. – URL https://doi.org/10.1145/3335203.3335734

[Huang *et al.* 2018] Huang, Xiaosa ; Wang, Shilin ; Sun, Tanfeng ; Liu, Gongshen ; Lin, Xiang: Steganalysis of Adaptive JPEG Steganography Based on ResDet. In: *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2018, Honolulu, HI, USA, November 12-15, 2018*, URL https://doi.org/10.23919/APSIPA.2018.8659600, 2018, S. 549–553

[Huang *et al.* 2011] Huang, Yongfeng ; Tang, Shanyu ; Yuan, Jian: Steganography in Inactive Frames of VoIP Streams Encoded by Source Codec. In: *IEEE Trans. Information Forensics and Security* 6 (2011), Nr. 2, S. 296–306. – URL https://doi.org/10.1109/TIFS.2011.2108649

[Huang *et al.* 2017] Huang, YongFeng ; Tao, HuaiZhou ; Xiao, Bo ; Chang, ChinChen: Steganography in low bit-rate speech streams based on quantization index modulation controlled by keys. In: *Science China Technological Sciences* 60 (2017), Nr. 10, S. 1585–1596

[Kahn 1996] Kahn, David: The History of Steganography. In: *Information Hiding, First International Workshop, Cambridge, UK, May 30 - June 1, 1996, Proceedings*, URL https://doi.org/10.1007/3-540-61996-8_27, 1996, S. 1–5

[Kennedy 2004] Kennedy, Joann: Steganography in the Corporate Environment. In: *SANS\GIAC Practical - GSEC Certification, Version 1.4c*

(2004). – URL https://cyber-defense.sans.org/resources/papers/gsec/steganography-corporate-environment-106511. – Access date: 2020-04-09

[Ker 2004] Ker, Andrew D.: Improved Detection of LSB Steganography in Grayscale Images. In: Fridrich, Jessica J. (Hrsg.): *Information Hiding, 6th International Workshop, IH 2004, Toronto, Canada, May 23-25, 2004, Revised Selected Papers* Bd. 3200, Springer, 2004, S. 97–115. – URL https://doi.org/10.1007/978-3-540-30114-1_8

[Ker 2005] Ker, Andrew D.: A General Framework for Structural Steganalysis of LSB Replacement. In: *Information Hiding, 7th International Workshop, IH 2005, Barcelona, Spain, June 6-8, 2005, Revised Selected Papers*, URL https://doi.org/10.1007/11558859_22, 2005, S. 296–311

[Ker 2006] Ker, Andrew D.: Batch Steganography and Pooled Steganalysis. In: *Information Hiding, 8th International Workshop, IH 2006, Alexandria, VA, USA, July 10-12, 2006. Revised Selected Papers*, URL https://doi.org/10.1007/978-3-540-74124-4_18, 2006, S. 265–281

[Ker 2007a] Ker, Andrew D.: Batch steganography and the threshold game. In: *Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, USA, January 28, 2007*, URL https://doi.org/10.1117/12.703334, 2007, S. 650504

[Ker 2007b] Ker, Andrew D.: A Capacity Result for Batch Steganography. In: *IEEE Signal Process. Lett.* 14 (2007), Nr. 8, S. 525–528. – URL https://doi.org/10.1109/LSP.2006.891319

[Ker 2008] Ker, Andrew D.: Perturbation Hiding and the Batch Steganography Problem. In: *Information Hiding, 10th International Workshop, IH 2008, Santa Barbara, CA, USA, May 19-21, 2008, Revised Selected Papers*, URL https://doi.org/10.1007/978-3-540-88961-8_4, 2008, S. 45–59

[Ker 2017] Ker, Andrew D.: The Square Root Law of Steganography: Bringing Theory Closer to Practice. In: Stamm, Matthew C. (Hrsg.) ; Kirchner,

Matthias (Hrsg.) ; VOLOSHYNOVSKIY, Sviatoslav (Hrsg.): *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2017, Philadelphia, PA, USA, June 20-22, 2017*, ACM, 2017, S. 33–44. – URL https://doi.org/10.1145/3082031.3083235

[Ker *et al.* 2013a] KER, Andrew D. ; BAS, Patrick ; BÖHME, Rainer ; COGRANNE, Rémi ; CRAVER, Scott ; FILLER, Tomás ; FRIDRICH, Jessica J. ; PEVNÝ, Tomás: Moving steganography and steganalysis from the laboratory into the real world. In: *ACM Information Hiding and Multimedia Security Workshop, IH&MMSec '13, Montpellier, France, June 17-19, 2013*, URL https://doi.org/10.1145/2482513.2482965, 2013, S. 45–58

[Ker *et al.* 2013b] KER, Andrew D. ; BAS, Patrick ; BÖHME, Rainer ; COGRANNE, Rémi ; CRAVER, Scott ; FILLER, Tomás ; FRIDRICH, Jessica J. ; PEVNÝ, Tomás: Moving steganography and steganalysis from the laboratory into the real world. In: *ACM Information Hiding and Multimedia Security Workshop, IH&MMSec '13, Montpellier, France, June 17-19, 2013*, URL https://doi.org/10.1145/2482513.2482965, 2013, S. 45–58

[Ker and Böhme 2008] KER, Andrew D. ; BÖHME, Rainer: Revisiting weighted stego-image steganalysis. In: *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, San Jose, CA, USA, January 27, 2008*, URL https://doi.org/10.1117/12.766820, 2008, S. 681905

[Ker and Pevný 2012a] KER, Andrew D. ; PEVNÝ, Tomás: Identifying a steganographer in realistic and heterogeneous data sets. In: *Media Watermarking, Security, and Forensics 2012, Burlingame, CA, USA, January 22, 2012, Proceedings*, URL https://doi.org/10.1117/12.910565, 2012, S. 83030N

[Ker and Pevný 2014] KER, Andrew D. ; PEVNÝ, Tomás: The Steganographer is the Outlier: Realistic Large-Scale Steganalysis. In: *IEEE Trans. Information Forensics and Security* 9 (2014), Nr. 9, S. 1424–1435. – URL https://doi.org/10.1109/TIFS.2014.2336380

[Ker *et al.* 2008] Ker, Andrew D. ; Pevný, Tomás ; Kodovský, Jan ; Fridrich, Jessica J.: The square root law of steganographic capacity. In: Ker, Andrew D. (Hrsg.) ; Dittmann, Jana (Hrsg.) ; Fridrich, Jessica J. (Hrsg.): *Proceedings of the 10th workshop on Multimedia & Security, MM&Sec 2008, Oxford, UK, September 22-23, 2008*, ACM, 2008, S. 107–116. – URL https://doi.org/10.1145/1411328.1411349

[Ker and Pevný 2012b] Ker, Andrew D. ; Pevný, Tomás: Batch steganography in the real world. In: *Multimedia and Security Workshop, MM&Sec 2012, Coventry, United Kingdom, September 6-7, 2012*, URL https://doi.org/10.1145/2361407.2361409, 2012, S. 1–10

[Ker and Pevný 2012c] Ker, Andrew D. ; Pevný, Tomás: Batch steganography in the real world. In: *Multimedia and Security Workshop, MM&Sec 2012, Coventry, United Kingdom, September 6-7, 2012*, URL https://doi.org/10.1145/2361407.2361409, 2012, S. 1–10

[Kodovskỳ 2011] Kodovskỳ, Jan: On dangers of cross-validation in steganalysis / Citeseer. URL http://dde.binghamton.edu/kodovsky/pdf/Tech-2011-cv.pdf, August 2011. – Technical report

[Kodovský and Fridrich 2008] Kodovský, Jan ; Fridrich, Jessica J.: On completeness of feature spaces in blind steganalysis. In: *Proceedings of the 10th workshop on Multimedia & Security, MM&Sec 2008, Oxford, UK, September 22-23, 2008*, URL https://doi.org/10.1145/1411328.1411352, 2008, S. 123–132

[Kodovský and Fridrich 2011] Kodovský, Jan ; Fridrich, Jessica J.: Steganalysis in high dimensions: fusing classifiers built on random subspaces. In: *Media Forensics and Security III, San Francisco Airport, CA, USA, January 24-26, 2011, Proceedings*, URL https://doi.org/10.1117/12.872279, 2011, S. 78800L

[Kodovský and Fridrich 2012] Kodovský, Jan ; Fridrich, Jessica J.: Steganalysis of JPEG images using rich models. In: *Media Watermarking, Security, and*

Forensics 2012, Burlingame, CA, USA, January 22, 2012, Proceedings, URL https://doi.org/10.1117/12.907495, 2012, S. 83030A

[Kodovský and Fridrich 2013a] Kodovský, Jan ; Fridrich, Jessica J.: Quantitative steganalysis using rich models. In: *Media Watermarking, Security, and Forensics 2013, Burlingame, CA, USA, February 5-7, 2013, Proceedings*, URL https://doi.org/10.1117/12.2001563, 2013, S. 86650O

[Kodovský and Fridrich 2013b] Kodovský, Jan ; Fridrich, Jessica J.: Quantitative steganalysis using rich models. In: *Media Watermarking, Security, and Forensics 2013, Burlingame, CA, USA, February 5-7, 2013, Proceedings*, URL https://doi.org/10.1117/12.2001563, 2013, S. 86650O

[Kodovský *et al.* 2012] Kodovský, Jan ; Fridrich, Jessica J. ; Holub, Vojtech: Ensemble Classifiers for Steganalysis of Digital Media. In: *IEEE Trans. Information Forensics and Security* 7 (2012), Nr. 2, S. 432–444. – URL https://doi.org/10.1109/TIFS.2011.2175919

[Kouider *et al.* 2013] Kouider, Sarra ; Chaumont, Marc ; Puech, William: Adaptive steganography by oracle (ASO). In: *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo, ICME 2013, San Jose, CA, USA, July 15-19, 2013*, URL https://doi.org/10.1109/ICME.2013.6607427, 2013, S. 1–6

[Kurak and McHugh 1992] Kurak, Charles ; McHugh, John: A cautionary note on image downgrading. In: *Eighth Annual Computer Security Applications Conference, ACSAC 1992, November 30 - December 4, 1992, San Antonio, Texas, USA*, IEEE, 1992, S. 153–159. – URL https://doi.org/10.1109/CSAC.1992.228224

[Lerch-Hostalot and Megías 2019] Lerch-Hostalot, Daniel ; Megías, David: Detection of Classifier Inconsistencies in Image Steganalysis. In: Cogranne, Rémi (Hrsg.) ; Verdoliva, Luisa (Hrsg.) ; Lyu, Siwei (Hrsg.) ; Troncoso-Pastoriza, Juan R. (Hrsg.) ; Zhang, Xinpeng (Hrsg.): *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec*

2019, Paris, France, July 3-5, 2019, ACM, 2019, S. 222–229. – URL https://doi.org/10.1145/3335203.3335738

[Leyden 2010] Leyden, John: Russian spies dumped in Vienna after swap. In: *The Register - Independent news and views for the tech community. Part of Situation Publishing* (2010). – URL https://www.theregister.co.uk/2010/07/09/russia_spy_exchange/. – Access date: 2020-04-09

[Li *et al.* 2014] Li, Bin ; Wang, Ming ; Huang, Jiwu ; Li, Xiaolong: A new cost function for spatial image steganography. In: *2014 IEEE International Conference on Image Processing, ICIP 2014, Paris, France, October 27-30, 2014*, URL https://doi.org/10.1109/ICIP.2014.7025854, 2014, S. 4206–4210

[Li *et al.* 2018] Li, Bin ; Wei, Weihang ; Ferreira, Anselmo ; Tan, Shunquan: ReST-Net: Diverse Activation Modules and Parallel Subnets-Based CNN for Spatial Image Steganalysis. In: *IEEE Signal Process. Lett.* 25 (2018), Nr. 5, S. 650–654. – URL https://doi.org/10.1109/LSP.2018.2816569

[Liao and Yin 2018] Liao, Xin ; Yin, Jiaojiao: Two Embedding Strategies for Payload Distribution in Multiple Images Steganography. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, URL https://doi.org/10.1109/ICASSP.2018.8462384, 2018, S. 1982–1986

[Lu *et al.* 2016] Lu, Xiaorong ; Huang, Liusheng ; Yang, Wei ; Shen, Yao: Concealed in the Internet: A Novel Covert Channel with Normal Traffic Imitating. In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), Toulouse, France, July 18-21, 2016*, URL https://doi.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0060, 2016, S. 285–292

[Lubenko and Ker 2012] Lubenko, Ivans ; Ker, Andrew D.: Going from small to large data in steganalysis. In: *Media Watermarking, Security, and*

Forensics 2012, Burlingame, CA, USA, January 22, 2012, Proceedings, URL https://doi.org/10.1117/12.910214, 2012, S. 83030M

[Luo *et al.* 2010] Luo, Weiqi ; Huang, Fangjun ; Huang, Jiwu: Edge adaptive image steganography based on LSB matching revisited. In: *IEEE Trans. Information Forensics and Security* 5 (2010), Nr. 2, S. 201–214. – URL https://doi.org/10.1109/TIFS.2010.2041812

[Mazurczyk and Szczypiorski 2008] Mazurczyk, Wojciech ; Szczypiorski, Krzysztof: Steganography of VoIP Streams. In: Meersman, Robert (Hrsg.) ; Tari, Zahir (Hrsg.): *On the Move to Meaningful Internet Systems: OTM 2008, OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part II* Bd. 5332, Springer, 2008, S. 1001–1018. – URL https://doi.org/10.1007/978-3-540-88873-4_6

[McAfee 2017] McAfee: McAfee Labs Threats Report. In: *McAfee* (2017). – URL http://mcafee.ly/2sXowrq. – Access date: 2017-06-01

[Murdoch and Lewis 2005] Murdoch, Steven J. ; Lewis, Stephen: Embedding covert channels into TCP/IP. In: *International Workshop on Information Hiding* Springer (Organizer.), 2005, S. 247–261

[Ndiaye *et al.* 2017] Ndiaye, Papa ; Chaumont, Marc ; Yedroudj, Mehdi ; Zakaria, Ahmad: Stéganographie et Stéganalyse des images JPEG Couleur. In: *CORESA'2017, COmpression et REprésentation des Signaux Audiovisuels, Caen, France, 20-21 septembre* (2017). – URL http://www.lirmm.fr/~chaumont/publications/CORESA-2017-NDIAYE-CHAUMONT-YEDROUDJ-ZAKARIA-SteganographieCouleur.pdf

[Pan *et al.* 2016] Pan, Yuanfeng ; Ni, Jiangqun ; Su, Wenkang: Improved Uniform Embedding for Efficient JPEG Steganography. In: *Cloud Computing and Security - Second International Conference, ICCCS 2016, Nanjing, China,*

July 29-31, 2016, Revised Selected Papers, Part I, URL https://doi.org/10.1007/978-3-319-48671-0_12, 2016, S. 125–133

[Pasquet *et al.* 2014] Pasquet, Jérôme ; Bringay, Sandra ; Chaumont, Marc: Steganalysis with cover-source mismatch and a small learning database. In: *22nd European Signal Processing Conference, EUSIPCO 2014, Lisbon, Portugal, September 1-5, 2014*, URL http://ieeexplore.ieee.org/document/6952885/, 2014, S. 2425–2429

[Pedregosa *et al.* 2011] Pedregosa, F. ; Varoquaux, G. ; Gramfort, A. ; Michel, V. ; Thirion, B. ; Grisel, O. ; Blondel, M. ; Prettenhofer, P. ; Weiss, R. ; Dubourg, V. ; Vanderplas, J. ; Passos, A. ; Cournapeau, D. ; Brucher, M. ; Perrot, M. ; Duchesnay, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[Pennebaker and Mitchell 1992] Pennebaker, William B. ; Mitchell, Joan L.: *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992

[Pevný *et al.* 2010] Pevný, Tomás ; Filler, Tomás ; Bas, Patrick: Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In: *Information Hiding - 12th International Conference, IH 2010, Calgary, AB, Canada, June 28-30, 2010, Revised Selected Papers*, URL https://doi.org/10.1007/978-3-642-16435-4_13, 2010, S. 161–177

[Pevný *et al.* 2009] Pevný, Tomás ; Fridrich, Jessica J. ; Ker, Andrew D.: From blind to quantitative steganalysis. In: *Media Forensics and Security I, part of the IS&T-SPIE Electronic Imaging Symposium, San Jose, CA, USA, January 19-21, 2009, Proceedings*, URL https://doi.org/10.1117/12.805601, 2009, S. 72540C

[Pevný *et al.* 2012] Pevný, Tomás ; Fridrich, Jessica J. ; Ker, Andrew D.: From Blind to Quantitative Steganalysis. In: *IEEE Trans. Information Forensics and Security* 7 (2012), Nr. 2, S. 445–454. – URL https://doi.org/10.1109/TIFS.2011.2175918

[Pevnỳ and Ker 2015] Pevnỳ, Tomáš ; Ker, Andrew D.: Towards dependable steganalysis. In: *Media Watermarking, Security, and Forensics 2015* Bd. 9409 International Society for Optics and Photonics (Organizer.), 2015, S. 94090I

[Pevný and Ker 2015] Pevný, Tomás ; Ker, Andrew D.: Towards dependable steganalysis. In: Alattar, Adnan M. (Hrsg.) ; Memon, Nasir D. (Hrsg.) ; Heitzenrater, Chad (Hrsg.): *Media Watermarking, Security, and Forensics 2015, San Francisco, CA, USA, February 9-11, 2015, Proceedings* Bd. 9409, SPIE, 2015, S. 94090I. – URL https://doi.org/10.1117/12.2083216

[Pevný and Nikolaev 2015] Pevný, Tomás ; Nikolaev, Ivan: Optimizing pooling function for pooled steganalysis. In: *2015 IEEE International Workshop on Information Forensics and Security, WIFS 2015, Roma, Italy, November 16-19, 2015*, URL https://doi.org/10.1109/WIFS.2015.7368555, 2015, S. 1–6

[Pibre *et al.* 2016] Pibre, Lionel ; Pasquet, Jérôme ; Ienco, Dino ; Chaumont, Marc: Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch. In: *Media Watermarking, Security, and Forensics 2016, San Francisco, California, USA, February 14-18, 2016*, URL http://ist.publisher.ingentaconnect.com/contentone/ist/ei/2016/00002016/00000008/art00014, 2016, S. 1–11

[Provos 2001] Provos, Niels: Defending Against Statistical Steganalysis. In: *10th USENIX Security Symposium, August 13-17, 2001, Washington, D.C., USA*, URL http://www.usenix.org/publications/library/proceedings/sec01/provos.html, 2001

[Qian *et al.* 2015] Qian, Yinlong ; Dong, Jing ; Wang, Wei ; Tan, Tieniu: Deep learning for steganalysis via convolutional neural networks. In: *Media Watermarking, Security, and Forensics 2015, San Francisco, CA, USA, February*

9-11, 2015, Proceedings, URL https://doi.org/10.1117/12.2083479, 2015, S. 94090J

[Ryan 2012] Ryan, D.: *E - Learning Modules: Dlr Associates Series*. AuthorHouse, 2012. – URL https://books.google.fr/books?id=bYxMVVzdV80C. – ISBN 9781468575200

[Sallee 2005] Sallee, Phil: Model-Based Methods For Steganography And Steganalysis. In: *Int. J. Image Graphics* 5 (2005), Nr. 1, S. 167–190. – URL https://doi.org/10.1142/S0219467805001719

[Sarkar *et al.* 2007] Sarkar, Anindya ; Solanki, Kaushal ; Madhow, Upamanyu ; Chandrasekaran, Shivkumar ; Manjunath, Bangalore S.: Secure Steganography: Statistical Restoration of the Second Order Dependencies for Improved Security. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2007, Honolulu, Hawaii, USA, April 15-20, 2007*, URL https://doi.org/10.1109/ICASSP.2007.366226, 2007, S. 277–280

[Schwamberger and Franz 2010] Schwamberger, Valentin ; Franz, Matthias O.: Simple algorithmic modifications for improving blind steganalysis performance. In: *Multimedia and Security Workshop, MM&Sec 2010, Roma, Italy, September 9-10, 2010*, URL https://doi.org/10.1145/1854229.1854268, 2010, S. 225–230

[Sedighi *et al.* 2016a] Sedighi, Vahid ; Cogranne, Rémi ; Fridrich, Jessica J.: Content-Adaptive Steganography by Minimizing Statistical Detectability. In: *IEEE Trans. Information Forensics and Security* 11 (2016), Nr. 2, S. 221–234. – URL https://doi.org/10.1109/TIFS.2015.2486744

[Sedighi *et al.* 2016b] Sedighi, Vahid ; Cogranne, Rémi ; Fridrich, Jessica J.: Content-Adaptive Steganography by Minimizing Statistical Detectability. In: *IEEE Trans. Information Forensics and Security* 11 (2016), Nr. 2, S. 221–234. – URL https://doi.org/10.1109/TIFS.2015.2486744

[Sedighi *et al.* 2015] Sedighi, Vahid ; Fridrich, Jessica J. ; Cogranne, Rémi: Content-adaptive pentary steganography using the multivariate generalized Gaussian cover model. In: *Media Watermarking, Security, and Forensics 2015, San Francisco, CA, USA, February 9-11, 2015, Proceedings*, URL https://doi.org/10.1117/12.2080272, 2015, S. 94090H

[SentinelOne 2019] SentinelOne: Hiding Code Inside Images: How Malware Uses Steganography. In: *SentinelOne* (2019). – URL https://www.sentinelone.com/blog/hiding-code-inside-images-malware-steganography/. – Access date: 2020-04-10

[Sharifzadeh *et al.* 2020] Sharifzadeh, Mehdi ; Aloraini, Mohammed ; Schonfeld, Dan: Adaptive Batch Size Image Merging Steganography and Quantized Gaussian Image Steganography. In: *IEEE Trans. Information Forensics and Security* 15 (2020), S. 867–879. – URL https://doi.org/10.1109/TIFS.2019.2929441

[Sharp 2001] Sharp, Toby: An Implementation of Key-Based Digital Signal Steganography. In: *Information Hiding, 4th International Workshop, IHW 2001, Pittsburgh, PA, USA, April 25-27, 2001, Proceedings*, URL https://doi.org/10.1007/3-540-45496-9_2, 2001, S. 13–26

[Shulmin and Krylova 2017] Shulmin, A. ; Krylova, E.: Steganography in Contemporary Cyberattacks. In: *SECURELIST, Kaspersky* (2017). – URL https://securelist.com/steganography-in-contemporary-cyberattacks/79276/. – Access date: 2017-08-03

[Simmons 1983] Simmons, Gustavus J.: The Prisoners' Problem and the Subliminal Channel. In: Chaum, David (Hrsg.): *Advances in Cryptology, Proceedings of CRYPTO '83, Santa Barbara, California, USA, August 21-24, 1983*, Plenum Press, New York, 1983, S. 51–67

[Solanki *et al.* 2005] Solanki, Kaushal ; Sullivan, Kenneth ; Madhow, Upamanyu ; Manjunath, B. S. ; Chandrasekaran, Shivkumar: Statistical restoration for robust and secure steganography. In: *Proceedings of the 2005 International Conference on Image Processing, ICIP 2005, Genoa, Italy, September 11-14, 2005*, URL https://doi.org/10.1109/ICIP.2005.1530256, 2005, S. 1118–1121

[Solanki *et al.* 2006] Solanki, Kaushal ; Sullivan, Kenneth ; Madhow, Upamanyu ; Manjunath, B. S. ; Chandrasekaran, Shivkumar: Provably Secure Steganography: Achieving Zero K-L Divergence using Statistical Restoration. In: *Proceedings of the International Conference on Image Processing, ICIP 2006, October 8-11, Atlanta, Georgia, USA*, URL https://doi.org/10.1109/ICIP.2006.312388, 2006, S. 125–128

[Song *et al.* 2015] Song, Xiaofeng ; Liu, Fenlin ; Yang, Chunfang ; Luo, Xiangyang ; Zhang, Yi: Steganalysis of Adaptive JPEG Steganography Using 2D Gabor Filters. In: *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2015, Portland, OR, USA, June 17 - 19, 2015*, URL https://doi.org/10.1145/2756601.2756608, 2015, S. 15–23

[Stein 2019] Stein, Eliya: Confiant Malwarebytes Uncover Steganography Based Ad Payload That Drops Shlayer Trojan On Mac Users. In: *Confiant Malwarebytes* (2019). – URL https://blog.confiant.com/confiant-malwarebytes-uncover-steganography-based-ad-payload-that-drops-shlay – Access date: 2020-04-09

[Tan and Li 2014] Tan, Shunquan ; Li, Bin: Stacked convolutional auto-encoders for steganalysis of digital images. In: *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2014, Chiang Mai, Thailand, December 9-12, 2014*, URL https://doi.org/10.1109/APSIPA.2014.7041565, 2014, S. 1–4

[Tirkel *et al.* 1993] Tirkel, Anatol Z. ; Rankin, GA ; Van Schyndel, RM ; Ho, WJ ; Mee, NRA ; Osborne, Charles F.: Electronic watermark. In: *Digital Image Computing, Technology and Applications (DICTA'93)* (1993), S. 666–673

[Tsang and Fridrich 2018] Tsang, Clement F. ; Fridrich, Jessica J.: Steganalyzing Images of Arbitrary Size with CNNs. In: *Media Watermarking, Security, and Forensics 2018, Burlingame, CA, USA, 28 January 2018 - 1 February 2018*, URL https://doi.org/10.2352/ISSN.2470-1173.2018.07.MWSF-121, 2018

[Upham 1997a] Upham, Derek: Jpeg-jsteg. In: *Computer software-Modification of the Independent JPEG group's JPEG software (release 4) for I-bit steganography in JFIF output files* (1997)

[Upham 1997b] Upham, Derek: Jsteg embedding scheme. (1997). – URL http://www.tiac.net/users/korejwa/jsteg.htm

[Upham 2008] Upham, Derek: Jpeg-jsteg-v4. (2008). – URL http://www.funet.fi/pub/crypt/steganography/jpeg-jsteg-v4.diff.gz

[Wallace 1991] Wallace, Gregory K.: The JPEG Still Picture Compression Standard. In: *Commun. ACM* 34 (1991), Nr. 4, S. 30–44. – URL https://doi.org/10.1145/103085.103089

[Wei *et al.* 2018] Wei, Qingde ; Yin, Zhaoxia ; Wang, Zichi ; Zhang, Xinpeng: Distortion function based on residual blocks for JPEG steganography. In: *Multimedia Tools Appl.* 77 (2018), Nr. 14, S. 17875–17888. – URL https://doi.org/10.1007/s11042-017-5053-7

[Westfeld 2001] Westfeld, Andreas: F5-A Steganographic Algorithm. In: *Information Hiding, 4th International Workshop, IHW 2001, Pittsburgh, PA, USA, April 25-27, 2001, Proceedings*, URL https://doi.org/10.1007/3-540-45496-9_21, 2001, S. 289–302

[Westfeld and Pfitzmann 1999] Westfeld, Andreas ; Pfitzmann, Andreas: Attacks on Steganographic Systems. In: *Information Hiding, Third International Workshop, IH'99, Dresden, Germany, September 29 - October 1, 1999, Proceedings*, URL https://doi.org/10.1007/10719724_5, 1999, S. 61–76

[Whitehead 2003] Whitehead, David: *10*. Bd. 1: *Aineias The Tactician*. 2. The address : Bristol: Bristol Classical Press, 2003. – ISBN 978-1853996276

[Wu *et al.* 2005] Wu, Mingqiao ; Zhu, Zhongliang ; Jin, Shiyao: A New Steganalytic Algorithm for Detecting Jsteg. In: *Networking and Mobile Computing, Third International Conference, ICCNMC 2005, Zhangjiajie, China, August 2-4, 2005, Proceedings*, URL https://doi.org/10.1007/11534310_112, 2005, S. 1073–1082

[Xia *et al.* 2019] Xia, C. ; Guan, Q. ; Zhao, X. ; Wu, K.: Improved JPEG Phase-Aware Steganalysis Features Using Multiple Filter Sizes and Difference Images. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2019), S. 1–1. – URL https://doi.org/10.1109/TCSVT.2019.2954041

[Xia *et al.* 2017] Xia, Chao ; Guan, Qingxiao ; Zhao, Xianfeng ; Xu, Zhoujun ; Ma, Yi: Improving GFR Steganalysis Features by Using Gabor Symmetry and Weighted Histograms. In: *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2017, Philadelphia, PA, USA, June 20-22, 2017*, URL https://doi.org/10.1145/3082031.3083243, 2017, S. 55–66

[Xu 2017] Xu, Guanshuo: Deep Convolutional Neural Network to Detect J-UNIWARD. In: *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2017, Philadelphia, PA, USA, June 20-22, 2017*, URL https://doi.org/10.1145/3082031.3083236, 2017, S. 67–73

[Xu *et al.* 2016] Xu, Guanshuo ; Wu, Han-Zhou ; Shi, Yun Q.: Ensemble of CNNs for Steganalysis: An Empirical Study. In: *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec*

2016, Vigo, Galicia, Spain, June 20-22, 2016, URL https://doi.org/10.1145/2909827.2930798, 2016, S. 103–107

[Yahya 2019] Yahya, Abid: *Steganography Techniques for Digital Images*. 2019. – OCLC: 1136434077. – ISBN 9783319785974

[Yedroudj *et al.* 2018] Yedroudj, Mehdi ; Comby, Frédéric ; Chaumont, Marc: Yedroudj-Net: An Efficient CNN for Spatial Steganalysis. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, IEEE, 2018, S. 2092–2096. – URL https://doi.org/10.1109/ICASSP.2018.8461438

[Yedroudj *et al.* 2019] Yedroudj, Mehdi ; Comby, Frédéric ; Chaumont, Marc: Steganography using a 3 player game. In: *CoRR* abs/1907.06956 (2019). – URL http://arxiv.org/abs/1907.06956

[Yousfi *et al.* 2019] Yousfi, Yassine ; Fridrich, Jessica ; Butora, Jan ; Giboulot, Quentin: Breaking ALASKA: Color Separation for Steganalysis in JPEG Domain. In: *Proceedings of the 7th ACM Workshop on Information Hiding and Multimedia Security*. Paris, France, Juli 2019 (IH&MMSec'19)

[Zakaria *et al.* 2018] Zakaria, Ahmad ; Chaumont, Marc ; Subsol, Gérard: Quantitative and Binary Steganalysis in JPEG: A Comparative Study. In: *26th European Signal Processing Conference, EUSIPCO 2018, Roma, Italy, September 3-7, 2018*, URL https://doi.org/10.23919/EUSIPCO.2018.8553580, 2018, S. 1422–1426

[Zakaria *et al.* 2019a] Zakaria, Ahmad ; Chaumont, Marc ; Subsol, Gérard: Pooled Steganalysis in JPEG: how to deal with the spreading strategy? In: *IEEE International Workshop on Information Forensics and Security, WIFS 2019, Delft, The Netherlands, December 9-12, 2019*, IEEE, 2019, S. 1–6. – URL https://doi.org/10.1109/WIFS47025.2019.9035096

[Zakaria *et al.* 2019b] Zakaria, Ahmad ; Chaumont, Marc ; Subsol, Gérard: Stéganalyse groupée en JPEG: comment gérer la stratégie d'étalement? In:

GRETSI'19, XXVIIème Colloque francophone de traitement du signal et des images (2019). – URL http://www.lirmm.fr/~chaumont/publications/GRETSI_2019_ZAKARIA_CHAUMONT_SUBSOL-Pooled_Steganalysis.pdf

[Zeng *et al.* 2017] Zeng, Jishen ; Tan, Shunquan ; Li, Bin ; Huang, Jiwu: Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis. In: *Media Watermarking, Security, and Forensics 2017, Burlingame, CA, USA, 29 January 2017 - 2 February 2017*, URL https://doi.org/10.2352/ISSN.2470-1173.2017.7.MWSF-324, 2017, S. 44–49

[Zeng *et al.* 2018] Zeng, Jishen ; Tan, Shunquan ; Li, Bin ; Huang, Jiwu: Large-Scale JPEG Image Steganalysis Using Hybrid Deep-Learning Framework. In: *IEEE Trans. Information Forensics and Security* 13 (2018), Nr. 5, S. 1200–1214. – URL https://doi.org/10.1109/TIFS.2017.2779446

[Zheng *et al.* 2007] Zheng, Dong ; Liu, Yan ; Zhao, Jiying ; Saddik, Abdulmotaleb E.: A survey of RST invariant image watermarking algorithms. In: *ACM Computing Surveys (CSUR)* 39 (2007), Nr. 2, S. 5–es

[Zhu *et al.* 2018] Zhu, Jiren ; Kaplan, Russell ; Johnson, Justin ; Fei-Fei, Li: HiDDeN: Hiding Data With Deep Networks. In: Ferrari, Vittorio (Hrsg.) ; Hebert, Martial (Hrsg.) ; Sminchisescu, Cristian (Hrsg.) ; Weiss, Yair (Hrsg.): *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV* Bd. 11219, Springer, 2018, S. 682–697. – URL https://doi.org/10.1007/978-3-030-01267-0_40

[Zichi Wang 2016] Zichi Wang, Zhaoxia Y.: Hybrid distortion function for JPEG steganography. In: *Journal of Electronic Imaging* 25 (2016), S. 25 – 25 – 4. – URL https://doi.org/10.1117/1.JEI.25.5.050501

[Zitzmann *et al.* 2011] Zitzmann, Cathel ; Cogranne, Rémi ; Retraint, Florent ; Nikiforov, Igor V. ; Fillatre, Lionel ; Cornu, Philippe: Statistical Decision Methods in Hidden Information Detection. In: *Information Hiding - 13th International Conference, IH 2011, Prague, Czech Republic,*

May 18-20, 2011, Revised Selected Papers, URL https://doi.org/10.1007/978-3-642-24178-9_12, 2011, S. 163–177

Résumé

La stéganographie par lot consiste à dissimuler un message en le répartissant dans un ensemble d'images, tandis que la stéganalyse groupée consiste à analyser un ensemble d'images pour conclure à la présence ou non d'un message caché.

Il existe de nombreuses stratégies d'étalement d'un message et on peut raisonnablement penser que la stéganalyste ne connaît pas celle qui est utilisée, mais il peut supposer que la stéganographe utilise un algorithme d'insertion unique (sous-entendu pour toutes les images). Dans ce cas, la solution la plus appropriée pour la stéganalyse groupée est d'utiliser un détecteur quantitatif (qui prédit la taille du message caché) à image unique, d'évaluer pour chaque image la longueur du message caché (qui peut être nulle) et de faire la moyenne des tailles (également considérés comme des scores) obtenus sur un sac d'images.

Et si, maintenant, le stéganalyste pouvait discriminer la stratégie d'étalement ? Supposons que la stratégie d'étalement appartient à un ensemble de stratégies qu'il connaît. La stéganalyste pourrait-elle utiliser un algorithme de stéganalyse groupé meilleur que la moyenne des scores ? La stéganalyste pourrait-elle obtenir des résultats proches du scénario dit "clairvoyant" où l'on suppose qu'elle connaît exactement la stratégie d'étalement ? Dans cette thèse, nous essayons de répondre à ces questions en proposant une architecture de stéganalyse groupée construite en utilisant un détecteur quantitatif d'images et une fonction de groupement optimisée.

La première contribution est une comparaison entre les algorithmes de stéganalyse quantitatifs afin de décider lequel est le mieux adapté à la stéganalyse groupée. Nous proposons aussi d'étendre cette comparaison aux algorithmes de stéganalyse binaires. Pour cela nous proposons une méthodologie pour utiliser ces deux types d'algorithmes que ce soit pour de la stéganalyse binaire ou quantitative.

Le coeur de la thèse se situe dans la deuxième contribution. Nous étudions le scénario où le stéganalyste ne connaît pas la stratégie d'étalement. Nous proposons une fonction de groupement discriminative optimisée sur un ensemble de stratégies

d'étalement qui permet d'améliorer la précision de la stéganalyse groupée par rapport à une simple moyenne. Pour faire apprendre cette fonction de groupement, nous utilisons des techniques d'apprentissage supervisé.

Les résultats empiriques obtenus avec six stratégies d'étalement différentes et un détecteur d'image unique de l'état de l'art confirment notre hypothèse. Notre fonction de groupement obtient des résultats proches d'un stéganalyste *clairvoyant* qui est censé connaître la stratégie d'étalement.

Mots clés : Sécurité multimédia, Stéganographie par lot, Stéganalyse groupée, Apprentissage machine.

Abstract

Batch steganography consists of hiding a message by spreading it into a set of images, while pooled steganalysis consists of analyzing a set of images to conclude whether or not a hidden message is present. There are many strategies for spreading a message and it is reasonable to assume that the steganalyst is not aware of which one is being used, but it can assume that the steganographer uses a single embedding algorithm (implied for all images). In this case, the most appropriate solution for pooled steganalysis is to use a single-image quantitative detector (which predicts the size of the hidden message), to evaluate for each image the size of the hidden message (which can be zero) and to average the sizes (scores) obtained on a bag of images.

What if now the steganalyst could discriminate the spreading strategy? Assume that the spreading strategy belongs to a set of strategies she knows. Could the steganalyst use a pooled steganalysis algorithm that is better than averaging scores? Could the steganalyst obtain results close to the so-called "clairvoyant" scenario where it is assumed that the steganalyst knows exactly what the spreading strategy is? In this thesis, we try to answer these questions by proposing a pooled steganalysis architecture built using a quantitative image detector and an optimized pooling function.

The first contribution is a comparison between quantitative steganalysis algorithms in order to decide which one is best suited for pooled steganalysis. We also propose to extend this comparison to binary steganalysis algorithms. For this we propose a methodology to use both types of algorithms for both binary and quantitative steganalysis. The core of the thesis lies in the second contribution. We study the scenario where the steganalyst does not know the spreading strategy. We propose a discriminative pooling function optimized on a set of spreading strategies that allows to improve the accuracy of the pooled steganalysis compared to a simple average. This pooling function is learned using supervised learning techniques. Empirical results obtained with six different spreading strategies and a single state-of-the-art image detector confirm our hypothesis. Our pooling function obtains results close to a *clairvoyant* steganalyst who is supposed to know the spreading strategy.

Keywords: Multimedia Security, Batch Steganography, Pooled Steganalysis, Machine Learning.