Digital Forensic Techniques for Splicing Detection in Multimedia Contents

Marco Fontani

Ph.D Thesis in Information Engineering

University of Siena

UNIVERSITÀ DEGLI STUDI DI SIENA

Dipartimento di Ingegneria dell'Informazione
e Scienze Matematiche

Digital Forensic Techniques for Splicing Detection in Multimedia Contents

Marco Fontani

Ph.D Thesis in Information Engineering
XXVI Cycle, 2010-2013

Supervisor
Prof. Mauro Barni

Co-Supervisor
Prof. Alessandro Piva

Thesis reviewers
Prof. Anthony T. S. Ho
Prof. Fabio Roli
Prof. Pedro Comesaña Alfaro

Examination Committee
Prof. Anthony T. S. Ho
Prof. Fabio Roli
Prof. Marco Maggini

Siena
December 18, 2014

Contents

FOR *there are some who long to know for the sole purpose of knowing, and that is shameful curiosity; others who long to know in order to become known, and that is shameful vanity. To such as these we may apply the words of the Satirist: "Your knowledge counts for nothing unless your friends know you have it." There are others still who long for knowledge in order to sell its fruits for money or honors, and this is shameful profiteering; others again who long to know in order to be of service, and this is charity. Finally there are those who long to know in order to benefit themselves, and this is wisdom.*

Sermon XXXVI on The Song of Songs
St. Bernard of Clairvaux

REALISM *requires a certain method for observing and coming to know an object, and this method must not be imagined, thought of or organized and created by the subject: it must be imposed by the object. [...] This means that the method of knowing an object is dictated by the object itself and cannot be defined by me.*

The Religious Sense
Mons. Luigi Giussani

Chapter 1

Introduction

WHILE the reader goes through this introduction, thousands of events are being captured in the world using digital devices. A good deal of this huge population of multimedia contents will be stored somewhere and never be looked at again, while another considerable fraction will be uploaded on social networks and websites, thus becoming of public domain. But there is a relatively small part of the cake that will be used to convince someone that something happened. Be it in a newspaper, a breaking news website, or even within a court, there are multimedia contents that stand as proof of something, and are claimed to be a *credible* proof. In fact, people tend to be convinced rather easily by something that can be watched or heard, for mainly two reasons: senses are directly involved, much more than when reading or listening to someone, and there is an aura of objectivity around recordings. Unfortunately, the objectivity of multimedia contents is often overestimated: just by looking around, it is evident that creating fake digital contents is very common today. While fake pictures on gossip magazines can be considered harmless in most cases, there are scenarios in which the truthfulness of a multimedia content must be assessed, often without having access to any other information but the content itself.

Multimedia forensics is a relatively new discipline that seeks to solve this kind of problems: the basic idea behind is that any processing applied to a digital content leaves subtle traces, that can be analyzed to uncover the "digital history" of the object. Thus, the word *forensics* is not related only to the application field but rather to the methodology that is going to be followed: the analyst is given the content "as it is", and must retrieve any useful information about its past, in close similarity to what scientific police does on a crime scene.

In the last ten years, multimedia forensic researchers struggled to answer as many questions as possible about the past of a given image, audio or video, questions like "which was the recording device?", or "is the content authentic?". In particular, the second question attracted an increasing attention, because authenticity assessment is a crucial step in many fields: it is not hard to foresee the dangerousness of fake objects in medical, military, journalistic and legal contexts.

Among the different approaches to assess the authenticity of a content, *splicing detection* is probably the most promising: it consists in determining whether the analyzed content is the digital representation of something that really happened, or it has been forged by splicing together two or more contents, with a clever "cut-&-paste" job. When this happens, the forensic analyst hopes that each of the spliced signals brought some of its digital history with it, thus allowing to search for *inconsistencies* in the forged content. As the reader probably noticed, we used the word "signal"; as it will become evident in the rest of this thesis, indeed, images, audios and videos are different but share the same nature of digitized signals: all of them have been sampled, quantized, and possibly compressed at some point in their history. Therefore, while specific methods have to be devised for different media, the underlying rules can be inherited in some cases. Encouraged by this fact, in this thesis we address the problem of splicing detection in multimedia contents, working on the three main media: images, video and audio. For this reason, the thesis consists of three parts, one for each media.

The first part regards still images. When we began our studies on this topic, in 2010, a large family of tools for splicing detection was already available in the literature [1, 2]. Each tool differed from the others either in the specific trace it searched for (traces left by the camera [3], or by JPEG compression [4], etc.) or in the way the trace was searched for. Each tool was conceived alone, meaning that manipulation traces were searched independently one from the other. This scenario contrasted with the fact that the creation of a credible splicing typically involves many processing operations, so that the produced forgery shows more than one forensic trace. For this reason, we investigated the possibility of using several tools together, combining their outputs in an intelligent way. To this end, we developed a decision

fusion framework specifically tailored for image forensics, based on Dempster-Shafer Theory of Evidence (DST), which is a mathematical tool allowing to make inference in presence of uncertainty and lack of prior probabilities [5]. Besides merging the output of several tools, we found that the analyst can strongly improve the reliability of his conclusions by accounting for other clues like compatibility relationships between forensic traces, tool reliability under different working conditions, and so on.

In the second part of the thesis, we consider splicing detection in digital video. Despite the significant advancements made in the last years, forensic video analysis is not as advanced as image forensics [6]. This especially holds for splicing detection, where a few methods have been proposed, most of them working only under restrictive hypotheses. Moreover, the concept itself of video splicing is more involved with respect to images: the forger may want to change the meaning of the video on a frame-by-frame basis (*intra-frame* splicing), or to simply conceal an event by removing groups of frames (*inter-frame* splicing). These manipulations are deeply different, and need to be investigated with different techniques. In this thesis, we first introduce a new footprint, the Variation of Prediction Footprint (VPF), and show how it can be used to detect double video encoding. Then, building on the VPF, we develop two video forensic techniques for detecting both intra- and inter-frame video splicing. Noticeably, the intra-frame forgery detection technique is based on an important result coming from image forensics, that is double quantization analysis, properly adapted to the context.

The third and last part of the thesis is devoted to digital audio, namely to MP3 compressed tracks. Audio forensic roots date back to the '60s, when magnetic tapes were investigated to expose traces of manipulations. Today digital audio recordings are broadly used in the court, and their authenticity is often questioned [7]. As a contribution to this field, we investigated the possibility of "exporting" a technique that worked brilliantly for images, namely *calibration* [8], into the field of compressed audio, focusing on MP3 compression. Somehow similarly to what we did for video, we first propose a method to detect double compression and then build on such a method to localize traces of cut-&-paste operations.

Throughout the thesis it emerges that while image, audio and video foren-

sics are at different development stages, there are some concepts that stand like a common denominator in all of them. For example, there exists a robust link between double compression detection and splicing localization: this was already clear for images when we began our activity, and by leveraging on this concept also in video and audio we obtained interesting results. Another example regards multiple coding: regardless of the media, when the signal is compressed at decreasing quality the forensic analysis becomes very hard, if not impossible. Similarly, we also believe that the concepts we considered in the development of the fusion framework for digital images are also valid for other media, so that the framework presented in the first part of the thesis can also be used with audio and video forensic tools.

1.1 Overview and contribution

In the first part of the thesis we present a decision fusion framework for image forensics. After a brief introduction to image forensics in Chapter 2, highlighting the need for a synergic analysis, we enter the heart of our contribution in Chapter 3, where we first introduce those basic notions of Dempster-Shafer Theory that are essential to the rest of the chapter and then formalize the proposed *decision fusion framework*. The experimental validation of the framework is treated in Chapter 4, which also concludes the first part of the thesis, laying the basis for future developments.

The second part of the thesis is structured similarly: it begins with an introduction to video forensics, in Chapter 5. Following that, we present our contributions in Chapter 6: Section 6.1 presents the VPF and its application to the *detection of double encoding*; then Section 6.2 presents a *tool for inter-frame video splicing detection*, and its experimental validation. A new *tool for intra-frame splicing detection* is the topic of Section 6.3, that is also a good example of how methods for image forensics can be adapted to work on different domains with a moderate, though not negligible, effort. Chapter 7, finally, draws some concluding remarks. This part of the thesis is the result of a joint work with the University of Vigo.

The last part of the thesis begins with Chapter 8, that introduces audio forensics. Chapter 9 contains the proposed method for *double compression*

detection and its application as a tool for *fake quality detection* and *forgery localization* of MP3 audio tracks. The validity of the method is investigated in Chapter 10.

Before concluding the thesis, in Chapter 11 we give a brief summary of our contribution in the field of counter-forensics: while not directly related to splicing detection problem per se, counter-forensics is an important aspect of our discipline, and acted as a stimulating factor in the development of our contribution about decision fusion. Finally, we conclude the thesis in Chapter 12, summarizing the lessons learned through this work and outlining some possible future trends in multimedia forensics.

1.2 Activity within research projects

During our research activity we participated in several international research projects. This fact brought a significant contribution: we learned the importance of establishing contacts, sharing knowledge with other partners, and we hopefully advanced in the ability to focus the efforts toward specific objectives.

The LivingKnowledge (*Facts Opinions and Bias in Time*) project[1], funded by the European Commission under the FP7-FET programme and expired on February 2012, allowed us to take the first steps in the multimedia forensic field. The project studied the effect of diversity and time on opinions and bias. Our contribution focused on the application of multimedia forensics for recovering information on image history and manipulation, in order to investigate how visual contents may be used to affect opinions and bias of viewers. It was here that the need for decision fusion methods in image forensics became evident for the first time.

The REWIND project[2] (*REVerse engineering of audio-Visual content Data*), funded by the European Commission under the FP7-FET programme and expired on June 2014, accompanied the whole activity presented in this thesis. The goal of the project was to develop new theories and tools for investigating the digital history of multimedia contents. Also according to project reviewers opinion, REWIND reached and in some cases exceeded its

[1] `http://livingknowledge.europarchive.org`
[2] `http://www.rewindproject.eu`

objectives, so that it can be regarded as a successful story we are proud of being part of. Also, it was thanks to REWIND that the collaboration with the University of Vigo flourished, leading to some of the results presented in this thesis.

The collaboration with the University of Vigo was also fostered by the LIFTGATE[3] project (*Lifting Up the Research Potential of the Galician Telecomms Center*), that gave us the possibility to be a visiting student at GRADIANT[4] (Galician Research and Development Center in Advanced Telecommunications), located in Vigo; working with GRADIANT researchers gave an important impulse to our work on decision fusion.

Simultaneously, we worked on the AMULET project (*A MUlti-cLuE approach To image forensics*), funded by the U.S. Air Force European Office for Aerospace Research and Development (EOARD), that is currently active. The focus of the project is on the development of new techniques for multi-clue forensics analysis that, starting from the indications provided by a pool of forensics tools thought to detect the presence of specific artifacts, reach a global conclusion about the authenticity of a given image. In particular, it is of interest to investigate the practical applicability of different solutions for multi-clue analysis; to this end, we studied methods for merging information stemming from tools that do not require an exaggerated amount of training data, and we compared such methods with state of the art technologies. This research project was especially essential for the development of the first part of the thesis.

Finally, we had the possibility of interacting with industrial partners under the umbrella of MAVEN project[5] (*Management and Authenticity Verification of multimedia contENts*), funded by the European Commission under the FP7 - SME programme, also currently active. MAVEN focuses on the development of a set of tools for multimedia data management and security. More specifically, MAVEN objectives are centered on two key concepts: "search" and "verify"; MAVEN first searches for digital contents containing "objects" of interest (e.g., a face appearing on Closed-Circuit Television, CCTV). Once those objects are retrieved, advanced forensic analysis tools are applied to ver-

[3] `https://sites.google.com/a/gradiant.org/liftgate/`
[4] `http://www.gradiant.org`
[5] `http://maven-project.eu/`

ify their integrity and authenticity. Through the collaboration with forensic software companies like AMPED[6], we obtained stimulating information about the real needs of multimedia forensic analysts and about practical constraints that should be kept in mind during the research effort.

1.3 Publication list

The activity of the thesis resulted in the following publications, listed in reverse chronological order.

- International, peer reviewed journals:

    1. D. P. Dupplaw, M. Matthews, R. Johansson, G. Boato, A. Costanzo, **M. Fontani**, E. Minack, E. Demidova, R. Blanco, T. Griffiths *et al.*, "Information extraction from multimedia web documents: an open-source platform and testbed," *International Journal of Multimedia Information Retrieval*, pp. 1–15, 2014.
    2. T. Bianchi, A. De Rosa, **M. Fontani**, G. Rocciolo, and A. Piva, "Detection and localization of double compression in MP3 audio tracks," *EURASIP Journal on Information Security*, vol. 2014, no. 1, p. 10, 2014.
    3. **M. Fontani**, T. Bianchi, A. De Rosa, A. Piva, and M. Barni, "A forensic tool for investigating image forgeries," *Intl. Journal of Digital Crime and Forensics (IJDCF)*, vol. 5, no. 4, pp. 15–33, 2013.
    4. **M. Fontani**, T. Bianchi, A. De Rosa, A. Piva, and M. Barni, "A Framework for Decision Fusion in Image Forensics Based on Dempster-Shafer Theory of Evidence," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 4, pp. 593–607, 2013.
    5. S. Milani, **M. Fontani**, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, no. 1, 2012.

[6] `http://ampedsoftware.com/`

- International, peer reviewed conferences:

  1. A. De Rosa, A. Piva, **M. Fontani**, and M. Iuliani, "Investigating Multimedia Contents," in *ICCST 2014, IEEE Intl. Carnahan Conference on Security Technology*, October 2014, pp. 1–6.
  2. A. Gironi, **M. Fontani**, T. Bianchi, A. Piva, and M. Barni, "A video forensic technique for detecting frame deletion and insertion," in *ICASSP 2014, IEEE Intl. Conference on Acoustic Speech and Signal Processing*, May 2014, pp. 6226–6230.
  3. **M. Fontani** and M. Barni, "Countering anti-forensics by means of data fusion," in *IS&T/SPIE Electronic Imaging 2014*, February 2014.
  4. **M. Fontani**, E. Argones-Rúa, C. Troncoso, and M. Barni, "The Watchful Forensic Analyst: Multi-Clue Information Fusion with Background Knowledge," in *WIFS 2013, IEEE Intl. Workshop on Information Forensics and Security*, November 2013, pp. 1–6.
  5. D. Labartino, T. Bianchi, A. De Rosa, **M. Fontani**, D. Vazquez-Padin, A. Piva, and M. Barni, "Localization of forgeries in MPEG-2 video through GOP size and DQ analysis," in *MMSP 2013, IEEE Intl. Workshop on Multimedia Signal Processing*, September 2013, pp. 494–499.
  6. T. Bianchi, A. De Rosa, **M. Fontani**, G. Rocciolo, and A. Piva, "Detection and Classification of Double Compressed MP3 Audio Tracks," in *IHMMSEC 2013, ACM Workshop on Information Hiding and Multimedia Security*. New York, NY, USA: ACM, June 2013, pp. 159–164.
  7. D. Vazquez-Padin, **M. Fontani**, T. Bianchi, P. Comesana, A. Piva, and M. Barni, "Detection of video double encoding with GOP size estimation," in *WIFS 2012, IEEE Intl. Workshop on Information Forensics and Security*, December 2012, pp. 151–156.
  8. P. Bestagini, **M. Fontani**, S. Milani, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," in *EUSIPCO 2012, European Signal Processing Conference*, August 2012, pp. 1229–1233.

9. **M. Fontani**, T. Bianchi, A. De Rosa, A. Piva, and M. Barni, "A Dempster-Shafer framework for decision fusion in image forensics," in *WIFS 2011, IEEE Intl. Workshop on Information Forensics and Security*, December 2011, pp. 1–6.

The author of this thesis also contributed to the publications listed below; they are not discussed in details in the thesis as they deal with watermarking and counter-forensic. Chapter 11 gives a brief overview of our contributions to counter-forensics.

1. A. De Rosa, **M. Fontani**, M. Massai, A. Piva, and M. Barni, "Second-order statistics analysis to cope with contrast enhancement counter-forensics," to appear on *IEEE Signal Processing Letters*, 2014.

2. M. Barni, **M. Fontani**, and B. Tondi, "Universal Counterforensics of Multiple Compressed JPEG Images," in *IWDW 2014, IEEE Intl. Workshop on Digital-Forensics and Watermarking*, October 2013, pp. 1–15.

3. M. Barni, **M. Fontani**, and B. Tondi, "A universal attack against histogram-based image forensics," *Intl. Journal of Digital Crime and Forensics*, vol. 5, no. 3, pp. 35–52, 2013.

4. M. Barni, **M. Fontani**, and B. Tondi, "A universal technique to hide traces of histogram-based image manipulations," in *MMSEC 2012, ACM Multimedia and Security Workshop*. New York, NY, USA: ACM, September 2012, pp. 97–104.

5. **M. Fontani** and M. Barni, "Hiding traces of median filtering in digital images," in *EUSIPCO 2012, European Signal Processing Conference*, August 2012, pp. 1239–1243.

6. **M. Fontani**, A. De Rosa, R. Caldelli, F. Filippini, A. Piva, M. Consalvo, and V. Cappellini, "Reversible watermarking for image integrity verification in hierarchical PACS," in *MMSEC 2010, ACM Workshop on Multimedia and Security*. New York, NY, USA: ACM, September 2010, pp. 161–168.

1.4 Acknowledgments

When I had to take the decision about pursuing or not a Ph.D, one thing was evident to me: I was standing in front of great people to follow. After four intense years, I can definitely say that I was right. Prof. Mauro Barni, my advisor, introduced me to research in the most fascinating way: by letting me follow him. That means: follow his passion for studying and teaching and, more in general, his positive look to reality. He worked hard to show me how to make science, how to present my work, how to follow a project, and much more. But he was not alone in the challenging task of making a researcher out of me. Prof. Alessandro Piva, my co-advisor, and Dr. Alessia De Rosa (both from the University of Florence) were the first to enthusiastically introduce me to digital image processing and, more in general, to high quality research. Starting right from my Bachelor thesis, I always felt they were betting on me more that I would have done, and this still holds today. Besides my mentors, I would like to thank my thesis reviewers and members of the committee, Prof. Anthony Ho, Prof. Fabio Roli, Prof. Pedro Comesaña Alfaro and Prof. Marco Maggini for their insightful comments and suggestions. Moreover, I want to acknowledge the REWIND, AMULET, MAVEN and LIFTGATE projects for making possible the fruitful collaboration with very professional and friendly researchers (and, of course, for financial support).

I would like to thank all of my laboratory colleagues in Florence and Siena for their support and help: Tiziano, Pasquale, Massimo, Alessandro, Andrea, Benedetta, Siméon, Giulia, Riccardo and Tommaso. A special thanks go to David Vázquez-Padín (University of Vigo): he patiently introduced me to video processing, and shared with me the intuition and the research work that led to the Variation of Prediction Footprint. Working with David has been a great, exciting experience and I sincerely hope that our collaboration will keep going like our friendship does.

Since I deeply believe that what a man does in his job is just an expression of who he is, I want to thank all those persons who contribute to my education: my family, all my friends, especially those of the Movement of Communion and Liberation, and my wife Teresa, who always encouraged me (especially through her example) to pursue the great opportunities I had in front of me.

Part I

Decision Fusion Methods for Splicing Detection in Digital Images

Abstract

Images have always played a key role in the transmission of information, mainly because of their immediacy and presumed objectivity. In the last twenty years, the advent of digital imaging further fostered the use of pictures as the preferred way to convey convincing messages. However, digital imaging also gave a great impulse to image manipulation, and nowadays images are facing a trust crisis. Image Forensics has emerged as a possible way to solve the above crisis, by enabling the blind investigation of the processing history of an image. As more and more forensic algorithms are becoming available, it becomes of interest to devise intelligent methods to merge the information stemming from them, so to obtain a more comprehensive and reliable analysis. In doing that, the analyst may also exploit different kinds of background knowledge, some noticeable examples are information about the compatibility relationships between different forensic traces, and the reliability of each tool under different working conditions.

After providing a brief outline of image forensics, in this part of the thesis we deal with the problem of combining the information stemming from a set of heterogeneous image forensic tools; in particular, we propose a decision fusion framework based on Dempster-Shafer Theory of Evidence. Since this theory is not as widely spread as, say, the Bayesian inference theory, the most basic notions are introduced prior to describing the fusion framework.

Chapter 2

Introduction to Image Forensics

SINCE the beginning of their centuries-old history, photographic images were considered one of the strongest evidences supporting a fact: as the saying goes, "seeing is believing", and for years showing a shot of an event was sufficient to prove that the event had really happened. Today this is no longer true in general: no one trusts pictures printed on gossip magazines, as they are admittedly fake in most cases. It is also known that even an amateur user of Photoshop can forge credible fakes in minutes. Yet, there is a vast range of scenarios where a single picture can really make the difference: political, military, law and medical applications are just a few examples. When images are to be used in such applications, it is fundamental to investigate their processing history before accepting them as evidences. In most cases, this investigation must be performed blindly, meaning that there is no access to the images before their diffusion: this makes *active* solutions like digital watermarking [9] not viable, since they are based on hiding information inside the media before sending it out.

In the last years, image forensics emerged as a discipline that *blindly* analyzes images in the attempt to recover information about their processing history, by searching for subtle traces left by processing operations. Notice that this definition of "digital image forensics" has almost nothing to share with the wide branch of methods allowing to enhance the quality of an image, or to extract measures from its content[1].

As a matter of fact, a credible image forensic analysis can not rely on a single tool: forgeries can be created using a very wide range of methods, each leaving different traces within the content; moreover image forensic tools are

[1] We may think of the two disciplines as forming a cascade: first the trustability of an image is assessed, then its content is examined.

still not very reliable, so that a cross-analysis is often desirable.

In the rest of this chapter, we briefly outline the main achievements of digital image forensics, and finally introduce the need for a synergic analysis, where evidences coming from different analysis tools are combined to reach more sound conclusions.

2.1 What image forensics can do

The history of a digital image consists of several steps: it begins when the image is captured, which includes the automatic application of some in-device processing, possibly including lossy encoding. Following, other processing steps may occur: the perceptual quality of the image may be improved using enhancement operators, or the semantic content may be altered by inserting or removing objects. The image may also be replicated in several different versions, with different size, colors and file format. In this heterogeneous and multiform scenario, image forensics try to investigate the past of an image, without knowing anything but the image itself [2]. As this discipline evolved, different kinds of investigations have become possible; in the following we list those that received the greatest attention:

- *Source classification*, whose goal is to determine whether the image comes from a camera, a scanner, a mobile phone or it has been generated using computer graphics [10]. Techniques also exist allowing to discriminate between different camera brands or even models [11].

- *Source identification*, which differs from the previous class in that the exact device needs to be identified: given a group of cameras of the very same brand and model, the goal is to understand which one was used to capture the image. We can think of this as something similar to gun ballistics, which tries to understand whether a bullet has been fired by a specific gun: source identification is indeed known also as "camera ballistics" [12].

- *Reverse engineering of processing operators*, which aims at investigating the whole digital history of the image, for example detecting whether a specific processing operator, or even a chain of processing operators, has

been applied, possibly estimating their parameters and the chain order. This branch of forensics is among the most difficult ones, because traces left by different processing operations tend to alter of even cancel each other.

- *Authenticity verification*, whose goal is to understand whether the image has been maliciously manipulated so to alter its content, for example by pasting pixels from another image (so called "cut-&-paste" attack) or even from the image itself ("copy-move" attack).

- *Image philogeny* where, starting from a set of near-duplicate images, the goal is to reconstruct the dependency tree, which means telling which image originated which, and whether two images belong to the same tree [13].

Such a wide range of applications suggests that there is more information in a digital image about its history than we think. The next section briefly sketches the main methods existing today for exposing such information.

2.2 Methods for forensic analysis of digital images

The basic idea underlying Image Forensics is that every step of the lifecycle of a digital image leaves subtle, often imperceptible, footprints into the image itself; the presence of these footprints can hence be investigated in order to gather information about the "digital history" of the image. In the last ten years, many techniques have been developed following this methodology, that can roughly be classified into the following classes, based on the kind of footprint they rely on:

- *Acquisition based footprints.* This class groups all those traces that are left by the acquisition device during capturing. For example, most cameras use a Color Filter Array (CFA) followed by an interpolation filter to produce a color image starting from a single light sensor. As a consequence, the digital image brings traces of both the employed CFA pattern and the type of interpolation filter [14]. Moreover, the light sensor itself leaves a characteristic noise in each captured image,

known as Photoresponse Non-Uniformity Noise (PRNU) that is unique for that specific sensor, so that no two sensors exist leaving the same noise pattern [12].

- *Coding based footprints.* The vast majority of digital images is stored using the lossy JPEG algorithm, which processes the image in a block-wise fashion leaving characteristic artifacts both perceptually and statistically. While such artifacts are typically considered as an annoying consequence of lossy coding, they become a useful asset for assessing the processing history of an image. This becomes even more true because multiple compression steps may leave incremental traces and, what is more interesting, when the image is created by splicing together different pictures it is possible to distinguish between pixel blocks compressed once, twice and so on [15].

- *Editing based footprints.* This kind of footprints are left during image processing operations: manipulating images is getting easier thanks to the aid of powerful editing softwares, but there is a wide range of imperceptible inconsistencies that may be left into an image during processing. For example, by barely resizing or rotating an image we are forcing the software to interpolate pixel values, thus leaving detectable traces within the digital signal [16]. Moreover, by pasting a patch of pixels we introduce inconsistencies in the blurriness, contrast and saturation of different areas of the same picture [17, 18].

- *Geometric and physics based footprints.* When creating a fake image, one of the most challenging tasks is to arrange objects in such a way that their geometric and physical properties remain consistent. In practice, adapting the size, perspective, illumination and shadowing of an object is not trivial even for the most clever photo editor. This especially holds because our brain is not precise in computing vanishing points, adjusting proportions and so on. As a consequence, geometric and physic footprints have been successfully applied in image forensic [19, 20]. The most attracting feature of these kind of footprints is their robustness, as they remain present even after printing or recapturing[2].

[2]Image recapture consists in printing or displaying the image on a screen and then taking

The above list is not exhaustive; however the literature is rich of books [1, 21] and overview papers [2]. As stated in the Introduction of the thesis, our contribution is focused on authenticity verification, and more precisely on splicing detection. Indeed, this branch was (and still is) one of the most flourishing in image forensics: new methods are continuously devised, belonging to different classes among those described above. It is important to stress that there is no direct link between goals and methods, meaning that the same footprint can be leveraged on for different tasks; interestingly, splicing detection is by far the application that benefits most from the analysis of different kinds of footprints: for example, PRNU was initially studied as a tool for source identification, but it rapidly became one of the preferred means for authenticity verification [12]. Similarly, the presence of CFA demosaicing artifacts has been proposed initially for source classification [11], but it inspired tools for authenticity verification [14] as well.

2.3 The importance of a synergic analysis

The discussion above suggests splicing detection as a favourable scenario for studying the benefits that could be obtained by using several tools together in a synergic way. Besides the availability of different analysis methods, the intrinsic nature of image tampering also favours its choice as a case study for the role of data fusion in image forensics: in most cases, the creation of a forgery involves the application of more than a single processing tool, thus leaving a number of traces that can be used to detect the presence of tampering. We should also consider one more fact: while it may be easy for a skilled, possibly "forensic aware" image retoucher to conceal some traces of his work, it would be far more difficult for him to fool an heterogeneous set of analysis tools that account for many different traces. That is to say: chances for the analyst to reveal the manipulation increase significantly when many different clues are put together, making the "perfect crime" much harder to accomplish.

These considerations suggests to analyze the authenticity of images by using more than one tamper detection tool. Furthermore, existing forensic

a new shot of it with a digital camera.

tools are far from ideal and often give uncertain or even wrong answers, so, whenever possible, it is wise to employ more than one tool searching for the same trace. On top of that, it may also be the case that the presence of one trace inherently implies the absence of another, because the traces are mutually exclusive by definition: for example, theoretically a block of pixels cannot show traces of both aligned and not-aligned double compression. For these reasons, taking a final decision about the authenticity of an image relying on the output of a set of forensic tools is not a trivial task, thus justifying the design of proper decision fusion methods explicitly thought for this scenario.

2.3.1 Decision fusion in image forensics: possible approaches

The problem of taking a final decision about a hypothesis by looking at the output of several tools is an important task in decision fusion. There are basically three kinds of approaches to tackle with it. The first is to perform fusion at the *feature* level: a subset of the features extracted by the tools is selected and used to train a global classifier. The second is to consider the (usually scalar) output provided by the tools and fuse them (fusion at the *measurement*, or *score*, level). The last approach consists in fusing the binary answers of the tools, usually obtained by binarizing their soft outputs (fusion at the *abstract* level). An effective example of how these three strategies can be applied to a problem similar to the one addressed in this thesis is illustrated in the work by Kharrazi et al. [22], where fusion is used in a steganalysis framework. In fact, both in steganalysis and image forensics, tools usually extract some features from the image, perform measurements/classification by relying on them and finally produce an output, often probabilistic, which can be thresholded to yield a binary classification.

Although being promising in terms of performance, fusion at the feature level has some serious drawbacks, most importantly the difficulty of handling cases involving a large number of features (commonly addressed as "curse of dimensionality") and the difficulty to define a general approach to feature selection, since ad-hoc solutions are needed for different cases. Furthermore, feature selection in most cases is followed by some machine learning, that by definition is effective only when a training dataset can be prepared that is representative of a large part of the global population of samples. If this can

be done for training a single detector, creating a representative dataset of all possible image forgeries is practically unfeasible, especially in the case of photorealistic ones (assuming they must be created by a human).

Working at the other extreme, the abstract level, suffers from the complementary problem: a large amount of information is discarded when outputs are thresholded, so the discrimination power of the various tools is not fully exploited.

In image forensics, most of the existing works are based on feature level fusion [23] [24] [25]; a hybrid solution has been investigated by Bayram et al. [26], but still focusing on feature fusion. Noticeably, a fusion approach based on the same mathematical background we will build upon, that is Dempster-Shafer Theory, was proposed by Zhang et al. [27]. Also in that work, however, fusion is taken at the feature level hence inheriting the general drawbacks of such an approach, noticeably the lack of scalability and the need to retrain the whole system each time a new tool is added.

Fusion at the measurement level is a solution that gets around the above problems: the responsibility of selecting features and training classifiers (or other decision methods) is delegated to each single tool, thus favoring generality and expandability; at the same time, the loss of important information about tool response confidence is at least partially prevented. Fusion at the measurement level has been proposed in [28], where Sun et al. exploit Dempster's combination rule to devise an image steganalysis scheme that combines three algorithms to improve detection accuracy. As it will become evident in the next Chapter, Dempster's combination rule is an important tool of DST, but it is not sufficient alone to devise a framework that generalizes well to heterogeneous sets of tools. Finally, Costanzo et al. [29] recently proposed a framework based on fuzzy logic that combines tool outputs at the measurement level. While the framework in [29] is conceptually similar to the one proposed in this work, it requires the user to manually set a rather large set of parameters.

Chapter 3

A Dempster-Shafer Framework for Splicing Detection

IN this chapter we devise a framework for combining at the measurement level the evidence coming from two or more forgery detection algorithms. The goal is not to develop a specific multi-clue forgery detection tool, but to define a theoretical model that allows fusing a generic set of forensic tools, requiring as little prior information as possible. Hence, we rely on Dempster-Shafer Theory of Evidence (DST) as the basic mathematical tool, since this theory can model uncertainty and missing information in a very intuitive and sound way, and does not force to specify prior probabilities in the modeling phase. The proposed framework exploits knowledge about reliability of tools and about compatibility between different traces of tampering, and can be easily extended when new tools become available. It allows both a "soft" and a binary (tampered/non-tampered) interpretation of the fusion result, and can help in analyzing images for which taking a decision is critical due to conflicting data.

To test the effectiveness of the framework, we apply it to the splicing detection problem, which consists in determining whether a given region of an image has been pasted from another. During this process some traces are left into the image, depending on the modality used to create the forgery: the presence of each of these traces can be revealed by using one (or more) image forensic tools, each of which provides information about the presence of the trace it is looking for. Note that, in splicing *detection*, most forensic tools assume knowledge of the suspect region; that said, if no information is available, we could still run all tools in a block-wise fashion, and fuse their outputs at the block level.

The chapter is structured in two main parts: the first one (Section 3.1) gives a brief introduction to DST, while the second (Section 3.2) presents

the proposed framework. The experimental validation of the framework is addressed in Chapter 4.

3.1 Introduction to Desmpter-Shafer Theory of Evidence

Dempster-Shafer Theory of Evidence (DST) [30, 5] is a widely employed mathematical framework allowing to make reasoning and inference in contexts where uncertainty and lack of information are strong. Indeed, one of the most attractive features of DST is the capability of modeling uncertainty and doubt in an explicit and very simple way, especially compared to classical probability theory. When using classical probability theory for defining the probability of a certain event A, the additivity rule must be satisfied; so by saying that $Pr(A) = p_A$ one is also implicitly saying that $Pr(\bar{A}) = 1 - p_A$, thus committing the probability of an event A to that of its complementary $\bar{A}$. Most importantly, the additivity rule influences also the representation of ignorance: complete ignorance about a dichotomic event A in Bayesian theory is commonly represented by setting $Pr(A) = Pr(\bar{A}) = 0.5$ (according to the maximum entropy principle), but this probability distribution also models perfect knowledge about the probability of each event being 0.5 (as for coin tossing), thus making it difficult to distinguish between ignorance and perfectly known equiprobable events. Since reasoning in a Bayesian framework makes an extensive use of prior probabilities, which are often unknown, a wide usage of maximum entropy assignments is often unavoidable, leading to the introduction of extraneous assumptions. To avoid that, DS theory abandons the classical probability framework and allows to reason without the need to introduce a-priori probabilities. This further motivate the choice of using DST for developing our decision fusion framework, since in the image forensic field it is very difficult to get good estimates of prior probabilities.

3.1.1 Shafer's formalism

Let the frame $\Theta = \{x_1, x_2, \ldots, x_n\}$ define a finite set of possible values of a variable X; a proposition about X is any subset of Θ. We are interested in quantifying how much we are confident in propositions of the form "the

true value of X is in H", where $H \subseteq \Theta$ (notice that the set of all possible propositions is the power set of Θ, 2^Θ). To give an example, let us think of a patient that can either be affected by cancer or not: we can model this scenario defining a variable C with frame $\Theta = \{ac, nc\}$ where ac is the proposition "patient is affected by cancer", nc is the proposition "patient is not affected by cancer", and $(ac \cup nc)$ is the doubtful proposition "patient is or is not affected by cancer". Properly choosing the set Θ is very important in DST: it must represent the desired granularity of information that we want to (or we can) reach, so that choosing $\Theta = \{\text{car}, \text{truck}, \text{bus}, \text{scooter}, \text{motorbike}\}$ may be less appropriate than choosing $\Theta = \{\text{four-wheeled}, \text{two-wheeled}\}$ for some tasks.

The link between propositions and subsets of Θ allows to map logical operations on propositions into operations among sets. Each proposition is mapped onto a single subset and is assigned a basic belief *mass* through a Basic Belief Assignment, defined over the frame of the variable.

Definition 1. *Let Θ be a frame. A function $m^\Theta : 2^\Theta \to [0, 1]$ is called a Basic Belief Assignment (BBA) over the frame Θ if:*

$$m^\Theta(\emptyset) = 0; \qquad \sum_{A \in 2^\Theta} m^\Theta(A) = 1 \tag{3.1}$$

where the summation is taken over every possible subset A of Θ.

Continuing the previous example, after examining the patient a doctor could provide information that lead us to write the following basic belief assignment:

$$m^\Theta(X) = \begin{cases} 0.8 & \text{for } X = \{ac\} \\ 0.2 & \text{for } X = \{nc\} \\ 0 & \text{for } X = \{ac \cup nc\} \end{cases}. \tag{3.2}$$

Each set S such that $m^\Theta(S) > 0$ is called a *focal element* for m. In the following, we will omit the frame when it is clear from the context, writing m instead of m^Θ; furthermore, when writing mass assignments only focal elements will be listed (so the last row of eq. (3.2) would not appear). BBAs are the atomic information in DST, much like probability of single events in

probability theory. By definition, $m(A)$ is the part of belief that supports exactly A but, due to lack of knowledge, does not support any strict subset of A, otherwise the mass would "move" into the subsets. In the previous example, if we had assigned mass 0.85 to proposition $\{ac \cup nc\}$ and 0.15 to $\{ac\}$ it would have meant that there is some evidence for the patient being affected by cancer but, based on current knowledge, a great part of our confidence cannot be assigned to none of the two specific propositions. Whenever we have enough information to assign all of the mass to singletons[1], DST collapses to probability theory.

Intuitively, if we want to obtain the total belief for a set A, we must add the mass of all proper subsets of A plus the mass of A itself, thus obtaining the *Belief* for the proposition A.

Definition 2. *Given the BBA in 1, the Belief function* $Bel : 2^{\Theta} \rightarrow [0, 1]$ *is defined as follows:*

$$Bel(A) = \sum_{B \subseteq A} m(B).$$

$Bel(A)$ summarizes all our reasons to believe in A based on the available knowledge. There are many relationships between $m(A)$, $Bel(A)$ and other functions derived from these; here we just highlight that $Bel(A) + Bel(\bar{A}) \leq 1$ $\forall A \subseteq \Theta$ and $1 - (Bel(A) + Bel(\bar{A}))$ is the lack of information (or the amount of doubt) about A.

3.1.2 Combination rule

If we have two BBAs defined over the same frame, which have been obtained from two independent sources of information, we can use Dempster's *combination rule* to merge them into a single one. Notice that the concept of independence between sources in DST is not rigorously defined (as it is, for example, in Bayesian theory): the intuition is that different pieces of evidence must have been determined by different (*independent*) means [31].

Definition 3. *Let* Bel_1 *and* Bel_2 *be belief functions over the same frame* Θ *with BBAs* m_1 *and* m_2*. Let us also assume that* K*, defined below, is smaller*

[1] A singleton is a set with exactly one element.

than 1. Then for all non-empty $X \subseteq \Theta$ the function m_{12} defined as:

$$m_{12}(X) = \frac{1}{1-K} \cdot \sum_{\substack{A,B \subseteq \Theta: \\ A \cap B = X}} m_1(A) m_2(B) \tag{3.3}$$

where $K = \sum_{A,B:A\cap B=\emptyset} m_1(A)m_2(B)$, is a BBA function defined over Θ and is called the orthogonal sum of Bel_1 and Bel_2, denoted by $Bel_1 \oplus Bel_2$.

K measures the *conflict* between m_1 and m_2: the higher the K, the higher the conflict. The meaning of K can be understood from its definition, since it is obtained by accumulating the product of masses assigned to sets having empty intersection (which means incompatible propositions). Furthermore, we see that Dempster's combination rule treats conflict as a normalization factor: in practice, this means that the amount of conflicting evidence is proportionally "redistributed" to non-conflicting propositions. Later in this chapter, it will become evident that such a way of handling conflicting evidence can lead to counter-intuitive results in some cases.

Recall the example in (3.2), and suppose that we obtain evidence coming from another doctor, who is not a cancer specialist, about the variable C. Let us call m_1 the BBA in eq. (3.2) and m_2 the new assignment; so we have:

$$m_1(X) = \begin{cases} 0.8 & \text{for } X = \{ac\} \\ 0.2 & \text{for } X = \{nc\} \end{cases} \qquad m_2(X) = \begin{cases} 0.1 & \text{for } X = \{ac\} \\ 0.9 & \text{for } X = \{ac \cup nc\} \end{cases}.$$

Since the second doctor is not a specialist the information he provides is quite limited: most of the mass is assigned to doubt. Fusing the two pieces of information according to Dempster's rule results in:

$$m_{12}(X) = \begin{cases} \frac{0.8\cdot 0.1 + 0.8\cdot 0.9}{1-(0.1\cdot 0.2)} = 0.816 & \text{for } X = \{ac\} \\ \\ \frac{0.2\cdot 0.9}{1-(0.1\cdot 0.2)} = 0.184 & \text{for } X = \{nc\} \end{cases}.$$

We see that after fusion values are not far from those already assigned by m_1: this is perfectly intuitive, since the second doctor did not bring a clear contribution to the diagnosis. Notice also that for the same reason, and for the low confidence of the first doctor about absence of cancer, little conflict is observed ($K = 0.02$).

Dempster's rule has many properties [32]; in this work we are mainly interested in its associativity and commutativity, that is:

$$Bel_1 \oplus (Bel_2 \oplus Bel_3) = (Bel_1 \oplus Bel_2) \oplus Bel_3 \tag{3.4}$$

$$Bel_1 \oplus Bel_2 = Bel_2 \oplus Bel_1. \tag{3.5}$$

Despite its desirable properties, Dempster's rule is not idempotent; this means that observing twice the same evidence results in stronger beliefs. This is the reason why we need to introduce the hypothesis of independent sources in Dempster's combination rule. In practice, before letting a new source of information enter the system, we must always look at how the new information is collected, to ensure that we are not counting twice the same evidence. In our example, we must be sure that doctors did not talk with each other, did not use the same technology when performing measurements, and so on. Before moving to the next topic, it is worth to spend some words about the way Dempster's rule manages conflicting evidence. When we combine evidence using Dempster's rule, it is assumed that masses are assigned by reliable sources, acting like oracles: the responsibility of declaring doubt is demanded to sources. When this fact is not properly accounted for, it becomes easy to reach counter-intuitive conclusions. One noticeable example of this is Zadeh's paradox [33], that can be explained by re-visiting our example: let us suppose that two doctors provide the following BBAs, where $\{ac\}$ is the proposition "patient is affected by cancer", $\{ap\}$ is the proposition "patient is affected by pneumonia" and $\{af\}$ is the proposition "patient is affected by flu":

$$m_1(X) = \begin{cases} 0.9 & \text{for } X = \{ac\} \\ 0.1 & \text{for } X = \{ap\} \\ 0 & \text{for } X = \{af\} \end{cases} \qquad m_2(X) = \begin{cases} 0 & \text{for } X = \{ac\} \\ 0.1 & \text{for } X = \{ap\} \\ 0.9 & \text{for } X = \{af\} \end{cases}$$

Not surprisingly, using Demspter's rule to merge the above assignments results in a strong conflict ($K = 0.99$), and the merged BBAs is:

$$m_{12}(X) = \begin{cases} 0 & \text{for } X = \{ac\} \\ 1 & \text{for } X = \{ap\} \\ 0 & \text{for } X = \{af\} \end{cases},$$

meaning that, based on available knowledge, the patient is affected for sure by pneumonia. Such a result is counter-intuitive at a first glance: pneumonia was assigned only a 0.1 mass by both doctors, but after fusion it becomes a certainty. Yet, if we think about doctors as oracles, as Dempster's rule does, then it becomes evident that pneumonia is the only possibility, because both flu and cancer had been excluded, respectively, by Doctor 1 and Doctor 2. Quoting A. C. Doyle, the rationale is that "when you have eliminated the impossible, whatever remains, however improbable, must be the truth" [34]. In the example, doctors were totally resolved in excluding cancer and pneumonia (assigning a zero mass equals to declare the proposition impossible), and only flu is left as a possibility by both of them.

Although many different combination rules have been proposed treating conflict in a more cautious way, Dempster's rule is safe to use as long as sources of information are modeled taking into account their intrinsic restrictions; in the end, if an oracle were available, information fusion would not be useful at all.

3.1.3 Belief marginalization and extension

The combination rule expressed in (3) is applicable if the two BBAs, m_1 and m_2, are defined over the same frame, which means that they refer to the same propositions. Whenever we need to combine BBAs defined over different frames, we have to redefine them on the same target frame before the combination. This can be done by using *marginalization* and *vacuous extension*.

Definition 4. *Let m^Θ be a BBA function defined over a frame Θ, and let Ω be another frame. The vacuous extension of m^Θ to the product space $\Theta \times \Omega$, denoted with $m^{\Theta\uparrow\Theta\times\Omega}$, is defined as:*

$$m^{\Theta\uparrow\Theta\times\Omega}(X) = \begin{cases} m^\Theta(A) & \text{if } X = A \times \Omega, A \subseteq \Theta \\ 0 & \text{otherwise} \end{cases}$$

This allows to extend the frame of a BBA without introducing extraneous assumptions (no new information is provided about propositions that are not in Θ). That said, vacuous extension is not the only possible way to extend a BBA to a larger frame: it just provides the "least informative" extension.

The inverse operation of vacuous extension is marginalization.

Definition 5. *Let m^{Θ} be a BBA function defined on a domain Θ; its marginalization to the frame $\Gamma \subseteq \Theta$, denoted with $m^{\Theta\downarrow\Gamma}$, is defined as*

$$m^{\Theta\downarrow\Gamma}(X) = \sum_{A\downarrow X} m^{\Theta}(A)$$

where the index of the summation denotes all sets $A \subseteq \Theta$ whose projection on Γ is X.

To define the projection operator, let us introduce two product frames Θ and Γ, that are obtained as the cartesian product of the frames of some variables. Formally, we have $\Theta = F_1 \times F_2 \times \cdots \times F_k$ and $\Gamma = F_{S_1} \times F_{S_2} \cdots \times F_{S_z}$, where F_j is the frame of the j-th variable and S is a subset of the indices in $\{1, \ldots, k\}$. Each element of Θ will be a vector whose j-th component is a value in F_j. For instance, if $\Theta = X \times Y \times Z$ one possible element of Θ is (x_1, y_3, z_1), where $x_1 \in X$, $y_3 \in Y$ and $z_1 \in Z$. The projection operator maps each element $\theta \in \Theta$ into an element of $\gamma \in \Gamma$ by removing from θ all the components whose indices are not in S. For example, if we project the set $\Theta = X \times Y \times Z$ onto $\Gamma = X \times Z$ the element $(x_1, y_3, z_1) \in \Theta$ reduces to $(x_1, z_1) \in \Gamma$. The importance of extension and marginalization is that they allow to combine over a common frame BBAs originally referring to different frames, hence enabling us to fuse them with Dempster's rule.

3.2 The proposed framework

As we try to use the theoretical tools provided by DST to develop a framework for decision fusion, there are two main aspects that need to be discussed. The first one is how to fruitfully combine the information provided by different tools, once it is written in terms of Basic Belief Assignments. When we considered the toy problem of fusing information coming from two doctors (Section 3.1.2), we assumed they would directly provide their knowledge in the form of consistent BBAs. Now that we are dealing with image forensic algorithms, the way their output is mapped to BBAs becomes of fundamental importance, and needs to be carefully discussed. Therefore, the second fundamental aspect that needs to be considered is how to convert the output

provided by analysis instruments into Basic Belief Assignments. Fortunately these two topics can be treated separately, and we take advantage of this in the following: we first deal with the problem of merging the information provided by different tools, assuming it is already written in terms of BBAs. Then, starting from Section 3.3, we focus on the problem of mapping tool outputs into BBAs.

3.2.1 Modeling forensic tools and traces using DST

We now formalize the problem of merging the information that comes from different image forensic tools. To begin with, we clarify some of the terminology: we will talk about *tools* searching for forensic *traces*. With "tool" we mean an algorithm that, given an image and a suspect region, performs a set of operations aiming at detecting the presence of a forensic trace. With "trace", we mean a property that may be present, either in the suspect region or in the rest of the image, and that can be possibly searched for in different domains (e.g., the DCT or the pixel domain). For example, when an image undergoes two JPEG codings with misaligned quantization grids, some artifacts are left both at the pixel level (inconsistent blocking artifacts) and in the DCT domain (double quantization of DCT coefficients). It becomes evident that different tools can be devised to search for the same trace in different domains, and that fusing their outputs can potentially improve the accuracy. Of course, we also have tools searching for different traces: in these cases, we will take advantage of knowing the compatibility relationships between traces (for example, presence of one trace may imply absence of another). We can now state the two basic assumptions beneath the proposed framework for decision fusion:

- Compatibility relations among some or all the considered traces are known (for instance, we may know that two tools search for mutually-exclusive traces);
- Each tool gathers information independently of other tools (i.e., a tool is never employed as a subroutine of another, and no information is exchanged between tools), and by different means (each tool relies on a different principle or effect);

These assumptions are very reasonable in the current image forensics scenario. However, as it will be shown (Section 3.2.4) the first one can be relaxed arbitrarily, at the cost of a lower performance gain with respect to the use of single tools alone. The second assumption, instead, is needed to ensure that we can fuse tool responses using Dempster's rule. Intuitively, it means that if we observe two different tools supporting the same proposition, we are more confident than observing only one. On the other hand, if two tools searching for the same trace by exploiting the same model are available, it makes sense to discard the less reliable one, since its contribution is limited or null. That said, and also considering that the concept of independence in DST is not equivalent to statistical independence, we believe that possible limited dependencies between algorithms would not undermine the developed framework.

Formalization for a single tool

For sake of clarity, we start by formalizing the DST framework when only one tool is available, let us call it *ToolA*, which searches for a forensic trace called α and outputs a scalar value A. The key idea is to treat *ToolA* as a source of information about the presence of the trace it is looking for. To this aim, we define for α the frame $\Theta_\alpha = \{t\alpha,\ n\alpha\}$, where $t\alpha$ is the proposition "trace α is present" and $n\alpha$ is the proposition "trace α is not present". We model the information provided by *ToolA* about the presence of α with the following BBA over the frame Θ_α:

$$m_A^{\Theta_\alpha}(X) = \begin{cases} A_{\mathrm{T}} & \text{for } X = \quad \{(t\alpha)\} \\ A_{\mathrm{N}} & \text{for } X = \quad \{(n\alpha)\} \\ A_{\mathrm{TN}} & \text{for } X = \{(t\alpha) \cup (n\alpha)\} \end{cases} \tag{3.6}$$

where A_{T}, A_{N} and A_{TN} are functions mapping the response of the tool A into mass assignments; as anticipated, the definition of these functions will be the subject of Section 3.3. We see that this BBA assigns a mass to every element of the power set of Θ_α; $\{(\mathrm{t}\alpha) \cup (n\alpha)\}$ is the doubt that *ToolA* has about the presence of the trace, so it refers to the proposition "trace α is either present or not". It is worth to remark the importance of allowing tools expressing lack of certainty, especially in the image forensic field were no algorithm exists

that is highly reliable under any possible situation. If every tool only declares the actual degree of confidence about presence of the searched trace, this will make its contribution more valuable when it comes to be merged with others. Failing to do so, on the contrary, may result in counterproductive behaviors.

3.2.2 Introducing new tools

Suppose we want to introduce in our framework a new tool, *ToolB*, that satisfies the assumptions given at the beginning of this section. As we anticipated, two situations are possible: the new tool may either search for a trace that is already considered in the framework, or for a novel trace. Since Dempster's combination rule allows fusing only BBAs that are defined over the same frame of discernments, these two possible cases are addressed differently.

Introduction of a tool looking for a known trace If the trace searched by the new tool is already present in the framework (let us call it α, consistently with Section 3.2.1), application of the procedure in Section 3.2.1 will produce $m_B^{\Theta_\alpha}$, which can be directly fused with $m_A^{\Theta_\alpha}$ by using Dempster's rule, yielding:

$$m_{AB}^{\Theta_\alpha}(X) = \frac{1}{1-K} \cdot \begin{cases} A_{\mathrm{T}}B_{\mathrm{T}} + A_{\mathrm{T}}B_{\mathrm{TN}} + A_{\mathrm{TN}}B_{\mathrm{T}} & \text{for } X = \{(t\alpha)\} \\ A_{\mathrm{N}}B_{\mathrm{N}} + A_{\mathrm{N}}B_{\mathrm{TN}} + A_{\mathrm{TN}}B_{\mathrm{N}} & \text{for } X = \{(n\alpha)\} \\ A_{\mathrm{TN}}B_{\mathrm{TN}} & \text{for } X = \{(t\alpha) \cup (n\alpha)\} \end{cases} \tag{3.7}$$

where $K = A_{\mathrm{T}}B_{\mathrm{N}} + A_{\mathrm{N}}B_{\mathrm{T}}$. This BBA contains the information about the trace α gathered by the two distinct tools. We see that conflict is non-null, and is obtained by summing the masses for propositions in which the tools provide conflicting information about the presence of the trace. It is worth repeating that before introducing a new tool into the framework, the user should understand how the tool works and ensure that it does not replicate the investigation of a tool that is already present, since this would violate the request of independence of sources, and lead to a overestimation of the presence of the considered trace.

Introduction of a tool looking for a new trace If $ToolB$ searches for a novel kind of trace, say β, we have to introduce it into the framework by defining a new frame $\Theta_\beta = \{t\beta, n\beta\}$, where the propositions have the same meaning as in (3.6). The response of $ToolB$ will be used to assign masses to the variable Θ_β, leading us to $m_B^{\Theta_\beta}$. Since α and β are defined over different frames, $m_A^{\Theta_\alpha}$ and $m_B^{\Theta_\beta}$ cannot be fused directly. We need to introduce the common frame $\Theta_{\alpha\beta} = \Theta_\alpha \times \Theta_\beta$, so that we can (vacuously) extend both m_A and m_B to it, yielding:

$$m_{\mathrm{A}}{}^{\Theta_\alpha \uparrow \Theta_{\alpha\beta}}(X) = \begin{cases} A_{\mathrm{T}} & \text{for } X = \{(t\alpha, t\beta) \cup (t\alpha, n\beta)\} \\ A_{\mathrm{N}} & \text{for } X = \{(n\alpha, t\beta) \cup (n\alpha, n\beta)\} \\ A_{\mathrm{TN}} & \text{for } X = \{(t\alpha, t\beta) \cup (t\alpha, n\beta) \cup (n\alpha, t\beta) \cup (n\alpha, n\beta)\} \end{cases} \tag{3.8}$$

$$m_{\mathrm{B}}{}^{\Theta_\alpha \uparrow \Theta_{\alpha\beta}}(X) = \begin{cases} B_{\mathrm{T}} & \text{for } X = \{(t\alpha, t\beta) \cup (n\alpha, t\beta)\} \\ B_{\mathrm{N}} & \text{for } X = \{(t\alpha, n\beta) \cup (n\alpha, n\beta)\} \\ B_{\mathrm{TN}} & \text{for } X = \{(t\alpha, t\beta) \cup (n\alpha, t\beta) \cup (t\alpha, n\beta) \cup (n\alpha, n\beta)\} \end{cases} . \tag{3.9}$$

Equations (3.8) and (3.9) show what "vacuous extension" means in practice: for example, in the first line of (3.8) the mass A_{T} is assigned to the set $\{(t\alpha, t\beta) \cup (t\alpha, n\beta)\}$, which is the proposition "trace α is present, regardless of trace β". As expected, the mass assigned by $ToolA$ does not bring any information about β. Application of Dempster's combination rule to these two BBAs gives us the desired combination:

$$m_{\mathrm{AB}}{}^{\Theta_{\alpha\beta}}(X) = \begin{cases} A_{\mathrm{T}}B_{\mathrm{T}} & \text{for} X = \{(t\alpha, t\beta)\} \\ A_{\mathrm{T}}B_{\mathrm{N}} & \text{for} X = \{(t\alpha, n\beta)\} \\ A_{\mathrm{T}}B_{\mathrm{TN}} & \text{for} X = \{(t\alpha, t\beta) \cup (t\alpha, n\beta)\} \\ A_{\mathrm{N}}B_{\mathrm{T}} & \text{for} X = \{(n\alpha, t\beta)\} \\ A_{\mathrm{N}}B_{\mathrm{N}} & \text{for} X = \{(n\alpha, n\beta)\} \\ A_{\mathrm{N}}B_{\mathrm{TN}} & \text{for} X = \{(n\alpha, t\beta) \cup (n\alpha, n\beta)\} \\ A_{\mathrm{TN}}B_{\mathrm{T}} & \text{for} X = \{(t\alpha, t\beta) \cup (n\alpha, t\beta)\} \\ A_{\mathrm{TN}}B_{\mathrm{N}} & \text{for} X = \{(t\alpha, n\beta) \cup (n\alpha, n\beta)\} \\ A_{\mathrm{TN}}B_{\mathrm{TN}} & \text{for} X = \{(t\alpha, t\beta) \cup (t\alpha, n\beta) \cup (n\alpha, t\beta) \cup (n\alpha, n\beta)\} \end{cases} \tag{3.10}$$

Notice that, at this point, we are not considering whether traces α and β are compatible or not: we will take this information into account only later on, exploiting the associativity and commutativity of Dempster's rule. Consequently there is no reason why the two tools should be conflicting, as con-

firmed by the fact that $K = 0$ in the above formula, since they are searching for traces that are considered "unrelated".

The procedures in Section 3.2.2 can be repeated when another tool *ToolX* becomes available. The associativity of Dempster's rule, defined in eq. (3.4), allows to combine directly the BBA $m_{X_{tot}}$ of the new tool with the one currently available (that takes into account all the tools already in the framework), so we will always need to extend the frame of, at most, two BBAs: this is a considerably smaller effort with respect to extending the BBA and computing the combination rule for all the tools.

Notice that using traces as basic entities instead of tools responses improves the extendability of the framework: as a matter of fact, while new tools are being released quite often, many of them search for an already known trace; if this is the case, introducing a new tool is very simple since only its BBA has to be extended.

3.2.3 Managing configurations of tools

When combining multiple tools there is one practical fact that must be accounted for: it may happen that, for a given image, only part of the tools within the framework can be run, while others can't. For example, some tools may not be compatible with the image, due to its format, size, number of channels and so on. Is it possible for the analyst to handle these variants without increasing the complexity and extensiveness of the framework? DST offers a nice way to solve this issue. Given a set Θ, the following BBA, known as *vacuous* BBA, is the neutral element for Dempster's combination rule:

$$m_V^\Theta(X) = \begin{cases} 0 & \forall\, X \subset \Theta \\ 1 & \text{for } X = \Theta \end{cases}. \tag{3.11}$$

We see that $m_V^\Theta(X)$ is a valid BBA assigning all of the mass to the whole frame of discernment, which means that no knowledge is brought about elements of Θ. For any BBA m_X^Θ, we have:

$$m_X^\Theta \oplus m_V^\Theta = m_X^\Theta, \tag{3.12}$$

meaning that m_V^Θ can be fused an arbitrary number of times without modifying the available information.

Let us go back to our problem, and assume that there is a *ToolU*, searching for trace α, that cannot be run on the image under analysis. The analyst will simply write the following:

$$m_U^{\Theta_\alpha}(X) = \begin{cases} 0 & \text{for } X = \quad \{(t\alpha)\} \\ 0 & \text{for } X = \quad \{(n\alpha)\} \\ 1 & \text{for } X = \{(t\alpha) \cup (n\alpha)\} \end{cases}, \tag{3.13}$$

and use the framework without changes, *ToolU* will not contribute at all to the final belief about presence of trace α. In the extreme case where none of the tools available to the analyst can handle the image, we still obtain a valid BBA, telling that no information is available about searched traces. We point out that, despite its plainness, the above feature is not easily enabled by many machine learning techniques: every possible combination of tools would result in a different feature space, thus requiring a different training.

3.2.4 Modeling traces relationships

Up to this point we have considered traces as if they were unrelated to each other; as it will be shown in the following sections, this is usually not the case in real applications. This kind of information can contribute significantly to the joint interpretation of tools responses.

Suppose, for instance, that we have two traces α and β and that, due to their nature, only some of their combinations are possible. For example, it may be that the presence of α implies the absence of β, so, at least ideally, two tools searching for these traces should never detect tampering simultaneously. This information induces a *compatibility relation* between frames Θ_α and Θ_β, meaning that some of the elements of the cartesian product $\Theta_\alpha \times \Theta_\beta$ are impossible. Strictly speaking, these elements should have never entered the frame of discernment, because by definition such frame contains only *possible* propositions, (Section 3.1.1). However, since we may not know in advance which traces will be introduced in our framework, we need a way to include this knowledge only in the late stage of fusion, and update the frame accordingly. Fortunately, in DST we can easily model this information by using a standard belief assignment: we define a BBA on the domain $\Theta_\alpha \times \Theta_\beta$, that has only one focal set, containing the union of all propositions (i.e, combination

of traces) that are considered possible, while all others have a null mass. For example the following BBA:

$$m_{comp}(X) = \begin{cases} 1 & \text{for } X = \{(t\alpha, n\beta) \cup (n\alpha, t\beta) \cup (n\alpha, n\beta)\} \\ 0 & \text{for } X = \{(t\alpha, t\beta)\} \end{cases} \tag{3.14}$$

models the incompatibility between traces α and β. Once this BBA is specified, the simultaneous presence of α and β is no longer considered possible, and any evidence supporting it will be treated as conflicting information. Thanks to the commutative property of Dempster's combination rule, this BBA can be combined with those coming from traces in the final stage of fusion. In such a way, information about tools relationships are exploited only at the very end and hence do not hinder the extendability of the model.

Of course, we want to allow the analyst to specify traces relationships without forcing him to do that. However, the given formulation can be used also when the relationships between some of the traces are not known: it is sufficient not to put unknown propositions in the impossible set of m_{comp}, meaning that there is no clue against those propositions being possible.

The last step of the decision fusion process consists in fusing the compatibility BBA defined above with the BBA obtained combining evidences from all the available tools, yielding a global BBA m_{FIN}. Notice that in this last application of Dempster's rule all the conflict that may arise is due to incompatibilities between traces. Although this conflict is normalized away by Dempster's rule, the value of K can be recorded and used to evaluate how "unexpected" the output of tools were. Very high values of conflict may indicate that the image under analysis does not respect the working assumptions of one or more tools. The overall decision fusion approach described so far is summarized in Figure 3.1 for the case of two tools.

It is worth noting that we did not need to introduce a-priori probabilities about an image being original or forged, or prior probabilities of presence of traces: in a Bayesian framework, this would have been difficult to obtain.

3.2.5 Dealing with many traces: hierarchical modeling

Since the extension to novel traces is based on the cartesian product of sets, the number of variables in the framework grows exponentially with the num-

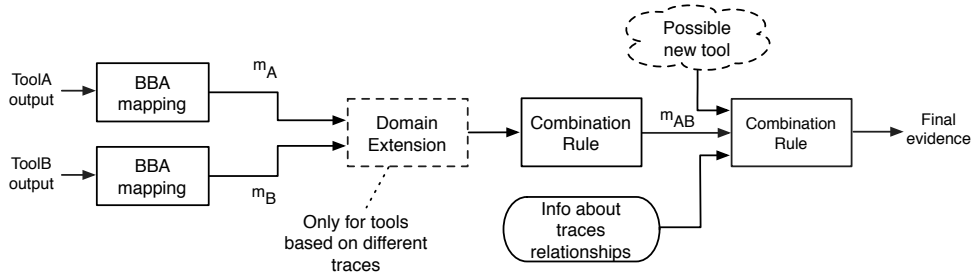


Figure 3.1: Block diagram of the proposed fusion approach. Notice that, when a new tool becomes available (represented in the dashed cloud), its BBA directly enters the final stage of the fusion, without the need to recombine information from previous tools.

ber of different traces. However, this consideration holds only if the user is interested in a fusion approach that fully preserves the granularity of information, meaning that, after fusing several different traces, the user wants to get the beliefs about presence/absence of each single trace separately. In practice, however, the presence of many traces is probably due to the fact that the framework is taking into account different classes of phenomena, e.g., traces related to camera artifacts, JPEG coding, geometrical inconsistencies, and so on. In such a scenario, it makes sense to treat each class of traces as a whole, and directly consider the contribution of each class when taking the final decision. This hierarchical fusion can be easily implemented within the proposed framework by using belief marginalization (see Definition 5) to collapse the contribution of several traces of the same class into a single variable, thus reducing the granularity of the information without hindering performance in terms of splicing detection. In Figure 3.2 we draw an example of hierarchical fusion applied to three different kinds of traces. Furthermore, compatibility among classes of traces can be introduced as well, at the end of the fusion chain.

3.2.6 Final decision rule

We are now ready to define the final output of the fusion procedure: we want to decide whether a given region of an image has been tampered with or not.

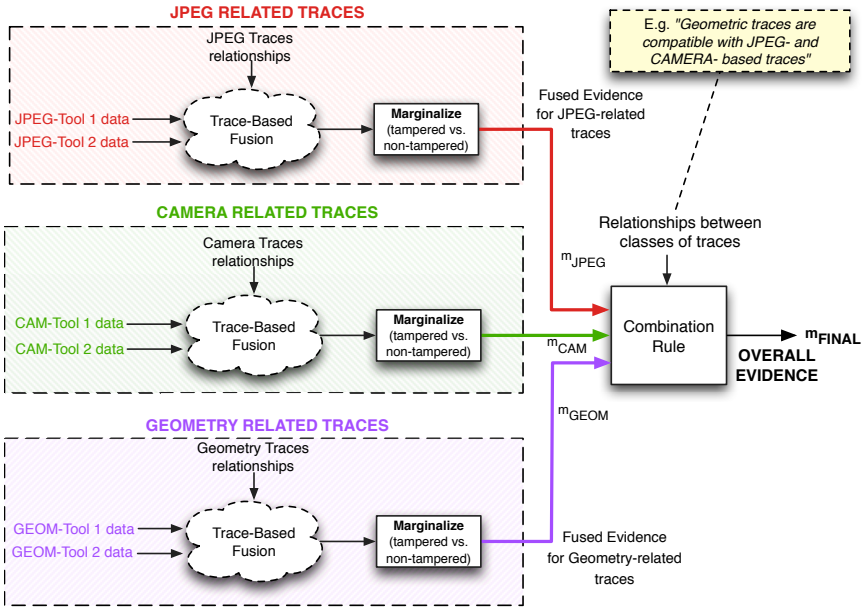


Figure 3.2: Block diagram illustrating the proposed approach to hierarchical fusion of traces of different kind. The "Trace-Based Fusion" bubble represents the schema in Figure 3.1.

To do so we consider the belief of two sets: the first one, T, is the union of all propositions in which at least one trace is detected, the second one, N, is the single proposition in which none of the traces is found (in the previous example it would be $N = (n\alpha, n\beta)$). The output of the fusion process therefore consists of two belief values, $Bel(T)$ and $Bel(N)$, calculated over the BBA m_{FIN} defined in Section 3.2.4. Optionally, we may also consider the normalization factor K (as defined in Section 3.1.2) of the last fusion step, involving the compatibility table. These outputs summarize the information provided by the available tools, without forcing a final decision. If a binary decision about image authenticity is required, an interpretation of these outputs must be made; the most intuitive binarization rule is to classify an image as tampered with when the belief for the presence of at least one trace is stronger than the belief for the total absence of traces, that is to say when $Bel(T) > Bel(N)$. Of course, we will probably want to meet a minimum distance requirement

between the two: a Receiver Operating Characteristic (ROC) curve can thus be obtained by classifying images according to $Bel(T) > Bel(N)+\delta$, sampling δ in [-1,1].

It is worth noting that evaluating belief values is a very simple task: only elementary operations among scalar values in [0,1] must be calculated (see for example mass assignments in equation (3.7)), since the model is built only once for a fixed set of tools, and need to be extended only when new sources of information become available.

3.3 From tool outputs to BBAs through background information

By now we have been discussing how to manage and fuse BBAs, assuming that they were provided by the forensic algorithms. Needless to say, this is not usually the case: since we are working at the measurement level, each tool is expected to output a scalar value that measures the presence of the trace within the image. To be used within the framework described in the previous section, this scalar output must be "mapped" to a BBA and, as it is shown in Figure 3.1, this must be done for each tool separately. From a formal point of view, denoting with $\mathcal{O}_i$ the set of possible outputs of the i-th forensic tool, searching for trace α, we want to derive a function

$$\mu_i : \mathcal{O}_i \to \mathcal{M}^{\Theta_\alpha}, \tag{3.15}$$

where $\mathcal{M}^{\Theta_\alpha}$ is the set of all possible BBAs defined on $\Theta_\alpha = \{t\alpha, n\alpha\}$.

Let us call o^i the output of the i-th tool, searching for trace α; without loss of generality, we can assume that higher values of the output indicate a stronger presence of the trace. Probably, the most intuitive way to map the output to a BBA is the following:

$$m_i(X) = \begin{cases} o^i & \text{for } X = \{(t\alpha)\} \\ 1 - o^i & \text{for } X = \{(n\alpha)\} \end{cases}, \tag{3.16}$$

which yields a valid BBA. Of course, this approach is very rigid, because it assumes a linear relationship between the output of the tool and the belief about the presence of the trace, which is false in most cases. Let us clarify

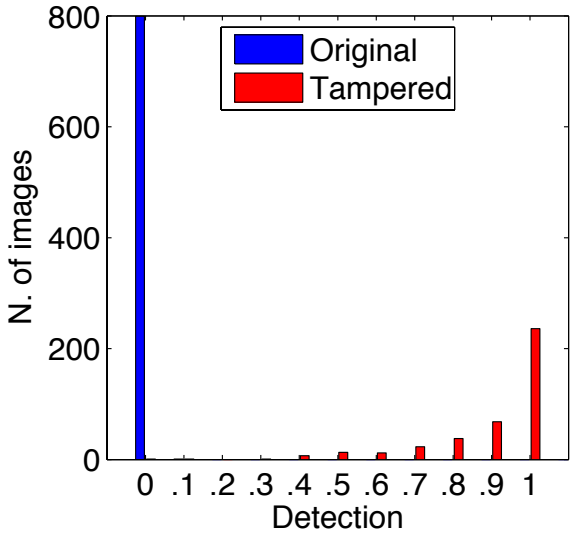


Figure 3.3: Histogram of outputs obtained by running the forensic tool in [35] on a set of images, some of which containing the forensic trace searched by the tool and some not. Red bars represent the outputs obtained on images containing the trace.

this point with an example: we ran the forensic tool[2] presented in [35] on a set of images, half of them containing the forensic trace and half of them not. Then, we collected the outputs separately for the two kinds of images and calculated their histograms, shown in Figure 3.3: as we can see, the output for images not containing the trace (blue bars) collapsed in the first bin, while outputs for the other class of images are more spread towards higher values. After seeing this picture, it is clearly wrong to interpret an output value of 0.5 as "uncertainty about the presence of the trace". The example above tells one trivial yet interesting thing: tool outputs must be properly *interpreted* before being merged with others. Here are several reasons for such a need: tools measure different things, and their output does not necessarily have a probabilistic meaning; when they are present, probabilistic models behind tools are often approximative (e.g., containing over-simplifications that lead to anomalous behaviors); finally, even the behavior of a fixed tool may vary

[2]This tool will be described later on in this chapter.

under different working conditions (e.g., when the analyzed region is very small or saturated). Therefore, passing from tool outputs to BBAs is not just a technical step, or a mere "conversion": it is a translation from a scalar measure to its interpretation in terms of belief about one proposition in the frame of discernment, which means, in our framework, presence or absence of the forensic trace. The question, then, moves to how such an interpretation should be made.

One possible solution is to use a kind of "fuzzy-reasoning", like in [29], defining for each tool some functions allowing to map the output to masses. Formally, this can be done by writing:

$$m_i(X) = \begin{cases} \mu_{\mathrm{T}}(o^i) & \text{for } X = \{(t\alpha)\} \\ \mu_{\mathrm{N}}(o^i) & \text{for } X = \{(n\alpha)\} \\ \mu_{\mathrm{TN}}(o^i) & \text{for } X = \{(t\alpha \cup n\alpha)\} \end{cases} . \tag{3.17}$$

In practice, functions μ_{T}, μ_{N} and μ_{TN} together form the mapping function mentioned in equation (3.15). Following the example provided above, these functions could be defined as in Figure 3.4, where only very low values of the output are interpreted as absence of the trace, values above 0.5 are interpreted as presence of the trace, and values between 0.1 and 0.4 are characterized by a fair amount of doubt. The doubt models the fact that, based on the experiment that originated the outputs in Figure 3.3, we do not know how values in that range should be interpreted.

There is no doubt that this mapping is much more appropriate than the trivial one defined in (3.16); this is confirmed by the fact that this method has been actually employed in the works by Costanzo et al. [29] and also in some of our works [36, 37]. Yet, one may object that this approach somewhat "hides" the problem inside the definition of mapping functions, still leaving much work to the user of the framework.

3.3.1 Interpretation of tool outputs based on DST

As an answer to the above issues, we adopted a different strategy based on DST itself. The idea behind the strategy is still to interpret the output of the tool based on its observed behavior on a suitable number of images, but to do that in a more refined way. Let us suppose that the analyst has, for a

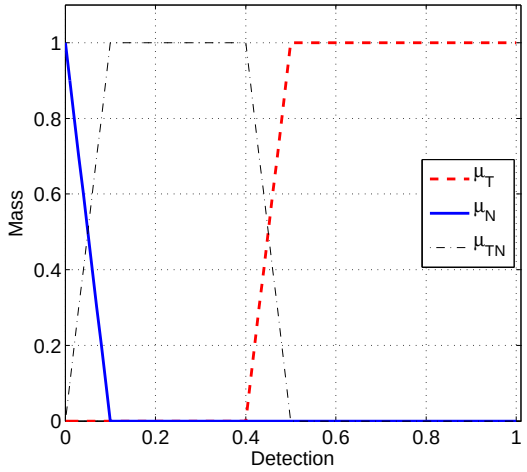


Figure 3.4: A possible way of defining output mapping functions for the forensic tool in [35].

given tool, a training set $\mathcal{T} = \{o^i : i = 1 \dots N\}$ of N training samples, where, for the i-th sample, o^i denotes the output of the tool. Each training sample belongs to one of the possible classes in $\mathcal{C} = \{C_0, C_1\}$, where C_0 is the class of images containing the searched trace, and C_1 the class of images without the trace.

As opposed to common classification problems, our goal here is not to assign an unseen sample u to one class in $\mathcal{C}$. Instead, we want to map it into a basic belief assignment over the frame Θ_α, reflecting the confidence of the tool about the presence of the searched trace. The key idea we build upon, that was first introduced in [38], is to model the elements of $\mathcal{T}$ as a source of evidence about u, and use Dempster's rule to pool the evidence. Intuitively, the closer a training sample is to u, the stronger will be the supporting evidence it provides, as shown in Figure 3.5.

Formally, let $\mathcal{T}_{u,k} \subset \mathcal{T}$ be the set of k training samples nearest to u according to some distance $d(\cdot, \cdot)$. Then, an element $t_i \in \mathcal{T}_{u,k}$ belonging to C_0

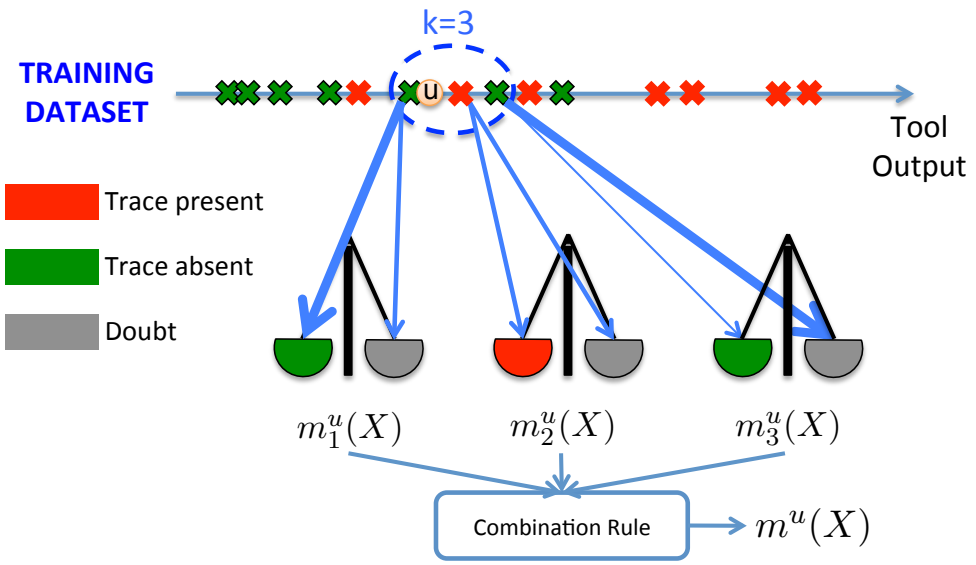


Figure 3.5: Graphic representation of the proposed method for mapping tool outputs to BBAs. The weight of each arrow connecting the unseen sample u one side of the balances model the amount of mass assigned to the corresponding proposition. As we can see, the farther the training sample is from u, the more mass goes to doubt.

provides the following BBA over Θ_α:

$$m_i^u(X) = \begin{cases} \beta e^{-\gamma d(u,t_i)} & \text{for } X = \{t\alpha\} \\ 1 - \beta e^{-\gamma d(u,t_i)} & \text{for } X = \{t\alpha \cup n\alpha\} \end{cases}, \tag{3.18}$$

where $\beta \in (0,1)$ denotes the maximum belief we commit to a single training sample, and γ controls the width of the kernel. On the contrary, an element t_i belonging to class C_1 provides:

$$m_i^u(X) = \begin{cases} \beta e^{-\gamma d(u,t_i)} & \text{for } X = \{n\alpha\} \\ 1 - \beta e^{-\gamma d(u,t_i)} & \text{for } X = \{t\alpha \cup n\alpha\} \end{cases}. \tag{3.19}$$

As to the distance function, a reasonable choice is

$$d(u,t_i) = ||u - t_i||^2,$$

provided that values are well distributed within a common interval, for example [0,1].

Equations (3.18) and (3.19) deserve a comment: a sample belonging to C_0 assigns some evidence to the proposition "u comes from an image containing the searched trace" and the rest of the evidence to the doubtful proposition "the image may or may not contain the trace". The same reasoning applies for samples belonging to C_1, as in equation (3.19). Notice that, when the unseen sample u is very far from t_i, this training sample will provide a BBA that is completely doubtful, instead of partitioning the mass between the two propositions $t\alpha$ and $n\alpha$. On the other hand, such a partitioning may occur after evidence pooling, when some of the k nearest neighbors belong to one class and some to the other, and they are all near to u. This situation means that the unseen sample lays in a "confused" part of the space: there are training samples near to it, but they belong to different classes.

Once the BBA assigned by each element in $\mathcal{T}_{u,k}$ has been calculated, we can use Demspter's combination rule to pool the evidence, yielding:

$$m^u(X) = \bigoplus_{i=1}^{k} m_i^u(X), \tag{3.20}$$

where $\oplus$ denotes the application of Dempster's orthogonal sum defined in (3.3) to all the m_i^u. The pooled BBA in (3.20) finally gives the desired interpretation of the tool output in terms of presence of the searched trace, based on training samples available to the analyst.

Compared to the simpler approaches proposed at the beginning of this section, the method above has a clear theoretical foundation that also serves as a guide for practical implementation, and requires virtually no input from the analyst (the parameters β, k and γ can be tuned with an automatic search). Yet, there is one aspect that is not directly considered, that is *tools reliability*. As it was discussed in Section 3.1, in DST theory it is of paramount importance to properly model the reliability of sources of information, so to avoid paradoxical situations. Looking back to equation (3.19) we notice that the maximum degree of certainty is mitigated by the parameter β, ensuring that we will never blindly trust one single training sample. Moreover, for tools with poor discrimination capabilities the pooled BBA (as defined in

equation (3.20)) will likely show distributed masses among the propositions $\{n\alpha\}$ and $\{t\alpha\}$, thus accounting for the lower reliability of the tool. That said, by getting to the bottom of tool reliabilities we may be able to adapt the interpretation of outputs based on the properties of the analyzed content, and this is the object of the next section.

3.3.2 Introducing background information

Now that we have a theoretical model for interpreting tool outputs by the light of training information, we can turn the attention to choosing which information should be used for training. The goal is to understand whether there is some background information that can help interpreting the output of a tool before moving to the decision fusion stage. Let us start with a general consideration: a common feature of all forensic tools is that when a footprint becomes "less detectable", algorithms relying on that footprint become less *reliable*, meaning that they do not discriminate well between presence and absence of the trace. Therefore, if we know which are the measurable properties that mostly affect the performance of a detector, we could use this information to adapt the interpretation of the output, decreasing the certainty when the tool is being used under unfavorable conditions and viceversa. Giving a formal definition of the detectability of a generic footprint is beyond the scope of this work; besides, the detectability of different footprints is affected by different parameters, and a golden rule seems hard to derive. These considerations suggest that the reliability of a given tool can be better investigated by using a sound experimental approach, that is, by conveniently testing the tool. To this end, we propose a possible procedure that the analyst may use to validate the reliability of the various tools as a function of a set of measurable parameters, so to establish if they actually impact the performance of the tools.

Identification of relevant properties

Suppose we have a set $\mathcal{F}$ of forensic tools whose goal is to tell if a given image contains a specific trace of forgery (we denote this hypothesis with $\mathcal{H}_1$) or not ($\mathcal{H}_0$). For simplicity, we assume that each tool $f \in \mathcal{F}$ outputs a score $s_f(x)$ (that may be, for example, the probability of the presence of the tampering

trace the tool is looking for), and decides for $\mathcal{H}_0$ when $s_f(x) \leq \tau$. In this way, the tool partitions the space of possible images $\mathcal{X}$ in two regions: Λ_0, containing the images for which $\mathcal{H}_0$ is accepted, and Λ_1, defined similarly for $\mathcal{H}_1$. According to classical detection theory, the probability of correct detection and false alarm for the specific tool and a given τ are defined, respectively, as:

$$P_D^f = \int_{\Lambda_1(\tau)} p(x|\mathcal{H}_1)\ \mathrm{d}x \quad \text{and} \quad P_{FA}^f = \int_{\Lambda_1(\tau)} p(x|\mathcal{H}_0)\ \mathrm{d}x,$$

where $p(x|\mathcal{H}_0)$ is the probability conditioned to the hypothesis that the image does not contain the trace and $p(x|\mathcal{H}_1)$ denotes the opposite case.

Now, let us assume that the analyst has access to a vector of independent measurable properties $p \in \mathcal{P}$, where $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2 \times \cdots \times \mathcal{P}_P$. We are interested in relating the performance of each tool to subsets of $\mathcal{P}$; for simplicity, we restrict one property at a time to a subrange of its possible values $\mathcal{R} \subset \mathcal{P}_j$. To do that, we define

$$\mathcal{R}_j = \mathcal{P}_1 \times \ldots \mathcal{P}_{j-1} \times \{\mathcal{P}_j \cap \mathcal{R}\} \times \ldots \mathcal{P}_P. \tag{3.21}$$

In practice, $\mathcal{R}_j$ denotes the set of images whose j-th property takes value in $\mathcal{R}$. Notice that the assumption of independent properties is made so to simplify the discussion; the framework can be adapted to account for the presence of dependent properties by redefining the set $\mathcal{P}$.

Using the above notation, we can write the probability of detection and the probability of false alarm of f when the analysis is restricted to a specific set of images (those for which the parameter j belongs to $\mathcal{R}$):

$$P_D^f(\mathcal{R}_j) = \int_{\Lambda_1(\tau)\cap\mathcal{R}_j} p(x|\mathcal{H}_1)\ \mathrm{d}x, \tag{3.22}$$

$$P_{FA}^f(\mathcal{R}_j) = \int_{\Lambda_1(\tau)\cap\mathcal{R}_j} p(x|\mathcal{H}_0)\ \mathrm{d}x. \tag{3.23}$$

Equations (3.22) and (3.23) give the probabilities for a given threshold τ. By varying τ, a ROC curve is generated, that is commonly used to evaluate the discrimination capability of a detector. By taking the integral of the ROC, the Area Under Curve (AUC) is obtained and, finally, the Gini coefficient

[39], denoted with ρ, can be used to summarize the performance of the tool:

$$\rho = 2 \times \mathrm{AUC} - 1. \tag{3.24}$$

By varying $\mathcal{R}_j$ in (3.21), the forensic analyst can investigate whether the performance of a tool change significantly when different subsets of $\mathcal{X}$ are considered.

3.3.3 Exploiting background information

Once the set of influencing properties has been determined, the problem is how to exploit them for improving the output interpretation. Interestingly, the system proposed in Section 3.3.1 can be adapted straightforwardly to account for background information. The idea is to expand the dimensionality of the "feature space", treating each influencing property as part of the problem (see Figure 3.6 for a graphical interpretation).

Formally, we modify equation (3.15) as

$$\mu_i : \mathcal{O}_i \times \mathcal{P}_1 \times \cdots \times \mathcal{P}_{N_i} \to \mathcal{M}^{\Theta_\alpha}, \tag{3.25}$$

where N_i denotes the number of influencing properties for the i-th tool. In other words, when the output of the tool for an image must be interpreted, the BBA is calculated not only from the output itself, but also considering the value assumed by the influencing properties on the specific image. This background information can be introduced easily by re-defining the training dataset as follows:

$$\mathcal{T} = \{t^i = (o^i, p^i_1, \ldots, p^i_P) : i = 1 \ldots N\} \tag{3.26}$$

where, for the i-th sample, o^i still denotes the output obtained from the tool and p^i_j denotes the value assumed by the j-th background property, properly scaled and normalized (see the Appendix for details). Similarly, also the query vector is updated with background information: $u = (o^u, p^u_1, \ldots, p^u_P)$. No other modification is needed: given a query vector, the nearest neighbors are searched, they provide a BBA as those in equations (3.18) and (3.19), and these BBAs are merged according to equation (3.20) to yield $m^u(X)$.

As desired, the pooled BBA is strongly influenced by background parameters: as one or more parameters move towards unfavorable values, samples

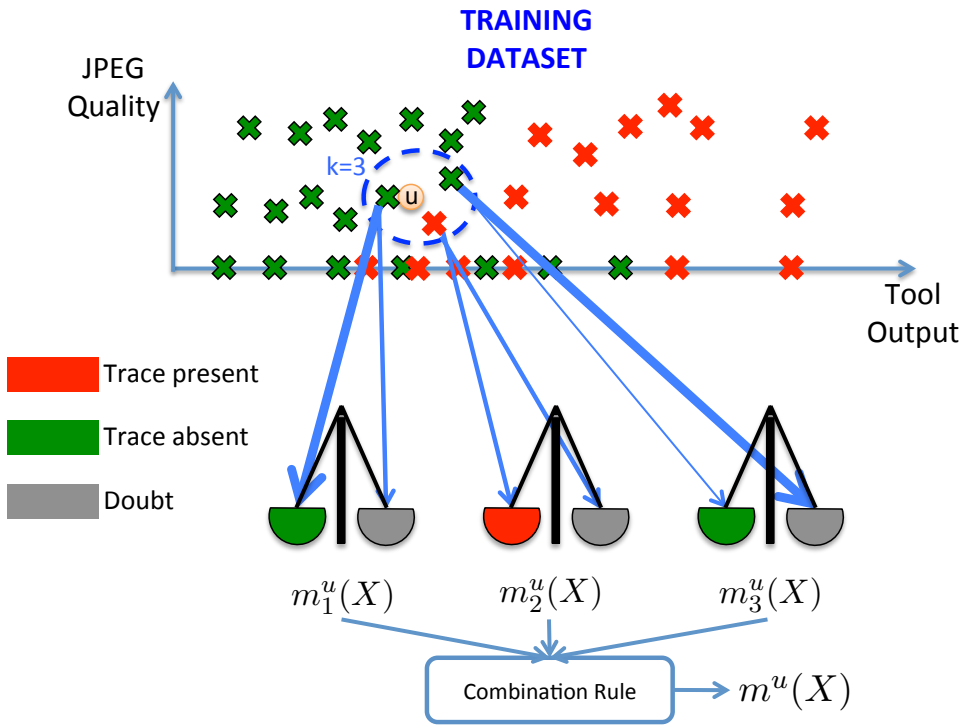


Figure 3.6: Graphic representation of the proposed way to include background information in tool output interpretation. In this example, only one image property is considered (the JPEG quality factor) for clarity.

in the dataset are likely to mix between the two classes, resulting in a less informative pooled BBA. Moreover, the final BBA will be increasingly doubtful as the unseen sample moves in unpopulated parts of the space, where few training samples are available: this perfectly models the fact that the analyst does not know how much the tool can be trusted in such working conditions.

After obtaining $m^u(X)$ for each of the tools available to the analyst, the decision fusion framework can be used to fuse them together and yield a global belief about the authenticity of the image.

Chapter 4

Experimental Validation and Concluding Remarks

THIS chapter investigates the effectiveness of the decision fusion framework presented in Chapter 3. We compare it with two other possible options for decision fusion at the measurement level, using two different datasets; we also investigate the impact of the inclusion of background information in the fusion scheme. The chapter is structured as follows: first, we define the case study we focus on, explaining the set of forensic traces we used to create an instance of the proposed framework, together with the set of tools that can detect those traces; as a second step, we deal with the interpretation of tools outputs and their mapping to BBAs, showing which are the selected image properties and motivating their choice. Having described the framework setup, we describe dataset generation together with the training and testing procedures. Finally, we compare the performance of the methods and comment them.

4.1 State of the art methods

Before diving into a detailed description, we describe the methods we compared our framework with. The first is the simple yet widely used logical disjunction (also known as "OR rule"): the image is classified as tampered if at least one tool detects its trace. Such a method was firstly proposed in forensics by Bayram et al. [26]. Logical disjunction is indeed one of the simplest and most widely used methods for decision fusion, and is quite well-suited to the proposed case study[1].

[1] Actually, this approach lays somewhere in the middle between the "*abstract*" and "measurement" level, since we take the logical sum of banalized outputs, but we also properly choose how to binarize them, without blindly relying on tools mechanism. Anyway, logical

As we mentioned in the Introduction, several methods have been proposed for decision fusion at the feature level in image forensics [23] [24] [25] [27], but they are typically based on feature selection and are therefore not directly comparable to the method proposed in this work. On the other hand, since most methods end up using a classifier (usually an SVM) the best we can do to compare our framework with them without exiting the measurement level is to train a SVM by using the scalar output of the tools as input features, and see how the SVM performs in discriminating between tampered and original images. By the way, to the best of our knowledge, this rather simple idea was never used in forensics before.

Finally, limiting to the proposed framework and to the SVM-based method, we compare performance obtained with and without using background information, so to investigate the benefits brought by using this kind of clue.

4.2 Reference case study and datasets

As already stated, we evaluated the validity of the new DST fusion framework by focusing on the detection of splicing attacks: a portion of an image (source) is cut and pasted into another image (host), thus producing a new content that is finally saved. Because most images are stored in JPEG format, a great deal of research has been carried out for the identification and detection of traces left by splicing attacks in JPEG images, so that several tools are available to search for them. In our experiments, we fused the outputs provided by five of these tools, searching for a total of three different traces.

4.2.1 Traces and tools

To explore all the features of the proposed scheme, we chose a set of algorithms such that some of them search for the same trace, and for which some combinations of traces are not possible. Namely, we are considering the following traces (see Figure 4.1 for a graphical explanation):

1. *Misaligned JPEG compression* (JPNA): this trace shows up when the investigated region is cropped from a JPEG image and pasted into the

disjunction is one of the most used approaches among the post-classification ones [22], so we decided to compare our method against it.

target picture without preserving JPEG grid alignment, performing a final JPEG compression. Therefore, pasted pixels undergo two misaligned compressions, while others do not.

2. *Double quantization* (JPDQ): when a portion of uncompressed pixels[2] is pasted into a JPEG image, and the final result is JPEG saved, the untouched region undergoes a double compression. This causes its DCT coefficients to be doubly quantized, leaving a characteristic trace in their statistics.

3. *JPEG ghost* (JPGH): this trace appears when a region is cut-and-pasted, respecting grid alignment, from a JPEG source image into the host one (which has not been JPEG compressed). When the obtained splicing is JPEG saved, the inserted part undergoes a second compression, while the outer part is compressed for the first time, thus introducing an inconsistency.

Given the above definitions, some combinations of traces are not possible. For example, an attack that introduces the JPDQ trace also introduces JPGH, while the contrary is not necessarily true; but, if both JPGH and JPNA are introduced, then also JPDQ must be present. These facts are best represented by using a tabular form (Section 3.2.4) with the compatibility relations, as in Tab. 4.1.

[2] or pixels that have been compressed according to a different grid.

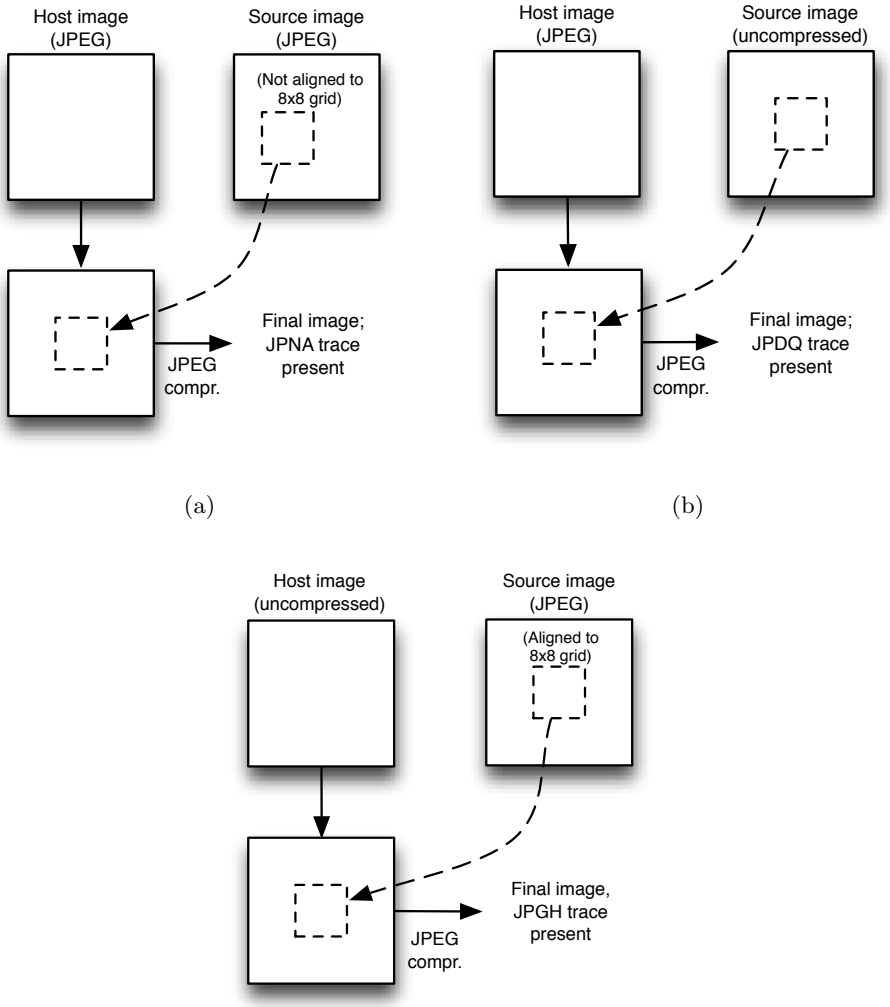


Figure 4.1: In these schemes three different configurations of cut&paste attacks are reported. The attack in (a) introduces a misaligned double compression, the one in (b) introduces the double quantization effect in the untouched part of the final image and the attack in (c) introduces the ghost effect in the pasted region.

Trace	**Possible**					**Excluded**		
JPNA	Y	N	N	Y	N	Y	Y	N
JPDQ	N	Y	N	Y	N	Y	N	Y
JPGH	N	Y	Y	Y	N	N	Y	N

Table 4.1: Detection compatibility: each column of the table forms a combination of presence (Y) and absence (N) of traces. We see that only 5 out of 8 combinations are possible.

Now that we have introduced the traces considered in our experiments, we list the adopted forensic tools (see Tab. 4.2). We employed two tools looking for JPNA, namely the one by Luo et al. [40] (*ToolA*) and the one by Bianchi et al. [41] (*ToolD*); two tools looking for JPDQ, the one by Lin et al.[4] (*ToolB*) and the one by Bianchi et al. [35] (*ToolE*); and the tool by Farid that searches for ghost traces [42] (*ToolC*).

Trace	**Tools**
JPNA	*ToolA* [40], *ToolD* [41]
JPDQ	*ToolB* [4], *ToolE* [35]
JPGH	*ToolC* [42]

Table 4.2: Coupling between traces and tools: for each trace, the list of adopted tools able to detect it is given.

As mentioned at the beginning of this section, usually the simple presence of a trace does not imply a splicing attack, but only that a common processing over the image occurred (for example, cropping a couple of rows from the top of the image would introduce a JPNA trace). Instead, *inconsistencies* in the presence of a trace through the image (i.e., high detection values for the suspect region and low for the other or vice-versa) are far more suspect. For this reason, each tool[3] is run both on the suspect region and on the remaining part of the image, and the absolute difference between the two is considered as the final output of each tool.

[3] *ToolC* is excluded since it already considers inconsistencies over the image.

4.2.2 Normalization of outputs

In order to pass from outputs to BBAs, it is important to recall that formulas (3.18) and (3.19) weigh the contributions of neighboring samples in the dataset based on their distance from the observed point. Since tool outputs and reliability properties are very different in nature and can assume different ranges of values, it is important either to select a sufficiently refined distance function or to normalize them properly. We opted for the second option: in the following, we first give a brief description of how each of the selected tools works, and then define the approach we adopted to obtain a scalar output from it. We will denote by $\hat{x}_W$ the output of tool W, and by x_W its normalized version:

- *ToolA* searches for misaligned compression by measuring inconsistencies in blocking artifacts in the spatial domain. Because features are classified by using an SVM (which we trained on a separated dataset, according to the original work) we train a model supporting probability estimates [43]. The resulting outputs are well spread in the interval [0,1] and need no further processing;

- *ToolB* and *ToolE* search for double quantization traces by employing two different statistical models to analyze the histogram of the DCT coefficients of the image. Both tools provide a probability map which gives, for each 8×8 pixel block, the probability of being original (i.e.,showing double quantization) or tampered (not showing double quantization). The final detection value is taken as the median (over the suspect region only) of the probability map. Being likelihood ratios, the outputs from these tools are very concentrated around 0 and 1, making their use problematic. We normalized the outputs using the following formulas: $\hat{x}_B = (\log_{10}(x_B)/15)+1$, and $\hat{x}_E = \log_{10}(x_E)/6+1$ for *ToolB* and *ToolE* respectively.

- *ToolC* searches for JPEG ghost artifacts by re-compressing the image at several different qualities and taking the difference between the given image and the re-compressed one. As such, this is a tool working in the spatial domain, like *ToolA*. Ghost effect is detected when the difference is small for the suspect region and not for the rest of the image. To evaluate

how much the two regions are separated, we used the KS statistic [42]. The value of this statistic can be directly used in the mapping phase without normalization;

- *ToolD* searches for misaligned double compression exploiting the fact that DCT coefficients exhibit an integer periodicity when the DCT is computed according to the grid of the primary compression. Being the shift of the grid unknown, the algorithm searches among all possible shifts the one that minimizes a specific metric (see [41] for details). We scale and invert this metric from [0,6] to [0,1] and normalize it as follows:

$$\hat{x}_D = \frac{\log_2(x_D)}{20\log_2(1.5)} + 1.$$

4.2.3 The synthetic forgery dataset

In order to generate a sufficiently large dataset, we collected a total of 630 uncompressed images representing a variety of scenes (indoor, outdoor, people, landscapes, etc.), all cropped to size 1536×1536 pixels. We considered as possible values for the size of the tampering: $64\times64, 128\times128, 256\times256, 512\times512$, and 1024×1024 pixels. Each tampering was created by pasting, in the center of the image, a region cut from another version of the *same* image. This tampering strategy creates forgeries that are virtually undetectable to the eye (see Figure 4.2 for some examples), and also mimics the work of an image editing expert, which would limit discontinuities along the boundary of the tampered region. By varying the way the splicing is produced, see Table 4.3, we generated splicings containing all the possible combinations of traces listed previously.

For tampered images, we let the quality of the first JPEG compression (Q_1) take values in the set $\{60, 65, \ldots, 100\}$, and the quality of the final compression is chosen as $Q_2 = \min\{Q_1 + \delta, 100\}$, where δ is chosen at random from the set $\{5, 10, 15, 20\}$.[4] Untouched images are compressed only once with $Q = \{65, 70, \ldots, 100\}$. By combining the above settings, from each uncompressed image the following files have been created:

[4]We do not investigate the case where $Q2 < Q1$ because it is a setting which most image forensic cannot deal with.

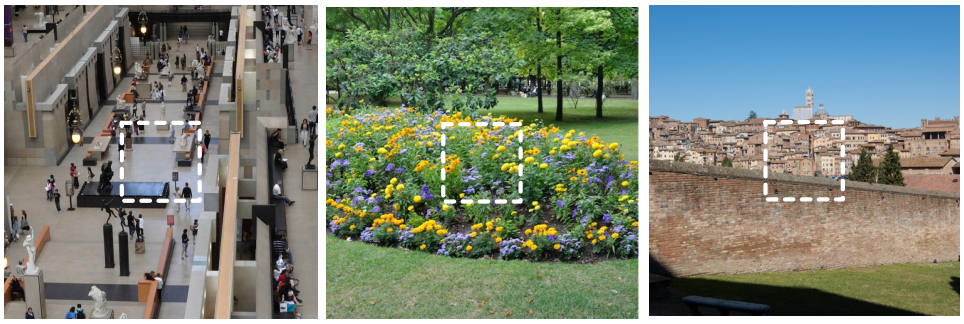


Figure 4.2: Some sample forgeries from the synthetic dataset: the spliced region, highlighted by the dashed square, has been taken from another version of the same image, thus creating an imperceptible forgery.

- 40 non-tampered JPEG images, by using all possible values for QF_1, and taking all possible sizes for the suspect (although not tampered) region;

- 40 forged images, by using all of the 5 possible sizes of the tampering and two random coupling for Q_1 and Q_2, thus obtaining 10 images forged according to each different procedure.

The dataset therefore consists of a total of 50400 JPEG images, half of them tampered. Each different class of splicing consists of $25200/4 = 6300$ sample images. During the creation of the dataset, we annotated both the average value and the standard deviation of pixels in the suspect region (in the case of a color image, the image is converted to the YCbCr space and the Y channel is considered). The resulting dataset is available for downloading[5], together with the output obtained from the 5 considered tools.

4.2.4 The realistic forgery dataset

We also studied a more realistic scenario: a team of students created 70 forgeries (some examples are given in Figure 4.3) using common photo editing

[5] `http://clem.dii.unisi.it/~vipp/index.php/download/imagerepository`

Class	**Procedure**	**Result**
Class 1	Region is cut from a JPEG image and pasted, breaking the 8x8 grid, into an uncompressed one; the result is saved as JPEG.	Inner region shows JPNA trace, external region does not. *Only tool A detects this trace.*
Class 2	Region is taken from an uncompressed image and pasted into a JPEG one; the result is saved as JPEG.	Outer region shows both JPDQ and JPGH traces, inner does not. *Tools B, E and C detect this trace*
Class 3	Region is cut from a JPEG image and pasted into an uncompressed one in a position multiple of the 8x8 grid; result is saved as JPEG.	The inner region shows JPGH effect, the outer does not. *Only Tool C detects.*
Class 4	Region is cut from a JPEG image and pasted (without respecting the original 8x8 grid) into a JPEG image; the result is saved as JPEG	The inner region shows JPNA, the outer shows JPDQ and JPGH. *All tools detect this trace.*

Table 4.3: Procedure for the creation of different classes of tampering in the training dataset.

software, respecting only a constraint about JPEG quality factors (the quality factor of the final compression is always higher than the one of the host image). Students were asked to provide both tampered images (along with ground truth masks) and original ones, for a total of 136 images. Although being rather small (creating good forgeries is a time consuming procedure) this dataset is crucial to understand how well the considered frameworks generalize to unseen cases. We will refer to this dataset as the "realistic" dataset. According to a realistic scenario, this dataset is used only for testing, and training will always be performed on images of the synthetic dataset.

Figure 4.3: Some sample forgeries from the realistic dataset: in the leftmost image the license plate has been pasted, while faces of celebrities have been substituted in the other two pictures.

4.2.5 Choice of reliability properties

Let us now apply the BBA mapping approach proposed in Section 3.3.2 to the above case study. We define a product set of four possibly relevant properties

$$\mathcal{P} = \mathrm{Q} \times \mathrm{Z} \times \mathrm{A} \times \mathrm{S},$$

defined as follows:

- Q - *compression strength:* lossy coding after the manipulation process discards some information, thus concealing the already vanishing footprints left by the processing. Stronger compressions are against the analyst, because they erase the footprints more deeply.
- Z - *size of the analyzed region:* most forensic tools rely either on a statistical model or on the extraction and classification of some features. In both cases, working with more data results in a more reliable analysis.
- A - *average value of pixels in the analyzed region:* many forensic tools do not work well in saturated regions (i.e., having very low or very high luminance values). This holds especially for DCT-based algorithms, where the truncation errors due to saturation introduce anomalies in DCT coefficients.
- S - *standard deviation of pixels in the analyzed region:* uniform (i.e., having very low standard deviation) content yields an extremely sparse DCT representation, that can hardly lead to a reliable forensic analysis.

We used the synthetic forgeries dataset to investigate the dependency of the performance of tools on the above properties. Figure 4.4 shows the ROC curves obtained by each tool in $\mathcal{F}$ for different ranges of the property Q, along with the value of ρ calculated for each curve: we can definitely state that this property strongly influences the performance of tools in $\mathcal{F}$ and, noticeably, some tools are more sensitive than others (compare, for example, the variation of the ρ value for JPGH and JPDQ).

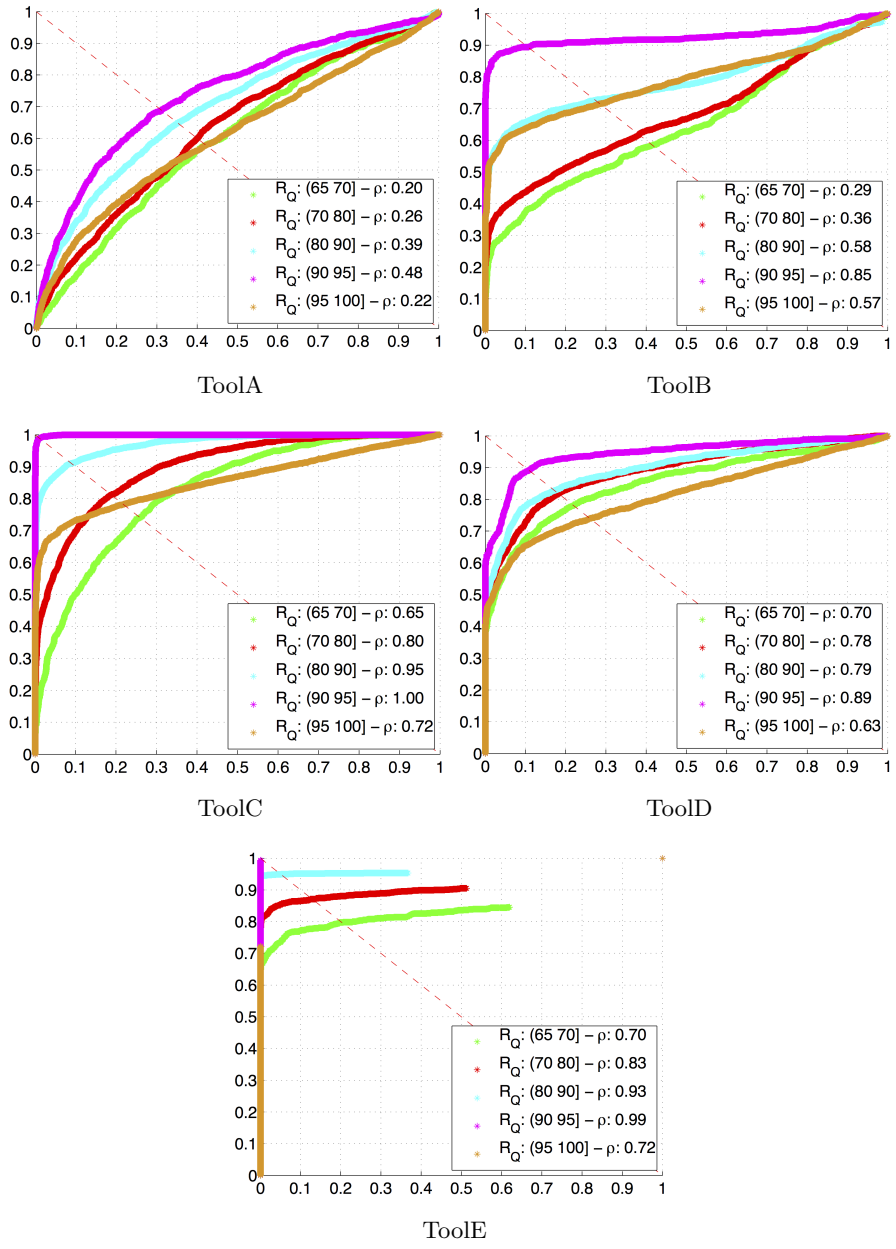


Figure 4.4: ROC curves for tools in $\mathcal{F}$ for different ranges of last JPEG compression quality factor: $\mathcal{R}_{\mathrm{Q}}(a, b]$ denotes the set of all images in the dataset whose last compression quality factor falls within $(a, b]$. In each plot, the probability of detection P_D^f is plotted against the probability of false alarm P_{FA}^f.

Instead of plotting similar figures for each of the investigated properties, we summarize with Table 4.4 the analysis for other elements of $\mathcal{P}$. We see that all the properties affect the performance of the tools and, most noticeably, not all the tools are affected in the same way. Consider, for instance, the size of the analyzed region (parameter Z): it strongly affects the performance of ToolC and ToolA but does not influence significantly ToolD. When performing a joint analysis, such an information can greatly help the analyst in reaching a correct global decision.

Since reliability parameters are very different in nature, it is necessary to normalize their values before using them. We used the following order-preserving functions to normalize them in the interval [0,1] (W denotes the normalized version of $\hat{\text{W}}$):

- Size of the suspect region: denote with X and Y the height and width of the image, then:
$$\text{S} = \frac{\log_2(\sqrt{X * Y}) - 3}{6}.$$
- Compression Quality Factor: $\text{QF} = \hat{\text{QF}}/100$.
- Average pixel value: $\text{AVG} = \hat{\text{AVG}}/255$.
- Standard deviation (STD): for natural images, the standard deviation will unlikely assume values higher than 100. Therefore, the scaled parameter is obtained as: $\text{STD} = \hat{\text{STD}}/100$.

4.3 Training procedure

For all the fusion techniques used in the tests we need to run a training phase. For creating train and test datasets, we divided the synthetic forgery dataset in two parts, with 80% of the images used for training and 20% for testing. The whole procedure is repeated 10 times to increase the statistical significance of the experiment. It is worth noting that, in the proposed framework, training affects only the BBA mapping phase, so it is performed separately for each tool. On the contrary, a SVM cannot be trained separately for each tool: it must "see", for each training image, the joint outputs coming from all

Tool	$\mathbf{R}^1_Z$: (0,64]	$\mathbf{R}^2_Z$: (64,128]	$\mathbf{R}^3_Z$: (128,256]	$\mathbf{R}^4_Z$: (256,512]	$\mathbf{R}^5_Z$: (512,1024]
ToolA	0	0.08	0.21	0.31	0.40
ToolB	040	0.39	0.36	0.31	0.21
ToolC	0.63	0.67	0.71	0.75	0.80
ToolD	0.74	0.75	0.74	0.73	0.72
ToolE	0.37	0.62	0.72	0.75	0.78

	$\mathbf{R}^1_A$: (0,30]	$\mathbf{R}^2_A$: (30,60]	$\mathbf{R}^3_A$: (60,150]	$\mathbf{R}^4_A$: (150,230]	$\mathbf{R}^5_A$: (230,255]
ToolA	0.15	0.19	0.23	0.14	-0.23
ToolB	0.09	0.35	0.38	0.25	0.19
ToolC	0.49	0.68	0.73	0.62	0.20
ToolD	0.58	0.78	0.80	0.60	0.36
ToolE	0.50	0.63	0.70	0.54	0.04

	$\mathbf{R}^1_S$: (0,5]	$\mathbf{R}^2_S$: (5,10]	$\mathbf{R}^3_S$: (10,15]	$\mathbf{R}^4_S$: (20,40]	$\mathbf{R}^5_S$: (40,60]
ToolA	0.07	0.13	0.18	0.21	0.30
ToolB	0.28	0.28	0.34	0.38	0.33
ToolC	0.51	0.69	0.70	0.73	0.74
ToolD	0.46	0.65	0.76	0.79	0.80
ToolE	0.31	0.60	0.65	0.71	0.73

Table 4.4: Impact of parameters Z, A and S on the performance of five image forensic tools. Intervals are chosen so to emphasize extreme values for each parameter.

tools, so to learn how to fuse them. Accordingly, training the SVM by providing it forged images containing all possible combination of traces would not be realistic, since it would require a dataset whose size grows exponentially with the number of traces. We find it more reasonable to limit the training dataset to original images and images containing all the forensic traces, i.e., images belonging to "Class 4" (see table 4.3). Of course, this restriction is applied to all the tested techniques. We also point out that tool outputs and

values of reliability properties have been normalized in the same way (Section 4.2.1) before being used with all the methods. In the following, more specific information about the training procedures for each method are given.

- *SVM fusion.* We used a radial basis function (RBF) kernel, whose parameters C and γ are selected through a grid search. The search was repeated independently two times, one for the SVM that is trained with both tool outputs and background information ($C = 4, \gamma = 4$), and one for the SVM trained only with tool outputs ($C = 256, \gamma = 0.5$).

- *DST fusion.* The output interpretation procedure presented in Section 3.3.2 was used for mapping tool outputs to BBAs. As for the SVM, the experiment was repeated twice, once including background information and once not. As to the parameters for the BBA mapping, we ran a grid search and chose, for both the experiments, $\beta = 0.8$, $\gamma = 8$ and $k = 12$.

- *OR-based fusion.* Since ROC curves are used to compare the various methods, we need to train an *aggregate* ROC for the five algorithms, which represents their behavior in terms of probability of detection (p_D) and false alarm (p_{FA}) after being combined with the OR operator. To obtain these curves, we uniformly sampled (with precision 10^{-3}) the ROC of each algorithm, considering only images that satisfy the corresponding working assumptions, as reported in Table 4.3. For each algorithm we saved the threshold associated with each p_{FA}. During the test phase, given a target overall probability of false alarm $\hat{p}_{FA}$, we chose for each algorithm the threshold corresponding to a probability of false alarm of $\hat{p}_{FA}/5$, and we used that threshold to binarize the output. Binarized outputs for each image were then combined with the OR operator, giving the final classification, that allows drawing a point of the overall ROC.

Concerning the realistic dataset, as we said, this dataset is used only for testing, while training is performed using synthetic images. When experiments are carried on the realistic dataset, 100% of the synthetic dataset is used to generate the training set, still according to the rules described above.

4.4 Results and discussion

The five forensic tools were run on the datasets, gathering the selected reliability properties from the images, and their outputs were combined by the different fusion methods. We use ROC curves to compare tool performance, also calculating the Gini coefficient to allow a more compact evaluation. Each ROC curve is obtained by averaging the results obtained on the 10 train-test selections; we also plot uncertainty bars showing the maximum and minimum probability of detection retained for different false alarm probabilities. For sake of clarity, we separately comment the results obtained for the synthetic dataset and those obtained for the realistic dataset.

Results on the synthetic dataset First of all, we show in Figure 4.5 the ROC curve obtained by executing each stand-alone forensic algorithm on the whole dataset. The reader will probably be surprised by the poor performance obtained by single algorithms, but they are perfectly reasonable since each algorithm is used to analyze all classes of images, not only those that are detectable with it. This approach is close to reality: a real analyst does not know in advance which kind of tampering could have been performed on the image under analysis. On the contrary, when a forensic algorithm is developed and evaluated in scientific literature, it is typically tested with images that are either original or tampered in a "detectable way". Although being useful to evaluate the discriminative power of a specific footprint, this approach may lead to a rather optimistic evaluation of tool performance.

As to the comparison between decision fusion frameworks, Figure 4.6 shows the results obtained with the three considered methods. We can state that the proposed method retains slightly better performance compared to the SVM: this is an encouraging result, especially if we consider that both training and test forgeries are synthetically generated in this dataset, and that we have a high ratio of training examples versus features (about 10,000 training examples in front of 9-dimensional, normalized features). Interestingly, logical disjunction also shows good performance on this dataset. The most evident conclusion we can draw from this experiment is that all fusion methods guarantee a sensible performance gain compared to single tools, thus confirming that decision fusion helps moving towards a more comprehensive

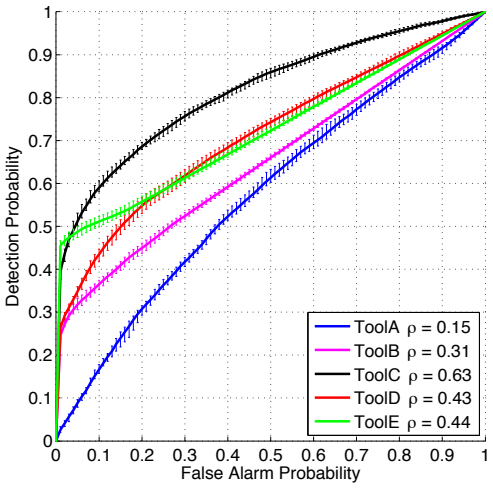


Figure 4.5: Performance obtained by each forensic tool alone on the synthetic test dataset. Although tools do not require a training phase, uncertainty bars are reported because their evaluation is repeated on different selections of the test dataset.

forensic analysis.

Let us now focus on the contribution brought by background information to the DST- and SVM- based methods. Figure 4.7 shows the performance of the proposed framework and those of the SVM system with and without the use of such an information. We can see that by including background information the analyst yields a clear advantage, regardless of the chosen fusion framework. This gain gets even more interesting if we consider that including background information has a negligible cost in terms of complexity, at least up to a small number of properties. On the other hand, the analyst should beware of selecting a vast set of influencing parameters, since this can potentially expose the framework to the "curse of dimensionality".

Results on the realistic dataset Focusing on the realistic dataset, Figure 4.8 shows single tools performance. We can see a variation in performance compared to curves in Figure 4.5, a fact that is not surprising because of the different nature of hand-made forgeries: small size and irregular shape of

the tampered area, post-processing following the cut-&-paste operation, and possibly other factors affect the forensic analysis.

The most interesting results are those obtained by the decision fusion methods on the realistic dataset. As Figure 4.9 shows, the proposed method strongly outperforms both the OR- and SVM- based approaches on such a dataset. The most evident fact is that the SVM-based method seems to suffer significantly the deep mismatch between the characteristics of the training and testing datasets. In fact, we believe that such a mismatch is unavoidable in practical situations, because it would be very hard to create a huge hand-made realistic dataset, resembling all possible kinds of operations the forger could do. It is much more reasonable, in our opinion, to define formally and unambiguously an automatic method to generate the training set, and then use the trained fusion system on realistic data. This is actually what is done by the proposed DST-based framework.

We still have to evaluate the impact of background information on the performance obtained on the realistic dataset: as shown in Figure 4.10, also from this point of view, the DST-based method is preferable, since it successfully exploits the presence of background information. On the other hand,

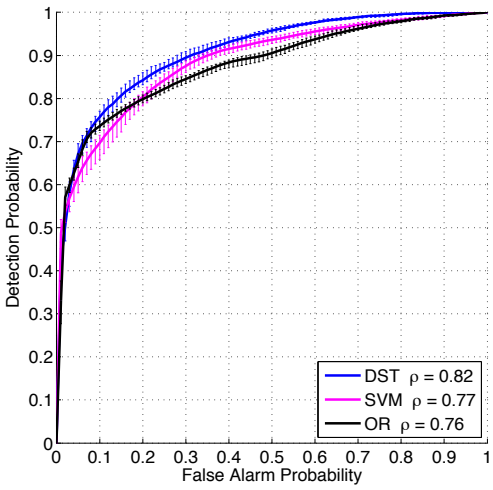


Figure 4.6: Performance obtained on the synthetic dataset by the proposed fusion framework and by the other methods.

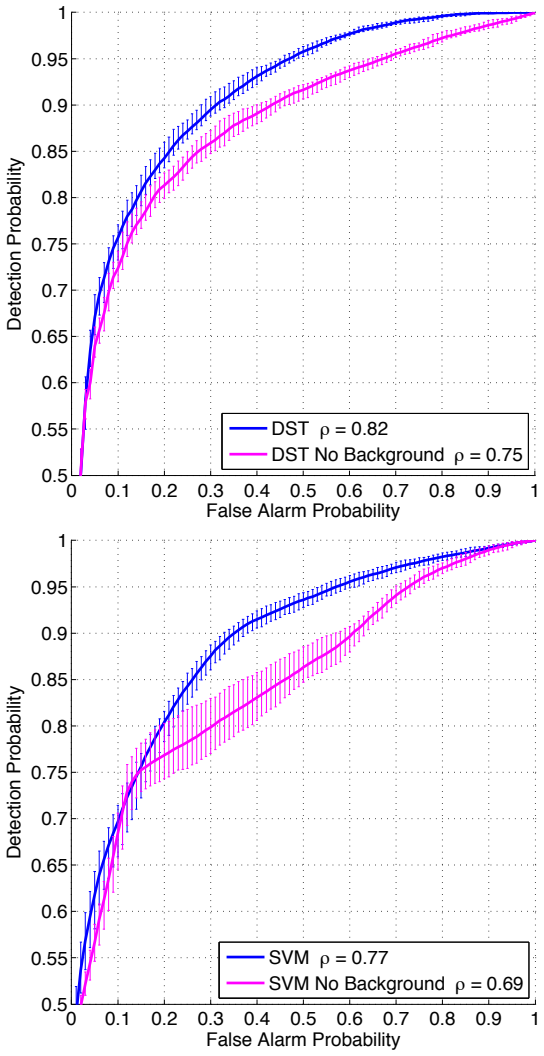


Figure 4.7: Comparison between performance of the background information aware fusion methods and their simpler version, that does not use such an information; results refer to the synthetic dataset.

the SVM method seems to be penalized by background information for low false-alarm rates, that are by far the most important in a forensic scenario.

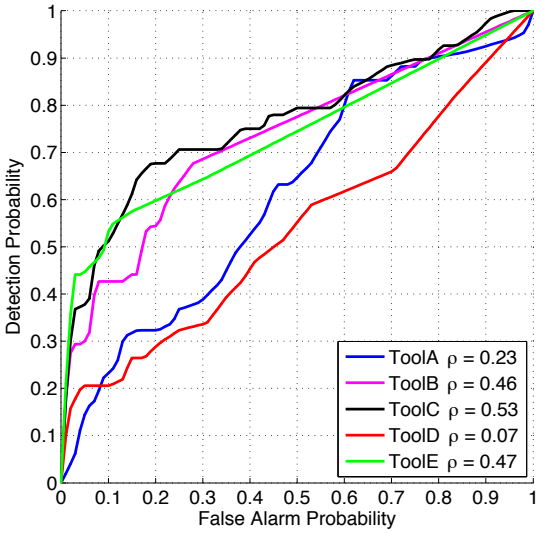


Figure 4.8: Performance obtained by each forensic tool alone on the realistic test dataset. Differently from Figure 4.5, uncertainty bars are not present because the whole realistic dataset is always used as the test set.

It is probably useful to remark that there are no differences in the "feature vectors" provided to the DST and SVM methods, meaning that both tool outputs and values of reliability properties are normalized in the same way, as explained in Section 4.2.5. Therefore, we should rather refer again to the mismatch between training and testing examples to explain the difference in the impact of background information on the two frameworks.

4.4.1 Noticeable case studies

Besides presenting the results obtained on the synthetic and realistic datasets as a whole, we believe it is interesting to isolate some noticeable case studies where the DST and SVM methods provide significantly different results. In order to select such examples, we focused on those images of the realistic dataset for which, using the threshold that give for both schemes a false alarm probability of 10%, the DST method provides a correct classification while the SVM does not. As the significant distance between the two ROC

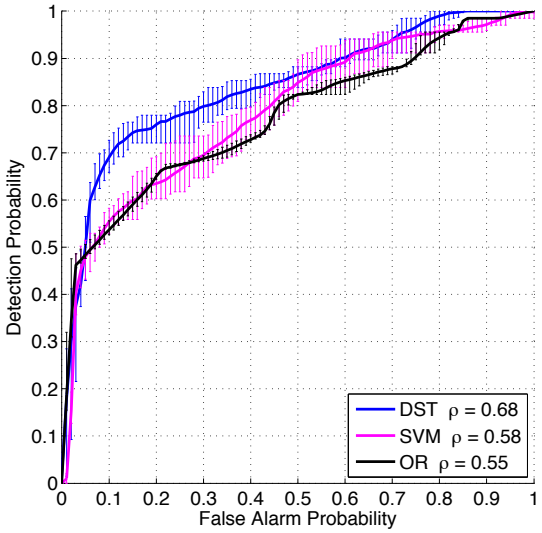


Figure 4.9: Performance obtained on the realistic dataset by the proposed fusion framework and by the other considered methods.

curves in Figure 4.9 suggests, at this "working point" there are several cases for which the DST method outperforms the SVM, specifically 16 out of 136 total images. Among these, we select and comment 5 of them in the following. For each case study, we report:

- the image and the suspect region on which algorithms have been run;
- the properties associated to the analyzed region that have been fed to the SVM and BBA-mapping modules;
- the output of forensic tools, and their interpreted version (available only for the DST method);
- the final, scalar output of the DST and SVM fusion frameworks.

Case study 1 (true negative) In the first case study (Figure 4.11), the DST method correctly classifies the image as original (the scalar output is 0.243), while the SVM labels it as tampered (scalar output: 0.652). Table 4.5

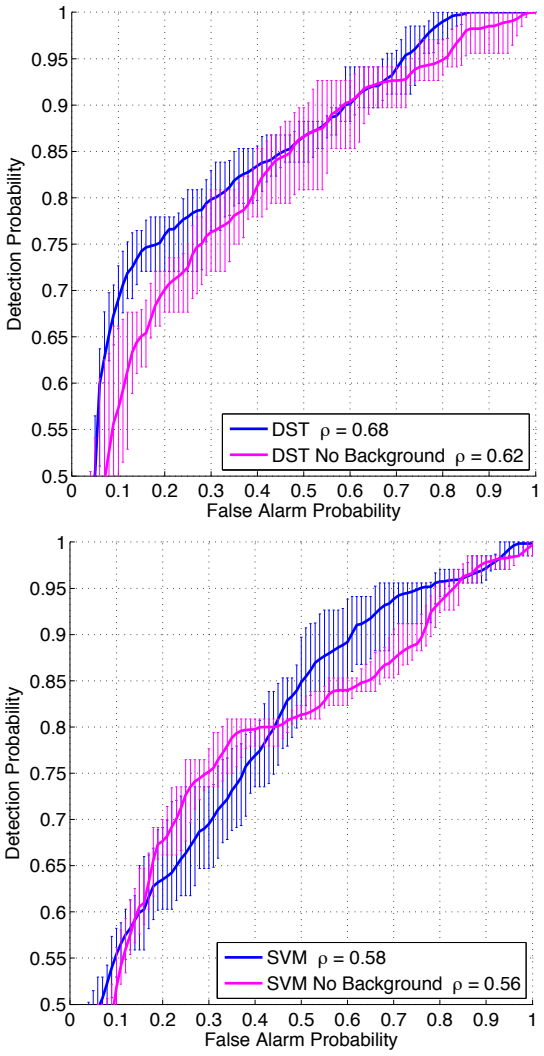


Figure 4.10: Comparison between performance of the background information aware fusion methods and their simpler version, that does not use such an information; results refer to the realistic dataset.

reports the values of relevant properties and the output from each forensic tool. We denote with $m(\mathrm{T})$, $m(\mathrm{N})$ and $m(\mathrm{D})$ the interpretation provided by

Figure 4.11: Image related to case study 1.

Compression	Size	Avg. value	St. Dev.
90	1413	74.45	38.50

	Tool A	Tool B	Tool C	Tool D	Tool E
Output	0.53	0.00	0.17	0.00	0.03
m(T)	1.00	0.01	0.00	0.00	0.12
m(N)	0.00	0.99	1.00	1.00	0.88
m(D)	0.00	0.00	0.00	0.00	0.00

Table 4.5: Background properties values, tool outputs and their mapping to BBAs related to case study 1.

the BBA-mapping module for propositions "the trace is present", "the trace is absent" and doubt respectively.

The most interesting fact of this case study is the strong conflict between the two tools searching for the JPNA trace (namely, Tool A and Tool D), highlighted in red in the table. This is already evident looking at the output (first line of the table), but it becomes even more dramatic if we consider the interpretation resulting from the BBA-mapping module. The direct consequence of such a conflicting situation is that belief stemming from these two

Compression	Size	Avg. value	St. Dev.
85	100	129.09	22.66

	Tool A	Tool B	Tool C	Tool D	Tool E
Output	0.23	0.00	0.43	0.03	0.08
m(T)	0.01	0.00	0.01	0.00	0.48
m(N)	0.99	1.00	0.99	1.00	0.52
m(D)	0.00	0.00	0.00	0.00	0.00

Table 4.6: Background properties values, tool outputs and their mapping to BBAs related to case study 2.

tools cancels, and is re-distributed based on the information available from other tools. Since none of the three remaining tools found any trace, the image is thus classified as original.

Case study 2 (true negative) Also in this case (Figure 4.12 and Table 4.6), the DST method correctly classifies the image as authentic (output value is 0.005), while the SVM detects tampering (0.713). The most relevant aspect of this case study resides in the BBA-mapping: notice that while the output of Tool C is 0.43, a value that is near to the decision threshold for that tool [42], the interpretation of such a value is definitely towards absence of the trace (values are highlighted in red). This is likely due to the fact that the resolution of the image is rather low, so that the suspect region consists of just a few hundreds pixels. For such small regions, it is not surprising to reach higher values of the KS statistic employed by Tool C also for untouched regions.

Case study 3 (true positive) Let us now consider a case (Figure 4.13 and Table 4.7) where the DST framework correctly detects tampering (output is 1.000) while the SVM wrongly labels the image as authentic (0.086). Once again, conflicting information plays a fundamental role: we see that tools searching for trace JPDQ (ToolB and ToolE) provide totally conflicting outputs. The conflicting belief is thus redistributed across plausible assignments

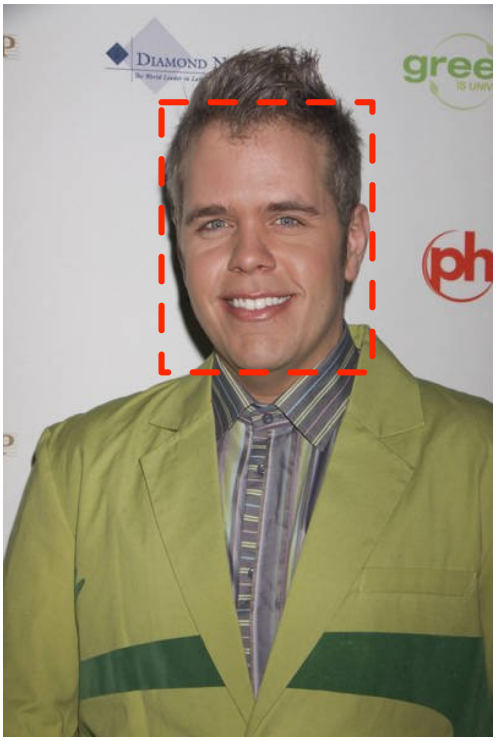


Figure 4.12: Image related to case study 2.

and, since the Tool C detected the JPGH trace, and the presence of only that trace is sufficient to declare the region tampered (according to Table 4.1), the final belief supports totally the tampering hypothesis.

Case study 4 (true positive) The last case study (Figure 4.14 and Table 4.8) highlights an important feature of the DST framework. As we can see from Table 4.8, in this case only Tool C detects traces of tampering, while all the other tools are highly confident that their trace is not present. As we explained in previous chapters, this fact should not lower the belief of the analyst about the presence of tampering: it may well be the case that only one trace was left during forgery creation. Therefore, as long as the combination with absence/presence of other traces is plausible, detecting that trace is sufficient to label the image as tampered. This is exactly the case at hand,

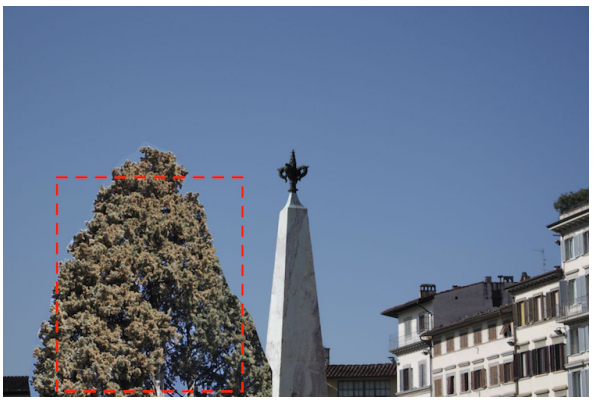


Figure 4.13: Image related to case study 3.

Compression	Size	Avg. value	St. Dev.
75	377	89.95	39.94

	Tool A	Tool B	Tool C	Tool D	Tool E
Output	0.01	0.99	0.65	0.07	0.00
m(T)	0.00	1.00	1.00	0.00	0.00
m(N)	1.00	0.00	0.00	1.00	1.00
m(D)	0.00	0.00	0.00	0.00	0.00

Table 4.7: Background properties values, tool outputs and their mapping to BBAs related to case study 3.

since the solely presence of trace JPGH is plausible (according to Table 4.1). The DST output for this image assigns 1.00 to the proposition "the image is tampered". On the other hand, the SVM probably let other tools output "smoothen" its decision, ending up with a fused score of 0.436.

4.4.2 Comments

Based on our experiments, we can finally state that the proposed method is preferable in realistic conditions because it is more "robust", meaning that differences between the training and testing datasets have a smaller impact on

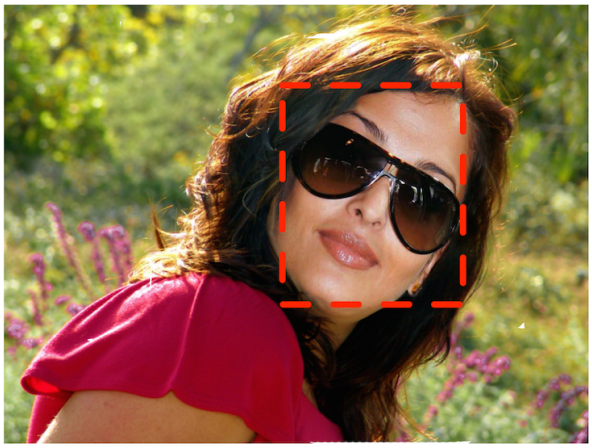


Figure 4.14: Image related to case study 4.

Compression	**Size**	**Avg. value**	**St. Dev.**
100	1100	97.47	57.08

	Tool A	**Tool B**	**Tool C**	**Tool D**	**Tool E**
Output	0.01	0.00	0.80	0.08	0.00
m(T)	0.00	0.02	1.00	0.00	0.00
m(N)	1.00	0.98	0.00	1.00	1.00
m(D)	0.00	0.00	0.00	0.00	0.00

Table 4.8: Background properties values, tool outputs and their mapping to BBAs related to case study 4.

performance. It should also be noted that, in contrast to SVM, the training phase of the DST method treats each tool separately. This fact has several advantages, most noticeably the possibility of adding new tools without re-training the whole system, and the possibility to select different sets of influencing parameters for each tool (although this was not necessary in the case we considered here). On the other hand, the weakest point of the proposed framework resides in the specification of compatibility relationships, since the number of combinations grows exponentially with the number of different traces, as discussed in Section 3.2.5. However, as we pointed out,

this fact holds only when the analyst wants to maintain a full granularity of the information. This means, besides evaluating the final belief as we did in the previous experiments, to be able to compute the belief for the presence of each single trace separately.

4.5 Concluding remarks

After several years of research image forensic is still a blooming discipline, with a constant increase of the number of available tools, investigating a wide range of manipulations. Since each tool analyzes a very specific trace of tampering, one tool alone cannot suffice for an image forensic analyst. Motivated by this fact, in this part of the thesis we have proposed a decision fusion framework based on DST, and we have shown that it provides a more robust and versatile instrument compared to the use of single tools alone. During the development of this framework, it became evident that interpreting the output of image forensic tools is not a trivial task, since their performance vary significantly depending on the analyzed content. For this reason, we proposed a simple method to assess which are the properties of the image that impact tool performance the most, and we developed a DST-based approach for mapping tool outputs to basic belief assignments, also taking into account background information. The final result is a comprehensive framework that allows introducing new tools in a rather simple way, and that keep the promise of increasing the reliability of the analysis, as the experimental results showed.

While we believe that the proposed framework provides a useful contribution to Image Forensics, we also think that it should be regarded as a starting point rather than a final work. To motivate such a claim, in this section we outline two possible developments of the framework, on which we are currently working.

4.5.1 Decision fusion for unsupervised forgery localization

One important fact that remained slightly hidden during the discussion of the DST-fusion framework is that it requires that the analyst selects the suspect region within the image. This step is crucial, since each tool is expected to

output a single scalar value, which is obtained by comparing in some way the selected region with the rest of the image. In some cases, this kind of user intervention is not practical, for several reasons: i) the user can hardly suspect about a region where something was hidden; ii) when a huge amount of images have to be analyzed, accurate inspection can be expensive; iii) the results produced by tools may vary even significantly when the same object is selected in different ways, and no golden rule exists in principle: an "abundant" selection may contain pixels from the background, while a "conservative" selection may result in small regions, leading to a less reliable statistical analysis. Based on these considerations, it would be desirable to develop *unsupervised* tools that allow forgery *localization*, e.g. by producing a probability map that associates to each (block of) pixel the probability of being tampered.

The above problem has been addressed in several ways in the image forensic literature: a first class of unsupervised forgery localization algorithms looks for the presence of tampered objects by decomposing the image under analysis into subparts. In region-wise approaches, the image is first segmented into homogeneous regions and then each region is analyzed separately [44]; in block-wise approaches, the image is split into sliding square windows, and each image block is processed independently. Inconsistencies in the presence or the absence of specific footprints related to acquisition, coding, or editing of one or more sub-parts of the image indirectly reveal that some processing has been applied on a particular region of the image [3, 45]. Concerning the limits of these methods, in the region-wise approach very often the segmentation does not produce reliable results without a priori information about the possible tampered area. In the block-wise approach, usually a sufficiently large portion of the image (e.g. a $B \times B$ block, with $B \geq 100$) is needed for a reliable statistical analysis of the footprint, so that only a coarse grained localization of tampering is possible.

A last class of unsupervised tamper localization algorithms is represented by forensic schemes designed to localize in an automatic way the tampered regions with a fine-grained scale, near to pixel-level resolution. These methods usually consider pixels of the image (or coefficients in a transform domain) as a mixture of two components, namely the tampered part and the untouched

part. If the two models can be separated, the likelihood ratio criterion gives a measure of the confidence for each pixel of being tampered. Thus, the output of these methods is a likelihood map indicating for each image pixel (or small image block) its probability of being tampered. To the best of our knowledge, only few algorithms exploiting the presence of double JPEG compression [4, 35, 15] or the artifacts due to CFA interpolation [14] belong to this category. The main limit of these approaches is the strong dependence of the results on local and global properties of the image (content, dimension, compression etc) and by the noiseness of the output map, so that it is always necessary to apply a postprocessing (often assisted) phase to obtain reliable results.

An important limit of all the approaches proposed so far is that they are based on the observation of a single forensic trace. By the light of our study about the importance of decision fusion for forgery detection, we argue that also fusing the output of several forgery localization tools could produce much more reliable localization maps.

The most intuitive approach to extend our data fusion framework to forgery localization would be to simply apply the whole procedure separately to each single element of the map (also called "analysis block", from now on). However, this choice is potentially misleading because of the nature of forgery localization tools. Indeed, as stated above, the accuracy of forgery localization tools is strongly affected by the local properties of the image: for example, very smooth or saturated regions are critical for many tools (see, for example, [14, 35]), so that values assumed by the map in those regions are less reliable. As a consequence, attention must be paid to properly interpret the output of the tools applied locally.

There is another fundamental difference between forgery detection and forgery localization. Independently from the analysis domain (e.g., pixel or DCT domain), unsupervised forgery localization tools typically assume that the image is the mixture of two components: one component deriving from parts of the image that were manipulated, and one deriving from unaltered parts [35, 15, 4]. When for some reason the two components are not correctly separated, the produced localization map is practically useless, although it assigns a sensible value to each region; Figure 4.15 shows an example of such

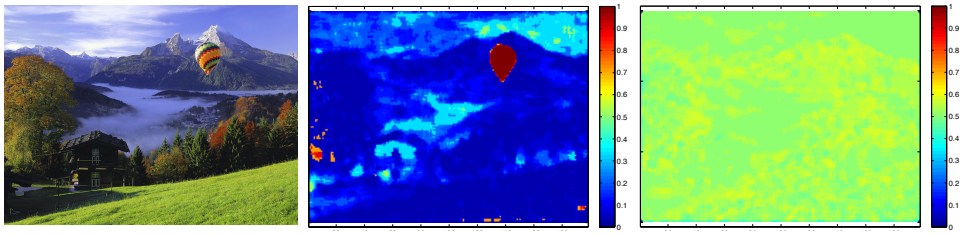


Figure 4.15: A forged image (the baloon is pasted) and the forgery localization map obtained with the tool in [35] on the tampered file (center plot) and on a re-compressed version of the tampered file (right-most plot). As we can see, in the latter case the map is not discriminative as it takes values near to 0.5 everywhere; on the contrary, the same value in the center plot clearly characterizes non-tampered regions.

a situation.

Based on the above discussion, we can state that extending the proposed framework to forgery localization is not trivial, though very interesting. Some preliminary work on this topic confirms that the obtainable benefits are in line with those observed for forgery detection.

4.5.2 Decision fusion as a means for countering anti-forensics

As a new challenge to Image Forensics, anti-forensic (AF) methods are emerging, whose goal is to remove the footprints left during processing, making forensic analysis harder [46]. On the other hand, AF tools may leave their own footprints, and counter-anti-forensic (CAF) tools are being designed to detect them as well.

In such a scenario, the forensic analyst needs to simultaneously tackle with both the variety of detectable footprints and the presence of an adversary equipped with AF tools. If CAF tools are available to the analyst, a more robust analysis can be carried out, provided that outputs from IF tools and their CAF versions are interpreted properly, thus going back to information fusion. To the best of our knowledge, so far information fusion and CAF have been investigated only separately.

In a recent work [47] we began investigating the possibility offered by the

adoption of a data fusion framework in a CAF scenario. Also in this case, one could opt for the intuitive solution of using all the available tools in an additive fashion ("OR" fusion rule), that means classifying the image as tampered when either an IF or a CAF tool detects the footprint it is looking for. This approach, however, does not take into account the following facts: i) IF tools may be searching for mutually exclusive traces, so some combinations of tool outputs could be excluded; ii) for a given footprint, IF and CAF algorithms are expected to be in contradiction, so if both kinds of tools detect their footprint this should at least raise some doubts about the correctness of the outputs; and iii) detecting some kinds of anti-forensic processing does not necessarily imply that the image is a fake (e.g., full-frame linear filtering may be seen as an AF technique, but usually it has to be considered a common, innocent operation). Based on the first results presented in [47], we can state that decision fusion becomes even more interesting in the presence of a forger who tries to conceal traces of tampering. Indeed, in such a situation it is fundamental for the analyst to gather as many clues as possible to reveal the presence of malicious activity.

There are several open issues in this direction: first, it would be interesting to put at work the theoretical advancements that are being pursued in the field of Adversarial Signal Processing, some of which include decision fusion in presence of an adversary [48]. It would also be interesting to investigate to which extent IF and CAF tools can be mixed together, in a balance between complexity of the system and granularity of the analysis.

4.5.3 Conclusion

As we can read on newspapers, the necessity of assessing the integrity and authenticity of digital images is growing every day. The awareness of people about the easiness with which fake contents can be created and about their dramatic impact on public opinion is also increasing; this is witnessed by some internet websites and services that appeared on the net in the few months: two noticeable examples are `www.stopfake.org` and `www.izitru.com`. During our studies, we had the possibility to see Image Forensics grow and reach new important results; we hope to have given a contribution to the development of a more comprehensive and practical way of interpreting the forensic analysis

of a digital image.

We believe the next years will bring image forensics at the next level, widening the interest of the commercial and forensic communities. The feedback coming from such communities can play a key role, because at the present time image forensic tools are often developed thinking to scenarios that do not exist outside the laboratory. On the other hand, there are issues that are considered "accessory" by our research community, like computational complexity, that impact severely the practical applicability of a solution. For these reasons, we believe it is necessary for image forensic researchers to establish contacts with the industry and law-enforcement agencies, so to let practical needs enter the lab and play a role in algorithm design and testing.

Part II

The Variation of Prediction Footprint: a Novel Tool for Video Forensics

Abstract

Recent advances in video compression have made possible the adoption of digital video technologies in many different fields, such as digital television broadcasting, video-telephony or Internet video streaming, among others. If we consider security applications, digital videos are even more important than images: every day millions of hours of video are recorded and stored by surveillance systems, that are meant to be used for security purposes. Once more, it is of fundamental importance to assess the authenticity of such footage before accepting it as an evidence.

Today video editing is no longer reserved to experts: many commercial and open-source software allow editing a video in a few minutes. As in the rest of this thesis, we are mainly interested in a particular kind of manipulation, namely splicing. As opposed to images, there are two ways to maliciously splice a video: the first one is to tamper with each frame separately, e.g., by introducing or hiding objects from frames; the second is to remove or insert a whole (group of) frames, without modifying the content of single pictures. Another important difference with respect to images is that digital videos are always stored in compressed format; as a consequence, any manipulation requires the forger to decompress the video, modify it, and finally re-compress it. This fact makes the detection of double compression an important task in video forensics.

After a brief introduction to video forensics (Chapter 5), our contributions are presented in Chapter 6: specifically, in Section 6.1 we propose a new footprint, called Variation of Prediction Footprint (VPF), and show that it can be used for double encoding detection. By leveraging on the VPF, we devise two forensic methods addressing both the localization of frame insertion and removal (Section 6.2) and the localization of manipulations within single frames (Section 6.3). Finally, Chapter 7 provides some concluding remarks and outlines the open issues. The works presented in this part of the thesis have been developed in collaboration with the University of Vigo (Spain).

Chapter 5

Introduction To Video Forensics

COMPARED to digital image forensics, video forensics is still an emerging field, for several reasons: creating fake images is much easier than creating fake videos; images are usually available in a few possible format, while videos can be encoded with many different schemes and, finally, videos usually undergo a stronger compression compared to images, making the forensic analysis more difficult. This contrasts with the fact that nowadays digital videos are probably used more than images for security tasks (e.g., video-surveillance systems are everywhere), so their trustability must be strengthened.

The goal of this chapter is to briefly review the state of the art in video forensics, with an emphasis on video splicing detection. Before digging into video forensic (Section 5.2), we find it appropriate to summarize some basic concepts of video coding (Section 5.1).

5.1 Video coding principles

The following paragraphs review briefly those aspects of video coding that are essential to understand the following chapters; the reader can easily find more details about video coding, e.g. in [49]. A digital video is basically a sequence of still pictures shot at a sufficiently high rate (typical values are 25 or 30 frames per second). The resulting signal can be conveniently compressed by reducing both spatial and temporal redundancy. In order to do that, common video coding algorithms like MPEG-2 [50], MPEG-4 [51] and H.264 [52] employ a block-based hybrid approach, and divide pictures into different types: intra-coded pictures, referred to as I-frames, and predictive-coded pictures, commonly named P-frames and B-frames (see Figure 5.1). During encoding, frames are grouped into GOPs (group of pictures) according

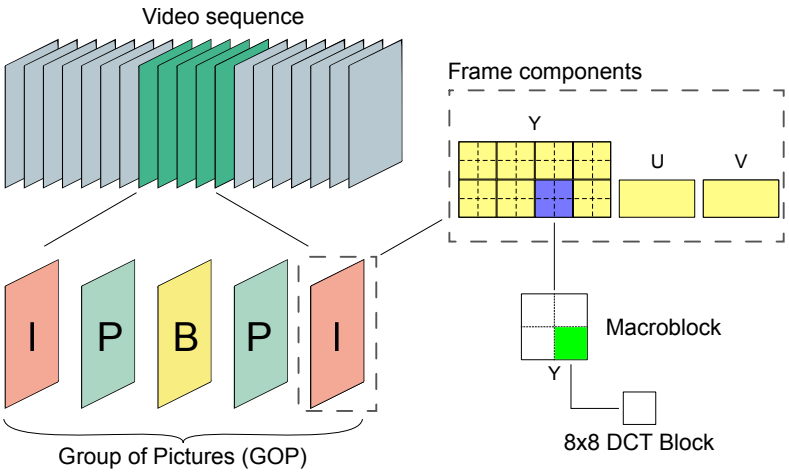


Figure 5.1: Structural elements in common video coding schemes.

to a structure that always starts with an I-frame and then allows a certain number of predictive frames. The total number of frames composing a GOP is called GOP size. When encoding a frame, the encoder divides it into macroblocks (MBs) and codes each MB separately; the most common size for MBs is 16×16 pixel, although this value can be adaptively chosen by some coding algorithms. For I-frames, MBs are always encoded without referencing to other frames, so that only spatial redundancy is eliminated. In order to achieve compression while maintaining a good perceptual quality, an approach similar to that used for JPEG image coding is adopted: the MB is sub-divided in 8×8 pixels tiles, and each tile is DCT transformed. The coefficients are quantized according to a matrix $W(i, j)$, where the element in position (i, j) is the quantization step for the same-positioned DCT coefficient. The employed matrix W is usually either the one defined by the video coding standard or is specified in the header of the video file; in both cases, it remains the same for all intra coded MBs. In order to allow the encoder to use different quantization strengths, the matrix can be multiplied by a scalar value k, so that the actual quantization step used for the (i, j)-th coefficient is given by $k \times W(i, j)$. Of course, higher values of k result in stronger compression and, hence, lower quality. The trade-off between quality and compression ratio can be adjusted either by fixing k, thus obtaining a constant bitrate (CBR) coding, or by

adjusting k dynamically based on the content of frames, resulting in a variable bitrate (VBR) coding. While VBR coding usually results in better perceptual quality, it is computationally heavier and does not fit well applications like video streaming.

Concerning predictive-coded pictures, the encoder adaptively chooses for each MB whether to use intra- or inter- prediction. In order to make this choice, the encoder searches the reference frame for a MB that matches well with the one at hand. If a good candidate is found, the displacement due to motion is compensated and the difference is evaluated, obtaining the so-called "prediction error". This error is usually a rather sparse signal, and it is compressed similarly to I-MBs: the DCT is calculated, and coefficients are quantized. The quantization matrix differs from the one used for I-MBs, because it is designed so to account for the different nature of the signal, in fact the prediction error contains more energy in high-frequency components. When a good candidate cannot be found, the encoder still has the option to encode the MB using intra-prediction, so that P- and B- frames can also contain I-MBs. Finally, the encoder also has the possibility to skip a MB, if the MB can be directly copied from a previous frame: these MBs are denoted as S-MBs. P- and B- frames differ in that P-frames can only make reference to previous frames, while also future frames are considered by B-frames (the "B" actually stands for "bidirectional"). Of course, using B-frames requires to encode pictures in an order different from the "natural" one: for a given GOP, the reference frames must be encoded first, before B-frames. The heavier processing and the necessity of keeping a frame buffer limit the use of B-frames in scenarios with restricted resources or with tight time constraints (e.g., real time encoding on mobile devices).

5.2 Previous works in video forensics

Video forensics is receiving increasing attention from the scientific community, so that many algorithms are emerging to analyze the integrity and the processing history of videos, as shown in a recent survey on this topic [6]. One of the most attracting topics has been the detection of multiple video encoding. The reason for such an interest is that, since videos are always compressed by

the acquisition device during capturing, any further processing will require to decode the video, perform changes and then "save" the manipulated video by recompressing it. Therefore, it is reasonable to see double encoding as a necessary condition for the presence of manipulations in a digital video. On the other hand, multiple compressions do not necessarily imply video forgery, and very different analysis techniques have been developed to assess the integrity of a digital video. In the following, we treat separately these two tasks.

5.2.1 Multiple encoding detection

When the research community began working on video forensics the first idea was to borrow as much as possible from image forensics. As a result, a large family of methods has been developed relying on the analysis of DCT coefficients. Su et al. [53] exploit the fact that the MPEG-2 video standard defines different quantization matrices for intra- and inter- coding modes; namely, the quantization steps for high frequencies are much higher in intra-coding mode. Due to this fact, when an I-frame is re-compressed as a P- or B- frame, its high-frequency DCT coefficients will be zero most of the times; on the other hand, frames that are encoded twice as predicted frames do not show this effect. Thus, authors propose to simply measure, for each frame, the energy of high-frequency DCT coefficients and use it to detect tampering using a threshold-based detector. The method can also detect double encoding when the GOP structure changes. Experimental results reported in [53] show a virtually perfect detection probability but also a high false alarm probability, namely 32%.

In [54], Liao et al. propose a method to detect double quantization in H.264 videos. Also in this case, authors leverage on the double quantization effect, and study the histogram of quantized DCT coefficients in the I-frames of the video. Experiments show that the method works only when the second encoding is at a higher quality than the previous one. Under these assumptions, an SVM classifier yields accuracies over 90%.
Tanfeng et al. propose to analyze the distribution of first significant digits (FSD) of DCT coefficients in I-frames: they derive a 12-dimensional vector from it, that is used to train a Support Vector Machine. The method can both detect double encoding and classify the second encoding as being at a

higher or lower bitrate than the first one. However, authors point out that this method has not been tested in cases where the two encodings are carried by different implementation of the MPEG-2 standard. Still based on FSD distribution analysis, Milani et al. developed a SVM classifier based on Benford's law [55]. This empirical law, which applies to many real-life sources of data, states that the distribution of the FSD of samples obeys a logarithmic decreasing law, so that "1" is the FSD in about 30% of the samples, "2" in 18% of them, and so on until "9", which occurs less than 5% of the times. The classifier in [55] is able to to tell how many compressions a given video underwent up to a maximum of 3 coding steps. Experiments show that the method performs very well in distinguishing never compressed video from compressed ones and, under some constraints on the quality of compressions in the chain, in distinguishing between videos encoded two or three times (75% of the test videos are labelled with the correct number of encodings).

Su et al. devised a method for double MPEG-2 compression is proposed that works also in presence of CBR coding [56]. The idea is to analyze only I-frames: MBs are partiotioned according to the quantization parameter that was used during encoding then the histogram of two specific DCT coefficients is considered for each group separately. Authors show that these histograms follow a monotonically decreasing trend when the video is encoded once, while they exhibit a convex shape in the presence of double encoding. Such a convex pattern is iteratively searched for, measured, and compared to a threshold to classify the video. Experimental results show an overall average between true positive rate (TPR) and true negative rate (TNR) of 94% when the video is re-encoded at the same quality (6 Mbps), which drops to 89% when the second encoding is carried with a bitrate of 4Mbps.

A promising approach has recently been proposed by Jiang et al. in [57] to detect double MPEG-4 encoding. Taking inspiration from a work by Pevny et al. on steganography [58], authors model adjacent DCT coefficients as a Markovian process. To do so, they evaluate the difference between adjacent coefficients (limiting the analysis only to some of them), and compute a transition probability matrix. Then, as in [58], they exploit some intrinsic regularities of such matrices to summarize them in a 162-dimensional vector, that is used to train a SVM classifier. The method is tested on videos encoded

twice in VBR mode, with relatively small quantization scale factors (up to 10); in these settings, performance are very interesting (mean value between TPR and TNR is around 95%) even when the second encoding is performed at a lower quality than the first one. The authors also show that, for some specific combinations of the first and second quantization, double encoding becomes undetectable with their method.

Leaving the DCT domain, Luo et al. propose to measure the strength of block artifacts (BAS) that are left during MPEG-2 encoding [59]. They generate several re-encoded versions of the video, removing each time one more frame from the beginning of the sequence, and then they measure the average BAS for each video. They show that, normally, this feature follows a periodic behavior; instead, when a video is compressed twice breaking the alignment between the GOP structures, an abnormal behavior appears, that in principle allows to detect the processing [59]. The authors do not provide a way to automatically detect such an abnormal behavior, nor they provide an analysis of the performance of the method over a set of test videos.

Up to now only a few works faced with the problem of detecting double compression performed using different codecs (also referred to as *transcoding*). Xu et al. consider the Fourier transform of DCT coefficients histogram, and they observe that such signal fits well a quadratic polynomial function if the video is encoded once using MPEG-4 [60]. On the contrary, spikes are present when the video is the re-encoded version of a previously MPEG-2 coded stream. By evaluating the goodness of the fitting, authors claim they are able to detect MPEG-2 to MPEG-4 transcoding. Performance depend on the bitrate used for the encoding: videos that were firstly encoded at higher bitrates will prove harder to be classified.

Starting from a video that is assumed to be double encoded, Bestagini et al. propose a way to identify the coding standard used for the first compression [61]. Their idea is to exploit the idempotency property of common coding schemes: assuming that VBR mode had been used for the first encoding, they re-encode the video under analysis with every possible encoder and every possible combination of the quantization parameters, then they measure the similarity between the resulting sequence and the analyzed video. The similarity shows a peak when the last encoding settings match those of

the first one. Performance are good when the second encoding undergone by the video is carried at very high quality, but rapidly decrease for stronger compression. Furthermore, the computational complexity of this approach is very high.

5.2.2 Video splicing detection

When talking about detection of forgeries in digital videos, it is helpful to distinguish between:

- *inter-frame* forgeries, where (group of) frames are entirely deleted, inserted or replicated;
- *intra-frame* forgeries, where the attacker alters the content of single frames (e.g. by introducing or removing objects).

Being very different, these attacks have been investigated using different techniques. An effective method for detecting the removal of frames was proposed by Wang et al. in [62]: the de-synchronization (induced by the tampering) between the GOP used for the first and for the second encoding is detected, by searching for a periodic behavior in the magnitude of motion vectors. This idea has been further investigated and improved by Stamm et al. in [63], where an unsupervised approach is devised for detecting frame removal/insertion and different strategies for GOP structuring are considered. A completely alternative method was developed by Su et al. in [53], leveraging on the different characteristics of quantization matrices employed for intra- and predictive-coded frames: if an I-frame is re-encoded as a P or B frame, a different quantization matrix will be used that preserves more energy in the high-frequency DCT coefficients. This fact is studied to detect periodic anomalies in the energy of some DCT coefficients, thus detecting that a de-synchronization in the GOP structure occurred. Finally, the same authors of [53] showed that their method for double encoding detection can also be used to detect frame removal [56].

Intra-frame forgery localization is probably the less studied field in video forensics, and most of the existing approaches work only under strict assumptions [6]. The most recent method is the one proposed by Wang et al. in [64],

where a DQ analysis is applied separately for each macroblock. The underlying idea is that when some of the MBs of a frame show the effects of double quantization while others don't, the last ones have been probably pasted from another sequence. This idea is borrowed from JPEG image forensics and, as such, the analysis makes sense only for frames that have been encoded twice as intra. The authors worked around this problem by assuming that Motion-JPEG encoding has been performed (i.e., only intra-coded pictures are used), thus heavily restricting the applicability of the method. Furthermore, in [64] the double quantization analysis is performed separately on each MB, leading to a computationally intensive analysis.

Chapter 6

Double Encoding Detection and Forgery Localization for Digital Videos

IN this chapter we propose a new video forensic footprint, called Variation of Prediction Footprint (VPF), and investigate its application to three different problems in video forensics. The chapter is thus divided in three main sections: the first one, Section 6.1, introduces the VPF and shows its usage as a tool for double video encoding detection, with estimation of the first compression GOP size in a double encoding setup. Then, Section 6.2 deals with the problem of detecting inter-frame video forgeries, showing how the VPF can be employed to detect this kind of manipulation. Finally, Section 6.3 considers the intra-frame forgery detection task and, by combining VPF and double quantization analysis, introduces a tool for authenticity verification also in this scenario.

Since the considered video forensic tasks are rather different, their experimental validation cannot be conveniently discussed in an aggregate manner (for example, it was not possible to use the same set of videos to test all the methods). For this reason, we discuss the experimental validation of each method within the section presenting the method itself.

6.1 Variation of prediction footprint for double encoding detection

As it emerged in Section 5.2.1, the state of the art is rich of methods targeting double encoding detection. Nevertheless, existing methods can hardly detect double encoding when different coding algorithms are used (e.g., MPEG-2 for the first compression and H.264 for the second); when this is possible, it comes at the cost of a computationally heavy analysis [61]. Furthermore, a

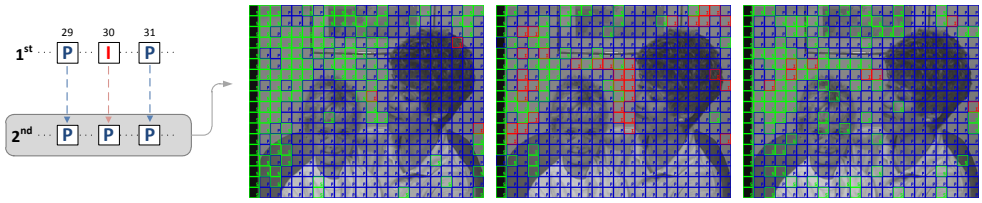
shared drawback of most existing techniques is the way they are affected by the second encoding, since their performance drop very rapidly as the strength of the last compression increases.

Motivated by these shortcomings and with the aim of generalizing the double encoding detection to a scenario with several codecs, different GOP sizes and distinct bitrates, we propose the VPF as a robust and very distinctive footprint based on the variation of the macroblock prediction types in the re-encoded P-frames. An advantage of this footprint is its presence in the twice encoded video without the need for further re-compression. Furthermore, given that the VPF appears only in P-frames that were intra-coded in the first encoding, we also describe a method to estimate the size of the GOP used in the first compression. It will become clear in the following sections that estimation of the GOP size is not only an important step toward assessing the processing history of a digital video, but can also enable further forensic analysis, e.g., tampering detection.

In the next section, we introduce our scenario for double encoding detection, analyzing why the VPF appears. In Section 6.1.2, we explain how this particular footprint can be measured and introduce a method for the estimation of the GOP size of the first compression. Section 6.1.3 presents the experimental results for evaluating the detection accuracy and the performance of the estimator.

6.1.1 The intuition behind the VPF

Although common video acquisition devices are becoming everyday more powerful, they are not capable of storing uncompressed videos, because this would require an exaggerated amount of persistent memory. As a consequence, acquisition devices compress the video during capturing, possibly at a very good quality. Since this encoding has to be carried in real-time, it is usually not possibile to exploit advanced features like B-frames (that require encoding pictures in non-temporal order, using a buffer), adaptive choice of GOP size, and so on. For these reasons, we assume that a video is captured using a device that performs a compression with an arbitrary yet fixed GOP size, denoted by G_1, and a fixed constant bitrate, represented by B_1. Then, we



(a) $G_1 = 30$, $G_2 = 50$ (b) Frame 29 (c) Frame 30 (d) Frame 31

Figure 6.1: Example where the VPF is present in frame 30. Leftmost picture shows the types of frames with indices 29, 30 and 31 for both compressions. The remaining three pictures represent the macroblock types for each frame. Red color is used for I-MB, blue color for P-MB and green color for S-MB. Both first and second encodings are carried out using the `x264` library, with a QP fixed to 20.

assume that the video is decoded and re-encoded: the second compression (that, for the moment, we assume to be temporally aligned with the first one) is carried out on the uncompressed sequence, but with a different GOP size, i.e., G_2 such that $G_2 \neq G_1$, and a fixed constant bitrate, i.e., B_2, that can be equal or different from the one used in the first compression. Considering this double encoding framework, we observe that a specific variation of the number of macroblocks coded as I-MB and S-MB shows up in the P-frames encoded as I-frames in the first compression.

To get a better understanding of this effect, let us consider the scenario illustrated in Figure 6.1: a double encoding with $G_1 = 30$ and $G_2 = 50$ is considered. The conversion between the types of frames for the indices 29, 30 and 31[1] is illustrated in Figure 6.1(a): we can see that frames 29 and 31 are predictive coded both in the first and in the second compression, while frame 30 turns from being an I-frame to a P-frame. Checking the corresponding macroblock types for the frame 30 in Figure 6.1(c), we can easily appreciate a noticeable increase of I-MBs and a considerable reduction of S-MBs. Hence, the VPF is present in frame 30.

The explanation of this effect is based on the different way an I-frame is encoded with respect to a P-frame. Generally, the quantization matrix or

[1]Note that we consider that the frame indices start counting from 0.

the quality factor for encoding an I-frame differs from the one considered for a P-frame because I-frames are used directly or indirectly as a reference for encoding several future frames. Besides, the following effects are observed:

- *Change of a P-MB or S-MB in homogeneous regions into an I-MB.* In general, the use of an I-MB in a P-frame is intended for encoding more efficiently a region where there is not a good match in the reference frames, like a new uncovered region. In this case, the compression of a reconstructed I-frame with a P-frame (whose reference frame will probably not be so correlated with this uncompressed frame) will lead to a less efficient encoding in general. However, if the changes introduced by the I-frame are small in homogeneous regions (for instance, like a change in the DC component of a whole block), then those blocks will be more efficiently coded as I-MBs than P-MBs where at least a motion vector should be considered and more bits would be needed. This is the main reason why I-MBs appear in smooth regions.

- *Change of S-MB in static regions into a P-MB.* The use of skipped macroblocks is very likely for any encoder given that neither residual information nor a motion vector is needed and a lot of bits are saved. Nevertheless, in the case we are studying, when a reconstructed I-frame replaces the encoding of a P-frame, small variations are introduced in static regions with respect to the reference frame and, thus, the use of a S-MB is no longer possible. Consequently, a P-MB must be used for satisfying the perceptual requirements.

As we stated earlier, even if each codec implements prediction and quantization in a different way, the final result is always coherent with the behavior described above, at least for those codecs based on DCT transform and motion compensation. Of course, the presence of VPF will also depend on the particular implementation of each codec, but since the main objective of any implementation is to reduce the bitrate according to a predefined quality, the observed behavior is consistent with any specific implementation.

As a conclusion, if we can detect these variations in the number of prediction types I-MB and S-MB, then we will be able to detect if a double encoding

of the same sequence has been carried out and, if this is the case, we have a way to estimate the size of the first GOP from these variations.

6.1.2 Measuring the VPF

In this section we show how the VPF can be used to detect double encoding and to estimate the GOP size of the first compression. The method we introduce is essentially based on two steps: first, the frames showing the VPF are localized, and the strength of the footprint is measured; secondly, since the obtained signal should show relevant peaks in correspondence of the positions of the I-frames of the first compression, a periodicity analysis is carried out.

In the rest of this section, the following notation is used: for a given video sequence $\underline{\boldsymbol{x}}(n)$, with $n = 0, \ldots, N-1$, being N the total number of frames, we denote with $i(n)$ and $s(n)$ respectively the number of I-MBs and S-MBs present in the n-th frame. We also recall that G_1 and G_2 are the GOP sizes used for the first and the second compression, respectively.

Peak extraction In this phase, we jointly consider the two signals $i(n)$ and $s(n)$. From Section 6.1.1, we know that in correspondence of those P-frames of the video that were encoded as intra in the first compression, the number of I-MB will increase while the number of S-MB will decrease. Obviously, we cannot consider directly $i(n)$, since for the I-frames of the second encoding all the macroblocks are I-MBs, resulting in very strong peaks that are not related to the first encoding. However, since G_2 is known, we can ignore peaks at frames $k\mathrm{G}_2$, where $k = 0, \ldots, \lfloor N/\mathrm{G}_2 \rfloor$. To do so, we substitute those elements of the array with the average value obtained from the previous and the following ones: $i(k\mathrm{G}_2) = (i(k\mathrm{G}_2 + 1) + i(k\mathrm{G}_2 - 1))/2$.

For the sake of clarity, we will denote by $\mathcal{P}$ the set of frames where the effect described in Section 6.1.1 is present. Specifically we define

$$\begin{aligned}\mathcal{P} = \{n \in \{0, \ldots, N-1\} : i(n-1) < i(n) \ \wedge \\ i(n) > i(n+1) \ \wedge \ s(n-1) > s(n) \ \wedge \ s(n) < s(n+1)\}.\end{aligned}$$

Based on this definition, we define a new vector that quantifies the strength

of the effect for every $n \in \mathcal{P}$ as follows

$$v(n) = \begin{cases} E(n), & \text{if } n \in \mathcal{P} \\ 0, & \text{otherwise} \end{cases}, \tag{6.1}$$

where $E(n)$ measures the strength of the effect in the n-th frame, defined as

$$\begin{aligned} E(n) = &\left|(i(n) - i(n-1))(s(n) - s(n-1))\right| + \\ &+ \left|(i(n+1) - i(n))(s(n+1) - s(n))\right|. \end{aligned} \tag{6.2}$$

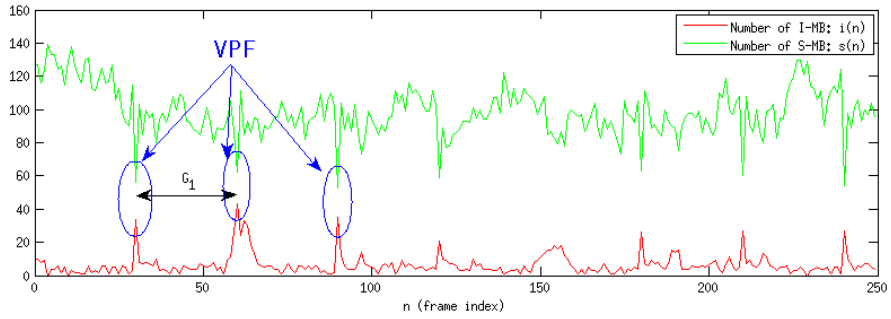
Indeed what we expect is, first, an increase in the number of I-MBs together with a decrease in the number of S-MBs and, then, a decrease in the number of I-MBs together with an increase in the number of S-MBs. This phenomenon is illustrated in Figure 6.2(a). Therefore, by taking the product of the variations of $i(\cdot)$ and $s(\cdot)$ we measure the magnitude of the effect we are considering, as shown in Figure 6.2(b).

Periodicity analysis The second step consists in investigating the periodicity of the extracted feature. If no periodic behavior is detected we can classify the video as singularly encoded; conversely, if a periodicity is present, then it will allow us to estimate G_1.

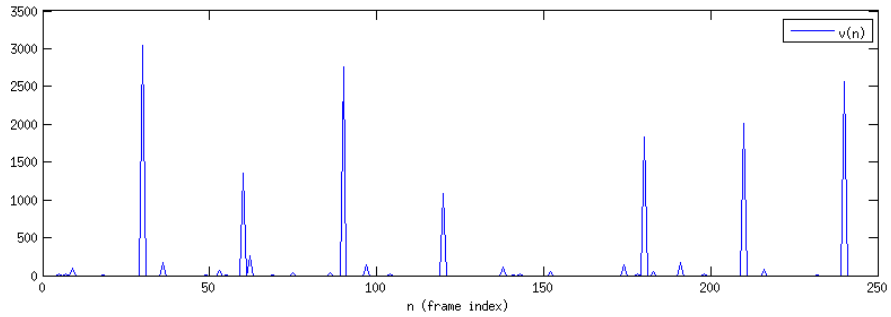
Usually, the periodicity of a signal is well-exposed in the frequency domain, e.g. by taking its Fourier transform. However, this approach is well-suited for cases where many periods of the signal are available, otherwise the frequency representation is noisy and periodicity estimation is inaccurate. On the other hand, we want our method to work also with a limited number of frames, so the frequency representation is not the best tool in our case.

For these reasons, we propose a simple yet effective approach for estimating the periodicity of peaks in $v(n)$, that is based on two steps: candidate GOP selection, and candidate evaluation.

Candidate GOP selection aims at determining a set of possible values for G_1. Since we are searching, in a sequence of integers, an element generating a subsequence of multiples of itself, it makes sense to restrict the search to the set of the Greatest Common Divisors (GCD) between all possible couples of elements of the sequence. Therefore, we define the set $\mathcal{C}$ of candidate GOPs



(a)



(b)

Figure 6.2: (VPF analysis carried on a double encoded video, with $G_1 = 30$ and $G_2 = 50$: in (a) the number of I-MBs and S-MBs as a function of the frame number is plotted, while (b) shows the derived strength signal, obtained according to equations (6.1) and (6.2).

as

$$\mathcal{C} = \{c \in \{2, \ldots, N\} : \exists\, n_1, n_2 \in \mathcal{P}, \mathrm{GCD}(n_1, n_2) = c\}.$$

Notice that evaluating $\mathcal{C}$ requires at most N^2 runs of the GCD algorithm, whose complexity is quadratic in the number of base-10 digits of its argument ($\lceil \log_{10} N \rceil$ at most, in our case). However, since the signal $v(n)$ is typically sparse (in the experiments presented in Section 6.1.3, $\sim 90\%$ components are null on average), the computational effort is surely affordable.

In the GOP estimation stage, each candidate value $c \in \mathcal{C}$ is associated with a goodness-of-fit value $\phi : \mathcal{C} \to \mathbb{R}$, that measures how well the choice of c

models the periodicity of the signal $v(n)$. Before giving the formal definition of $\phi(c)$, we briefly give the intuition behind this measure. Due to content related issues, like sudden changes of scene or strongly textured regions, the signal $v(n)$ could contain some noisy components, or could be missing some expected peaks at multiples of G_1. It is essential to define a goodness-of-fit measure that takes into account, for each candidate value c, the following aspects:

1. The strength of peaks that are located at multiples of c, given by
$$\phi_1(c) = \sum_{i=kc} v(i) \ , \quad \text{with } i \in \mathcal{P}, k \in \left[0, \left\lfloor \frac{N}{c} \right\rfloor\right] .$$
2. The absence of peaks that would be expected at multiples of c, quantified as
$$\phi_2(c) = \sum_{i=kc} \beta \ , \quad \text{with } i \notin \mathcal{P}, k \in \left[0, \left\lfloor \frac{N}{c} \right\rfloor\right] ,$$
where β is a constant penalization factor for missing peaks, that can be taken as $\beta = 0.1 \times \max_n\{v(n)\}$.
3. The strength of the most relevant periodic component with a period smaller than c, defined as
$$\phi_3(c) = \max_{z\in\{1,\ldots,c-1\}} \left\{ \sum_{k=0}^{\lfloor N/z \rfloor} v(kz) \right\} .$$

Then, we combine these three measures to define the function $\phi(c)$ as

$$\phi(c) = \phi_1(c) - \phi_2(c) - \phi_3(c), \tag{6.3}$$

where it is evident that ϕ_2 and ϕ_3 act as a penalization for the candidate c. Once the goodness of every candidate in $\mathcal{C}$ has been evaluated, we can classify the video as singularly or doubly encoded and, in the latter case, provide an estimate for G_1. The video $\underline{\boldsymbol{x}}(n)$ is assigned to a class with the following rule:

$$C(\underline{\boldsymbol{x}}) = \begin{cases} 1, & \text{if } \max_{c\in\mathcal{C}} \phi(c) > T_\phi \\ 0, & \text{otherwise} \end{cases} , \tag{6.4}$$

where T_ϕ is a threshold, $C(\underline{\boldsymbol{x}}) = 1$ for videos classified as doubly encoded, and $C(\underline{\boldsymbol{x}}) = 0$ for videos classified as singularly encoded. Whenever a video is classified as doubly encoded, the estimate of G_1 is

$$\hat{\mathrm{G}}_1 = \arg\max_{c \in \mathcal{C}} \phi(c). \tag{6.5}$$

6.1.3 Experimental validation

In this section we evaluate the performance of the proposed approach for double encoding detection and for GOP size estimation. To this end, a realistic setting is considered, which is often challenging for video forensics. We built the datasets for our experiments using 14 video sequences[2] with CIF resolution, i.e., 352×288 pixels, that are available in YUV-uncompressed format. Given that these sequences have different lengths, we always limit ourselves to the first 250 frames (that is, 10 seconds of video at 25 fps), in order to investigate the reliability of the proposed approach in the presence of short clips. Furthermore, in all the experiments, video encoding is performed specifying a target constant bitrate (CBR) and not by fixing the quantization parameters, since this is the typical encoding setting in a realistic scenario. As it was previously mentioned, adaptive GOP structures are not considered in this work. For all the tests, we have used the `libavcodec` and `x264` libraries (through `FFmpeg`) to encode/decode the videos.

Since we propose to use the VPF both for double encoding detection and GOP size estimation, we split the experiments into two parts; this choice also accounts for the different nature of these tasks, since detection and estimation need different evaluation criteria.

Double encoding detection

To test the discrimination capability of the proposed approach, we use the mentioned 14 raw sequences to create a dataset consisting of:

- 672 singularly encoded videos, by using all combinations of encoders and parameters in the right column of Table 6.1;

[2]Freely available at this website: http://trace.eas.asu.edu/yuv/
Chosen sequences are: *akiyo*, *bridge-close*, *bridge-far*, *coastguard*, *container*, *foreman*, *hall*, *highway*, *mobile*, *news*, *paris*, *silent*, *tempete*, *waterfall*.

- 672 doubly encoded videos, randomly sampling for each sequence 48 joint configurations for first and second encoding from those allowed by Table 6.1.

Since the proposed detection method relies on a threshold-based rule (see eq. (6.4)), we use ROC curves to evaluate its performance: we report in Figure 6.3 both the ROC of the proposed method on the whole dataset, and the performances obtained separately, differentiating the encoder employed for the second compression (which of course, is known to the analyst). It is worth noting that when the second encoding is carried out using H.264 (as we have seen, the most commonly used codec nowadays), the detector yields its best performance (94% detection rate for a false positive rate of 5%). This fact is not surprising: as usual in forensics, when the quality of the last compression is very low the footprint could be hidden by spurious effects. H.264 is known to provide better quality with respect to MPEG-x codecs for a fixed bitrate, thus limiting the negative impact on the detection of the VPF and, consequently, on the correct classification of the video. That said, the proposed method retains considerable accuracy also when MPEG-x codecs are used, and yields on average a detection rate of 80% when only 5% of false positives are allowed.

First GOP size estimation

In order to evaluate the performance of GOP size estimation we created a dataset of 32,256 doubly encoded videos, by compressing each of the 14 available sequences with all combinations of settings given in Table 6.1. Each sequence is analyzed in about 1.4 seconds on a desktop computer[3], but the actual analysis, that starts when types of macroblocks have been extracted, takes only 0.025 seconds.

We investigate the results of the estimation from different points of view: as a function of 1st and 2nd bitrate, as a function of the 1st and 2nd encoder, and as a function of the 1st and 2nd GOP size. Each time we investigate a parameter, all the other settings are marginalized out, i.e., results are averaged over them. Each estimate is classified as exact (that is, $\hat{G}_1$ from eq. (6.5)

[3]Intel Core2Duo @3.4GHz, 8GB RAM, running Ubuntu 10.04.

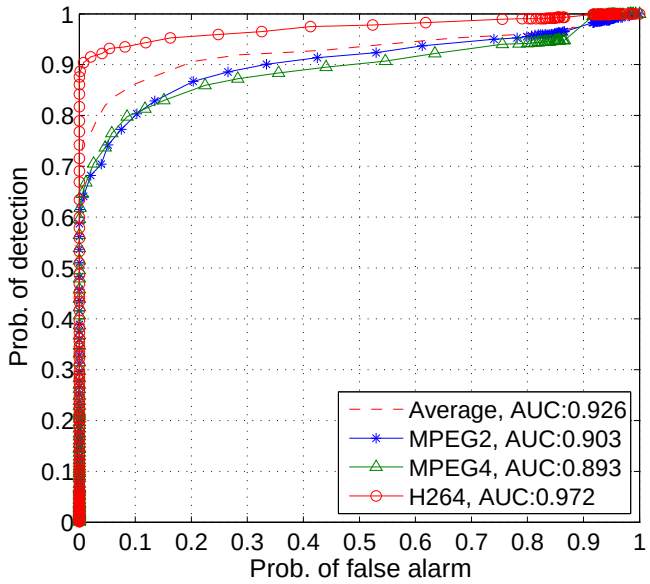


Figure 6.3: ROC curve for the proposed double encoding detector.

Parameters	**1st encoding**	**2nd encoding**
Encoder	{MPEG-2, MPEG-4, H.264}	{MPEG-2, MPEG-4, H.264}
Bitrate (kb/s)	{100, 300, 500, 700}	{100, 300, 500, 700}
GOP size	{10, 15, 30, 40}	{9, 16, 33, 50 }

Table 6.1: Parameters for Creating Doubly Encoded Sequences

actually matches G_1) or wrong, since we believe that having just an approximation of G_1 is not meaningful from a forensic point of view. Finally, since we are considering 14 different source sequences, for each experiment we report: i) average performance; ii) performance for the video sequence yielding best results (*paris* in all the experiments); iii) performance for the video yielding worst results (*waterfall* in all the experiments).

In the top of Figure 6.4 we report the performance as a function of B_1 - B_2. We see that lower bitrates for the first encoding result in higher performance, in agreement to what we said in Section 6.1.1: since low bitrates require strong quantization, acting like a lowpass filter, the number of blocks that

will be more conveniently encoded as I-MB increases. This is especially true for videos where uniform regions are present, like the *paris* sequence (which yields the best results), while textured content hides this phenomenon, as confirmed by the *waterfall* sequence (which is rich of textures). From the second compression point of view, it is confirmed that low bitrates negatively affect the performance, since they reduce the possible choices for the encoder when assigning macroblock types; nevertheless, even in the worst conditions, the proposed footprint is able to correctly estimate G_1 half of the times. The middle plot of Figure 6.4 shows the performance for different combinations of codecs. We see that reliability increases when the second encoding is carried out with H.264, in agreement to what we observed in Section 6.1.3 about the presence of VPF in doubly encoded videos.

Finally (bottom of Figure 6.4) we evaluate the performance for different combinations of G_1 and G_2. Results show an intuitive fact: as G_1 increases, the accuracy of the method drops. The most straightforward justification for this phenomenon is that, since we are using a fixed number of frames for the estimation, the higher G_1, the less number of periods we are able to observe. This, as expected, results in noisier estimates. Another interesting fact is that results improve as G_2 increases: in fact, this reduces the number of spurious effects induced by the GOP structure of the second compression.

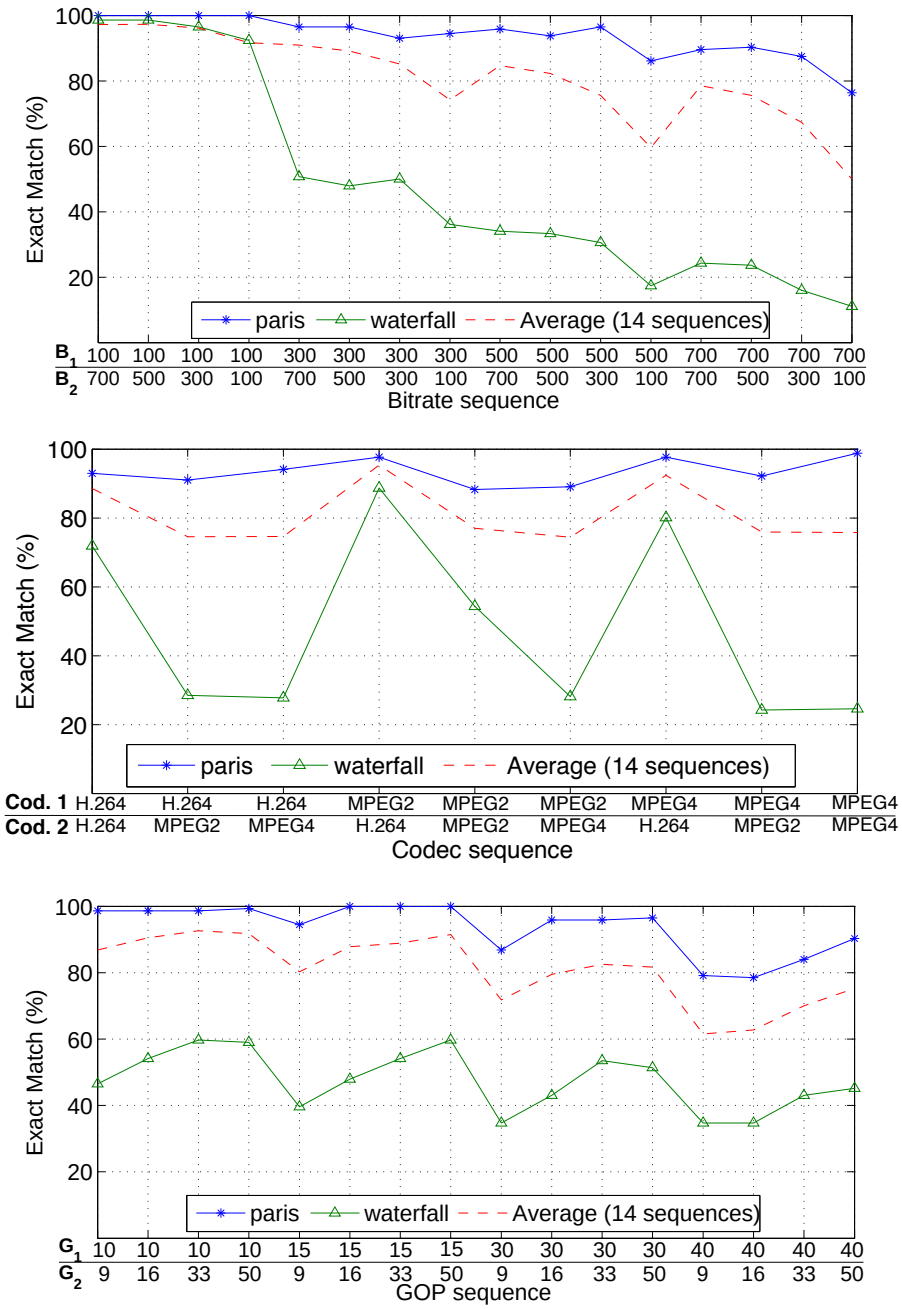


Figure 6.4: Performance of the method as a function of the B_1-B_2 bitrate (top), of the codec combination (middle), and of the G_1 - G_2 combination (bottom).

6.2 Detection of frame removal and insertion

As we anticipated in the introduction to this thesis, double encoding detection can often become a building block to devise splicing detection algorithms, as witnessed by many works in image forensics that are based on this idea [4, 35, 40]. Inspired by this paradigm, in this section we describe a novel method that, by building on the VPF analysis, enables detection of inter-frame forgeries in digital videos. Specifically, we first modify the VPF detection algorithm so to make it robust to frame removal between the two encodings; then, we design an algorithm that iteratively uses this generalized method to detect whether a misalignment in the frame structure of the video occurred between the two encodings. If this is the case, we detect the point where the misalignment occurs and also classify the attack distinguishing between frame deletion and insertion.

Compared to existing methods targeting the same task, the proposed approach has the advantage of being able to work when different coding algorithms are used in the first and second compression, a situation that is very common when dealing with DVs. Although this aspect was partially investigated also in [63], where different strategies for GOP structuring were simulated, the proposed algorithm has been practically tested using different off-the-shelf encoders, under various encoding configurations. Noticeably, the method retains acceptable performance even when the second compression is stronger than the former. Finally, the method inherits the computational simplicity induced by the nature of the VPF, so that the integrity of the video can be analyzed even without going through the decoding chain.

6.2.1 Shift-invariant VPF

Let us suppose that a video is captured, then some frames are removed using some editing software, and the final video is saved (i.e., re-encoded). If we maintain the assumptions stated at the beginning of Section 6.1.1, we still expect to find the VPF in the altered video but, due to the removal of frames, the peaks will no longer be periodic throughout the whole video. Instead, we expect to find a phase discontinuity located at the point where the cut took place, provided that the user did not eliminate a number of frames that is an

exact multiple of G_1. In the following, we show how to search and exploit such a discontinuity. First we modify the VPF measuring algorithm so to allow double encoding detection even when a group of leading frames is removed, and also estimate the number of removed frames. Then, we propose an iterative algorithm that, leveraging on the modified method, detects removal and insertion of frames within the video.

Increasing the robustness of VPF

We begin by noticing that even the removal of one frame at the beginning of the video (between the two encodings) would prevent the method described in Section 6.1 to work correctly. Indeed, periodic peaks would still appear, but since the set $\mathcal{C}$ is obtained by making use of the GCD operator, the correct value for G_1 could not be in $\mathcal{C}$ (neglecting noisy peaks). This becomes evident if we consider that when the r leading frames are removed before re-encoding, peaks would be located at $k\mathrm{G}_1 - r$; for $r \neq z\mathrm{G}_1$, $z \in \mathbb{N}$, such numbers are not divisible by G_1 and therefore the GCD between couples of elements in this set cannot be G_1. This limitation is clearly due to the fact that our original method was explicitly thought to detect double encoding, without considering any form of frame manipulation.

In order to overcome this drawback, we propose to evaluate the inherent periodicity of $v(n)$ by working on its autocorrelation $R_{vv}(\tau)$, evaluated for lags $\tau = 0, \ldots, N-1$. By using the very same approach for periodicity estimation presented in Section 6.1.2, but working on $R_{vv}(\tau)$ instead of $v(n)$, we achieve robustness against the removal of a set of leading frames. Furthermore, once we have an estimate $\hat{\mathrm{G}}_1$ of the period, we can also easily estimate the number of removed frames modulo $\hat{\mathrm{G}}_1$; this quantity will be termed *shift* from now on for brevity, and it will be denoted by $\hat{s}$. Let us introduce the following function:

$$\psi(s) = \frac{\sum_{i=0}^{\left\lfloor \frac{N-s}{\hat{\mathrm{G}}_1} \right\rfloor} v(i\hat{\mathrm{G}}_1 + s)}{N}, \tag{6.6}$$

that is the mean of the signal $v(\cdot)$ at multiples of $\hat{\mathrm{G}}_1$ displaced by s. Then, we can estimate the shift as:

$$\hat{s} = \arg \max_{s \in \{0,\ldots,\hat{\mathrm{G}}_1 - 1\}} \psi(s). \tag{6.7}$$

6.2.2 Iterative analysis for localizing frame removal

The VPF analysis works even when the number of available frames is limited: according to our experiments, $4{\cdot}\mathrm{G}_1$ frames are usually sufficient to obtain a correct estimate of G_1. This fact suggests that the video could be analyzed by splitting it into many sub-parts rather than as a whole. While this would not help in the case of a video that is just compressed twice, it would allow us to search for inconsistencies within a manipulated video. Such an approach would not be possible with the original method, since it is not invariant to frame shifting. However, by using the modified version proposed in Section 6.2.1, we can use a moving-window analysis to estimate the shift $\hat{s}$ at every iteration. Let us assume that the analysis window is moved by one frame at each iteration. In the case of a double compressed video (without frame removal), the shift is expected to be a periodic function of period G_1, which decreases linearly from $\mathrm{G}_1 - 1$ to 0 and then starts again from $\mathrm{G}_1 - 1$, as in top of Figure 6.5. On the contrary, if the number of removed frames is not a multiple of G_1, the proposed method would detect the correct value for G_1 both before and after the manipulation, but the shift would present a phase discontinuity in correspondence to the first removed frame, as in the bottom of Figure 6.5.

Inspired by this fact, we propose the following procedure to detect these discontinuities and remove those that are due to noise. Given a video, we first compute the signal $v(n)$ as defined in Section 6.1.2. In order to get a reliable estimate of G_1 even in the presence of a manipulation, we analyze the signal $v(n)$ with a sliding window of size W, shifting it by one frame at a time. At each step, we use the modified approach proposed in Section 6.2.1 to estimate G_1, thus obtaining a signal $g(n), n = 0, \ldots, N - W$ containing the estimate of G_1 at each window position. Then, the overall estimate of G_1 for the video is defined as:

$$\tilde{\mathrm{G}}_1 = \mathrm{mode}(g(n)), \tag{6.8}$$

where mode is the statistical mode of the signal. Using $\tilde{\mathrm{G}}_1$, we repeat the window-based analysis to estimate the value of the shift at each window, according to equation (6.7), thus obtaining the shift array $\sigma(n), n = 0, \ldots, N - W$ that was plotted in Figure 6.5. To better highlight the phase discontinuity in this signal, we remove the periodic component due to the shift of the

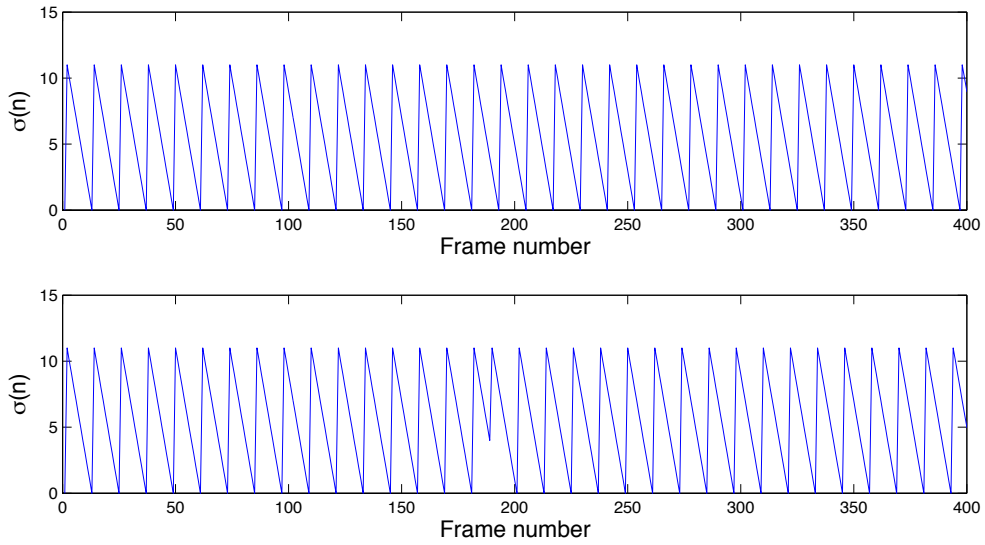


Figure 6.5: An example of the shift signal $\sigma(n)$ for a double compressed sequence (top) and a manipulated sequence (bottom).

window, and define:

$$\sigma_h(n) = \mod (n + \sigma(n), \tilde{\mathrm{G}}_1). \tag{6.9}$$

In the ideal case, $\sigma_h(n)$ should be a step function with value 0 until the cut is reached, then it should move to the value $C \bmod \mathrm{G}_1$, where C is the number of removed frames. Due to noise in the original signal $v(n)$, however, other peaks may be present in $\sigma_h(n)$ that are not related to manipulations. Such an impulsive noise can be safely mitigated by median filtering the signal $\sigma_h(n)$. Since we are mostly interested in the position of the discontinuity, the first order derivative $\sigma'_h(n)$ is computed from the filtered signal, and the set $\mathcal{L}$ is defined as:

$$\mathcal{L} = \{l : \sigma'_h(l) \neq 0, l \in \{0, 1, \ldots, N - W\}\}. \tag{6.10}$$

If the set $\mathcal{L}$ is empty, no frame removal or insertion is detected. If the set is not empty, a further analysis is carried out before classifying the video as manipulated. Indeed, we know that frame removal would cause a durable change in the step function $\sigma_h(n)$, since once the phase is broken the displacement should remain constant. This information helps us in discarding elements in

$\mathcal{L}$ that are due to noisy measurements; we propose to adopt the algorithm in Figure 6.6, where the value Δ denotes the minimum allowed distance between two consecutive peaks. The underlying idea is the following: when two elements $l, l' \in \mathcal{L}$ are found, first their "sign" (positive or negative derivative) is compared; if the signs are the same, it means that we have two successive increments/decrements in the value of $\sigma_h(n)$, and this is not plausible under the assumption of a single cut. So, based on the distance between the discontinuities, we decide whether to detect a cut in the former and discard the latter, or simply discard the former and continue with the analysis. On the other hand, if the two discontinuities are not of the same sign, we check whether $\sigma_h(n)$ takes the same values prior and after the discontinuities: if this is the case, a cut is detected in l and l' is discarded, otherwise l is discarded and the analysis continues.

6.2.3 Localization of frame insertion

Let us now suppose that the manipulated video is obtained by inserting a group of frames coming from a different sequence, that was encoded using a constant GOP size $\mathrm{G}_1^{\#}$ different from G_1. Under these conditions, the described algorithm is expected to detect two cutting points, one at the beginning and one at the end of the injected sequence, where the phase discontinuity changes again. Let us denote as l_1 and l_2 the frame indexes where the first and the second discontinuity are localized, respectively. In order to distinguish between two independent frame removals and a frame insertion, we can verify whether the estimate of the GOP value between l_1 and l_2 coincides with the estimate on the rest of the sequence or not. Formally, we propose to calculate:

$$\tilde{\mathrm{G}}_1^{\#} = \mathrm{mode}(g(n)), \quad l_1 \leq n \leq l_2 \tag{6.11}$$

$$\tilde{\mathrm{G}}_1 = \mathrm{mode}(g(n)), \quad n < l_1 \vee n > l_2. \tag{6.12}$$

Finally, frame insertion is detected if $\tilde{\mathrm{G}}_1^{\#} \neq \tilde{\mathrm{G}}_1$, and $\tilde{\mathrm{G}}_1^{\#}$ is chosen as the estimate of the GOP size for the pasted sequence.

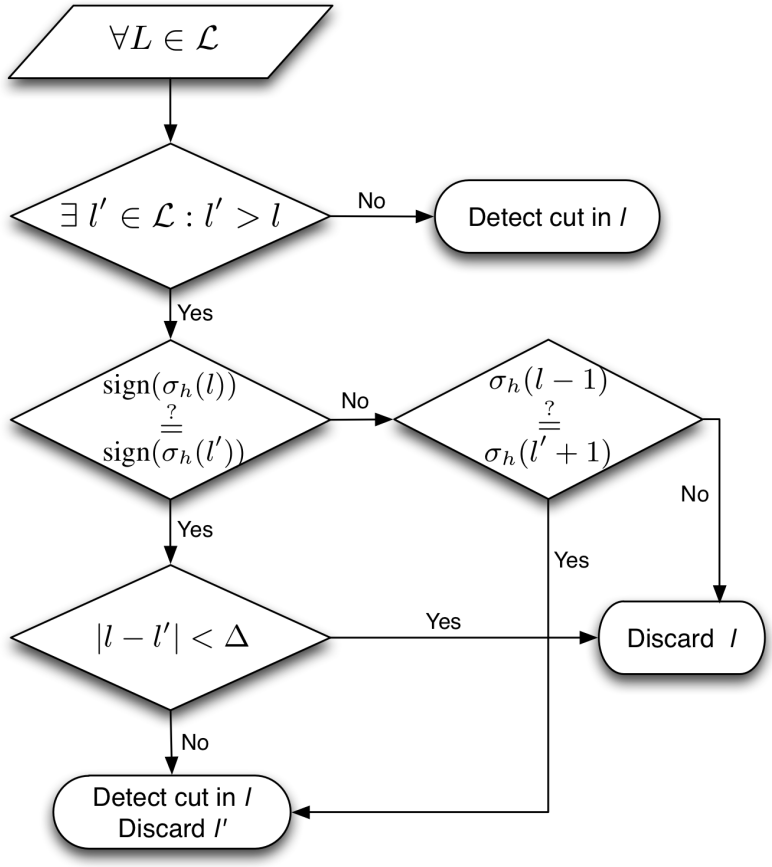


Figure 6.6: Algorithm for cut detection, given the set of indices where the signal $\sigma'_h(n)$ is not null.

6.2.4 Experimental validation

In this section we evaluate the performance of the proposed method for inter-frame forgery detection. Classification between single- and double-encoded videos is not addressed here because such a functionality has already been tested in Section 6.1.3.

We generated the dataset starting from the set of 14 YUV uncompressed CIF resolution videos mentioned in Section 6.1.3. Each sequence has been ex-

tended to 1250 frames using mirrored frames repetition, thus avoiding abrupt changes in content. Throughout the experiments, we used MPEG-2, MPEG-4 and H.264 as possible encoders[4]. Bitrates were taken from the set {100, 300, 700} (CBR compression mode was used); values for G_1 were allowed to take values in {12,15,31}, while values for G_2 were taken in {10,20,25}. By taking all possible combinations of the above parameters, a total of 10206 double compressed videos were generated. To generate videos with frame deletion, each sequence was decoded after the first compression, a group of 100 frames was removed from a random point, then the second encoding was performed. This originated 10206 manipulated test sequences.

Given these two sets of videos, we run the analysis, using $W = 100$, $\Delta = 50$, and using a width of 200 for the median filter. Since the VPF is affected by the encoding settings, the performance also depends on them; the two most important parameters are the codec and bitrate used for the first and second compression (see Section 6.1.3). For this reason, we plot marginalized results in Table 6.2 for the first/second compression codec and in Table 6.3 for the first/second compression bitrate. Accuracies were computed averaging the true negative rate and the true positive rate; a video was considered as correctly classified when the localized cutting point was less than W frames away from the true cutting point.

[4]The `libavcodec` and `x264` libraries were used, through `FFmpeg`.

C_1/C_2	MPEG-2	MPEG-4	H.264
MPEG-2	83.38 %	81.70 %	95.46 %
MPEG-4	81.83 %	79.39 %	96.25 %
H.264	76.10 %	76.19 %	88.32 %

Table 6.2: Accuracy in distinguishing double compressed and manipulated (by frame-removal) videos. Rows contain the first codec, columns the second.

B_1/B_2	100	300	700
100	85.44 %	89.68 %	91.27 %
300	77.95 %	86.55 %	88.80 %
700	75.93 %	81.04 %	81.94 %

Table 6.3: Accuracy in distinguishing double compressed and manipulated (by frame-removal) videos. Rows contain first bitrate (Kb/s), columns the second.

C_1/C_2	MPEG-2	MPEG-4	H.264
MPEG-2	77.48 %	77.91 %	89.99 %
MPEG-4	75.15 %	74.24 %	90.52 %
H.264	70.79 %	72.80 %	84.16 %

Table 6.4: Accuracy in distinguishing between frame insertion and frame removal. Rows contain the first codec, columns the second.

B_1/B_2	100	300	700
100	86.15 %	89.71 %	90.72 %
300	70.39 %	85.30 %	89.93 %
700	55.58 %	67.51 %	81.47 %

Table 6.5: Accuracy in distinguishing between frame insertion and frame removal. Rows contain first bitrate (Kb/s), columns the second.

To evaluate the performance of frame insertion localization, we designed the following experiment: starting from singularly compressed sequences, we selected two of them at random, decoded them, and pasted 350 frames from one sequence into the other one; finally, the spliced sequence was compressed. Since frame insertion can be detected only when the GOP sizes of the spliced sequences are different, we allowed G_1 to take values in $\{12, 15\}$ while $G_1^{\#}$ was set to 10. The GOP size of the second encoding, that is G_2, was chosen in the set $\{33, 40\}$. As stated in Section 6.2.3, the localization of frame insertion follows the localization of multiple cutting points. For this reason, we tested the capability of the method of distinguishing between the two manipulations. To this end, frame insertion within a video was considered as correctly localized (true positive) when the beginning of the insertion was detected no more than W frames away from its actual position and $G_1^{\#}$ was correctly estimated. On the other hand, true negatives are obtained when no frame insertion was localized in a video in which only frame removal took place. Accuracies, obtained by averaging the true positive and true negative rates, are reported in Table 6.4 for different first/second employed codecs and in Table 6.5 for different first/second encoding bitrate (results are averaged across all possible encoding configurations for the pasted segment).

6.3 Intra-frame tampering localization through VPF and double quantization analysis

In this section we present a method for localizing intra-frame forgeries in MPEG-2 videos. In this tampering scenario the forger starts from a video sequence, decodes the video and alters the content of a group of frames, e.g., by introducing and/or removing objects. As a final step, the video is re-encoded so to be stored and/or transmitted. The method proposed in this section works under the assumption that both encoding are carried out by using MPEG-2 in VBR mode (fixed quantizer, see Section 5.1), and that a different GOP size is chosen for the first and second encoding.

Since the proposed approach is composed of two separate analysis steps, we first give a sketch of the idea (Section 6.3.1) and then describe each step

in detail (Sections 6.3.2 and 6.3.3).

6.3.1 Sketch of the method

Forgery localization has been extensively investigated in image forensics, leading to the creation of a family of methods that takes an image as input and produce a probability map as output, associating each pixel (or block of pixels) with a probability of being tampered. Within this family, methods based on double quantization (DQ) analysis are very effective in terms of localization accuracy. DQ analysis exploits the trace left by multiple JPEG compressions of the image: at each compression step, the image is transformed in the DCT domain and the resulting DCT coefficients are quantized. Double quantization leaves characteristic traces in the signal, that can be leveraged to understand whether the whole image has been compressed more than once, and also to detect inconsistencies between the number of quantizations undergone by different regions of the image. It goes without saying that this kind of analysis is applicable when the target image is available in JPEG format.

Turning to videos, application of DQ-based forgery localization on frames is not possible *in general*. Indeed, in contrast to JPEG compression, video coding algorithms employ error prediction: even when the video is compressed twice with the very same coding parameters, at each encoding a new prediction is made for all P-frames thus generating completely new prediction errors. This makes it nearly impossible to model mathematically the behavior of coefficients of predicted frames. As a consequence, while it is still possible to detect traces of double compression in P-frames, the absence of an accurate mathematical model prevents forgery localization.

Although DQ analysis is not applicable to P-frames, there is still a chance for intra coded frames, for which the compression algorithm is much more similar to JPEG (see Section 5.1). More precisely, if a frame is compressed as intra *both* by the first and the second codec, we can treat it like a still image that has been JPEG compressed twice. This is especially true for the MPEG-2 codec, while the more recent MPEG-4 and H.264 allow variable block size decomposition and intra-frame prediction, thus deviating consistently from the JPEG analogy. Therefore, we study the possibility of using DQ analysis for forgery localization in MPEG-2 videos, and only for frames

that were encoded as intra both in the first and second encoding. In order to localize such frames we use the VPF: after estimating the GOP size of the first encoding we can easily retrieve the location of frames that have been intra coded twice; a DQ analysis is then carried on these frames, employing a mathematical model tailored for MPEG-2 compression. Figure 6.7 gives a schematic representation of the approach.

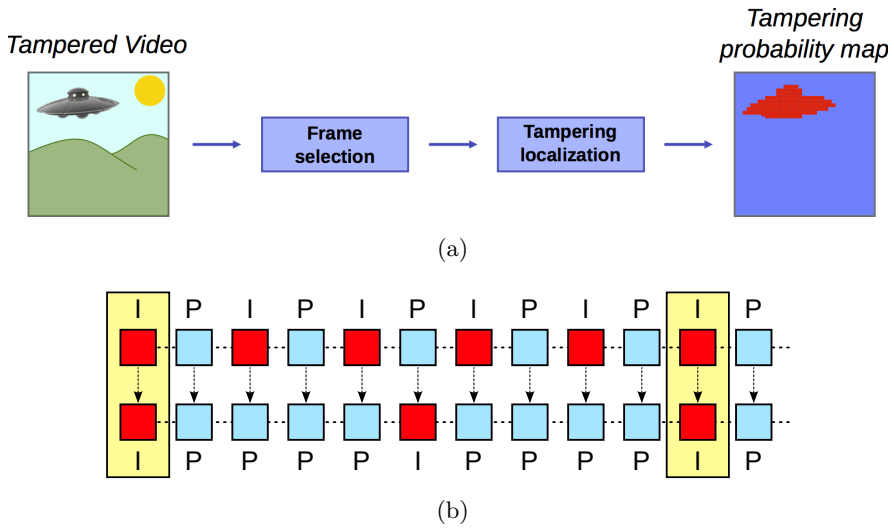


Figure 6.7: Schematic representation of the proposed approach: a subset of the frames is selected, then tampering localization is carried separately on each frame to obtain a forgery map (a). Specifically, those frames that have been intra-coded twice are selected (b).

6.3.2 Detection of frames encoded twice as intra

Let us assume that a video, composed by N frames, has been encoded twice using G_1 and G_2 as the GOP size for the first and second encoding respectively, where $G_1 \neq m \cdot G_2, \forall m \in \mathbb{N}$ (this condition is necessary since all the I-frames of the second compression must be excluded from the VPF analysis, see Section 6.1.2). Assuming a fixed GOP structure, the set of indices of the frames that

have been intra-coded twice is

$$\mathcal{C}_{\mathrm{G}_1,\mathrm{G}_2} = \{n \in \mathbb{N} : n = m \cdot \mathrm{lcm}(\mathrm{G}_1, \mathrm{G}_2) \wedge n \leq N, \forall m \in \mathbb{N}\},$$

where $\mathrm{lcm}(\mathrm{G}_1, \mathrm{G}_2)$ represents the least common multiple between G_1 and G_2. The cardinality $|\mathcal{C}_{\mathrm{G}_1,\mathrm{G}_2}|$ of the set is given by:

$$|\mathcal{C}_{\mathrm{G}_1,\mathrm{G}_2}| = 1 + \left\lfloor \frac{N}{\mathrm{lcm}(\mathrm{G}_1, \mathrm{G}_2)} \right\rfloor,$$

where $\lfloor \cdot \rfloor$ stands for the floor function. By using the method presented in Section 6.1, G_1 can be estimated, allowing to perform DQ-based forgery localization every $\mathrm{lcm}(\mathrm{G}_1, \mathrm{G}_2)$ frames. For relatively prime values of G_1 and G_2 the analysis can be carried out only once every $\mathrm{G}_1 \cdot \mathrm{G}_2$ frames. On the other hand, the GOP size is usually chosen from a set of possibilities, like 12 for PAL videos, 15 for NTSC videos, while hand-held devices like mobile phones often choose a GOP size around 30. At a frame rate of 25 fps, combinations of the mentioned values for G_1 and G_2 result in a satisfactory time resolution for the analysis.

A critical observation regards the use of the method proposed in Section 6.1: such a method, in fact, was tested on double encoded videos, without either intra- or inter- frame modifications between the two encodings. On the contrary, here we are assuming that the video is manipulated (by altering the content of a group of frames) before the second compression takes place. The robustness of the VPF in this scenario is addressed in Section 6.3.4.

6.3.3 Double quantization analysis for MPEG-2 intra-coded frames

According to the scenario we are considering, tampered frames that have been encoded twice as intra will consist of two groups of pixels: those that have not been modified, and which underwent a double quantization, and those that have been introduced between the two encodings (Figure 6.8). Even when these latter pixels come from a compressed sequence, they will unlikely be pasted respecting the 8×8 quantization grid of the host frame and, therefore, will not show traces of double quantization after the second encoding, thus

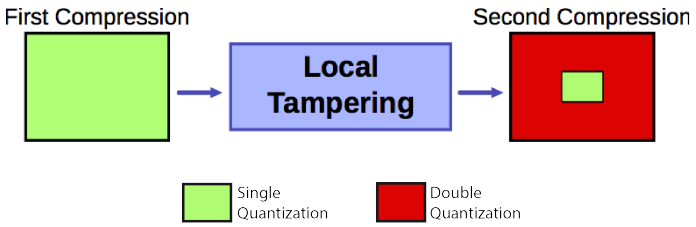


Figure 6.8: Idea underlying DQ-based forgery localization: pixels that do not expose traces of double quantization are considered as tampered.

making localization possible. This idea was firstly introduced in [4] and further refined in [35].

If we consider the histogram of a specific set of DCT coefficients (e.g., those in position (0,1) in all 8×8 blocks), we should see a mixture of two components: a comb-shaped component due to unaltered, double compressed regions, and a "standard" component due to newly introduced regions. As shown in Figure 6.9, we denote by $h_{DQ}(x)$ the normalized histogram[5] of DCT coefficients that underwent a double quantization, and by $h_{SQ}(x)$ the normalized histogram of coefficients that underwent a single quantization. The mixture of these two components can be modeled, as suggested in [35] as:

$$h_{mix}(x) = \alpha h_{DQ}(x) + (1 - \alpha) h_{SQ}(x), \tag{6.13}$$

where α is a scalar value determining the balance of the mixture. If we had access to h_{DQ} and h_{SQ}, we could associate each coefficient separately to the probability of belonging to one of the two components. Unfortunately, the only histogram available to the analyst is h_{mix}. The key idea, first proposed in [35], is to search for an estimate of both h_{SQ} and h_{DQ}.

Concerning h_{SQ}, we follow the approach in [35] and resort to the well known *calibration* technique, that was introduced in [65]. Given a quantized signal, calibration allows to obtain a reliable estimate of the histogram of the original (unquantized) signal. In the case of a quantized frame, calibration simply requires to dequantize and inverse transform the frame, remove a few

[5]Given the histogram of a distribution of samples, its normalized version is obtained by dividing each bin count by the total number of samples (i.e., the sum of all bin counts).

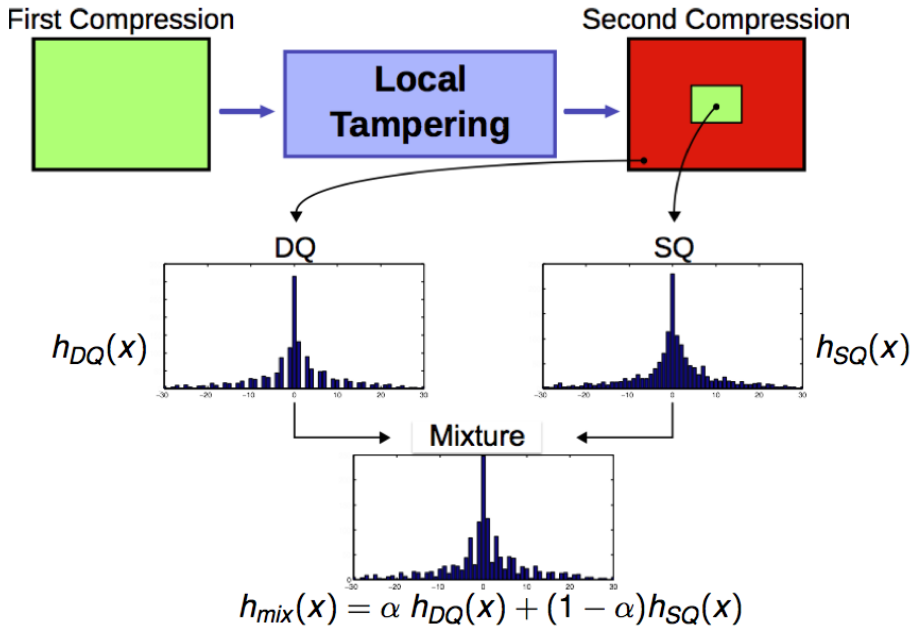


Figure 6.9: Graphical explanation of the DQ effect in presence of a tampered region.

rows and/or columns, and then go back to the DCT domain. We will denote by $h_{cal}(x)$ the histogram of calibrated DCT coefficients, that will be used as an approximation of never-quantized coefficients. Since k_2 is available from the video bitstream, we can write:

$$h_{SQ}(x) \simeq \tilde{h}(x) = \Delta_{k_2}(h_{cal}(x)), \tag{6.14}$$

where Δ indicates the quantization function, that will be described later in detail for the case of MPEG-2 compression.

After obtaining an estimate of h_{SQ}, we need to estimate also the other component of the mixture, that is h_{DQ}. Also in this case, we can follow the reasoning proposed in [35]: if we knew the quantization multiplier of the first compression, k_1, we could count how many bins of the histogram of unquantized coefficients fall within each bin of h_{DQ}, starting from the estimate we have of h_{SQ} (see Figure 6.10 for a graphical description). For the moment, let us call $n(x, k_1, k_2)$ the function implementing such a count, allowing us to

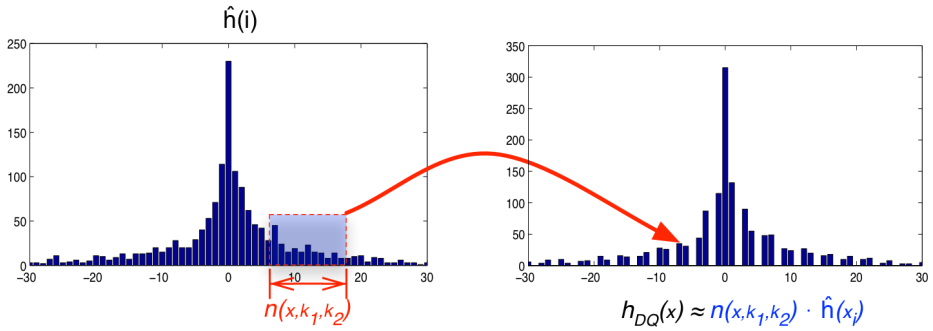


Figure 6.10: Graphical representation of the method for estimating h_{DQ}.

write:

$$h_{DQ}(x) = n(x, k_1, k_2) \cdot \tilde{h(x)}; \tag{6.15}$$

we will derive $n(\cdot)$ for the case of MPEG-2 compression later.

Let us now denote with $\mathcal{H}_0$ and $\mathcal{H}_1$ the hypothesis of being tampered and original, respectively, for the i-th coefficient. In the considered scenario (Figure 6.9), we can write:

$$p(x_i|\mathcal{H}_0) = h_{SQ}(x_i) \simeq \tilde{h}(x_i) \tag{6.16}$$

and

$$p(x_i|\mathcal{H}_1) = h_{DQ}(x_i) = n(x_i) \cdot \tilde{h}(x_i), \quad x_i \neq 0. \tag{6.17}$$

By Bayes rule, and assuming equal priors:

$$p(\mathcal{H}_1|x_i) = \frac{P(x_i|\mathcal{H}_1) \cdot P(\mathcal{H}_1)}{P(x_i|\mathcal{H}_1) \cdot P(\mathcal{H}_1) + P(x_i|\mathcal{H}_0) \cdot P(\mathcal{H}_0)} \tag{6.18}$$

$$= \frac{1}{1 + \frac{P(x_i|\mathcal{H}_0) \cdot P(\mathcal{H}_0)}{P(x_i|\mathcal{H}_1) \cdot P(\mathcal{H}_1)}} \tag{6.19}$$

$$= \frac{1}{1 + n(x_i)}, \tag{6.20}$$

where $n_i(x)$ is the $n(x)$ function for the i-th coefficient.

These steps are carried out separately for each DCT coefficient (usually only the first dozen of AC coefficients are used for the analysis). Then, for each

8×8 block, the probability of being tampered is "accumulated", assuming statistical independence between DCT coefficients, as:

$$p = 1/\left(\prod_{i|x_i \neq 0} n_i(x_i) + 1\right). \tag{6.21}$$

In order to be able to compute the above probabilities, we still need to: i) define a way to estimate k_1, since it is not available to the analyst; ii) derive the function $n(\cdot)$ for the MPEG-2 quantization scheme. To face with these problems, we necessarily deviate from the method proposed in [35] for JPEG images. In fact, some significant differences exist between the two cases:

- the dequantization formulas are different: MPEG-2 dequantization involves rounding towards zero [50], an operation that is not necessary in JPEG;
- in JPEG, the 8×8 quantization matrix defined in the header determines the quality factor; in MPEG-2, instead, there is a reference quantization matrix, defined in the standard, that is parameterized by the multiplier k to adjust the quantization strength (see Section 5.1);
- in JPEG, the quantization matrix is the same for the whole image; this holds also for MPEG-2 when a fixed quantizer is used, while the quantization matrix may change from frame to frame or MB to MB, for instance, for CBR coding.

Each of these facts has a direct implication on the method in [35]:

- because of the different dequantization formula, the function $n(\cdot)$ will likely change for MPEG-2;
- since all quantization coefficients are obtained by multiplying the reference matrix by k, it is not necessary to estimate a different quantization step for each coefficient (we can directly estimate k);
- in the case of CBR coding, that is left for future work, MBs that are not quantized using the same k must be analyzed separately.

That said, in the following we derive the function $n(x)$ for MPEG-2 quantization scheme. Let us denote the never-compressed DCT coefficient on the i-th row and j-th column of an 8×8 block with $x(i,j)$, where $i,j \in \{0,\dots,7\}$. Similarly, we denote with $u_1(i,j)$ the quantized version of the coefficient. According to the MPEG-2 standard [50], and following the proposed notation, the de-quantized version[6] of the DCT coefficients coming from a single compressed intra-coded frame is:

$$x_1(i,j) = \text{sign}\left(u_1(i,j)\right)\left\lfloor \frac{W(i,j)\cdot|u_1(i,j)|\cdot k_1}{16}\right\rfloor \tag{6.22}$$

for all coefficients apart from the DC, where $|\cdot|$ is the absolute value operator, $W(i,j)$ denote an element in the 8×8 quantization matrix, and k_1 and k_2 are the multipliers that parametrize the quantization matrix in the first compression and in the second compression, respectively.

Starting from (6.22), the most intuitive way to define the quantization is:

$$u_1(i,j) = \left[\frac{16\cdot x(i,j)}{k_1\cdot W(i,j)}\right], \tag{6.23}$$

where $[\cdot]$ represents the rounding to nearest integer operation.

Now, let us denote by $u_2(i,j)$ the re-quantized version of $x_1(i,j)$ (i.e., the double quantized coefficient): according to (6.22) and (6.23), omitting the position indices for brevity, we have:

$$u_2 = \left[\frac{16}{k_2\cdot W}\left(\text{sign}\left(\left[\frac{16\cdot x}{k_1\cdot W}\right]\right)\left\lfloor\frac{W\cdot\left|\left[\frac{16\cdot x}{k_1\cdot W}\right]\right|\cdot k_1}{16}\right\rfloor\right)\right].$$

Let us now determine the interval of values of x that are mapped to u_2 after the double quantization process. By definition u_2 is integer, so:

$$u_2 = [n] \qquad \Longrightarrow \qquad u_2 - \frac{1}{2} \le n < u_2 + \frac{1}{2}.$$

Hence, in our case:

$$u_2 - \tfrac{1}{2} \le \tfrac{16}{k_2\cdot W}\cdot\left(sign\left(\left[\tfrac{16\cdot x}{k_1\cdot W}\right]\right)\cdot\left\lfloor\frac{sign\left(\left[\frac{16\cdot x}{k_1\cdot W}\right]\right)\cdot W\cdot\left[\frac{16\cdot x}{k_1\cdot W}\right]\cdot k_1}{16}\right\rfloor\right) < u_2 + \tfrac{1}{2},$$

[6] Note that the MPEG-2 standard only defines the decoding algorithm, while developers are free to choose the specific implementation of the encoding algorithm.

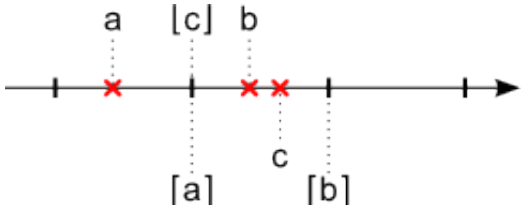


Figure 6.11: Graphical representation of (6.24).

and, taking out the leftmost factor of the central term:

$$\frac{k_2 \cdot W}{16}\left(u_2 - \frac{1}{2}\right) \leq sign\left(\left[\frac{16 \cdot x}{k_1 \cdot W}\right]\right) \cdot \left\lfloor \frac{sign\left(\left[\frac{16 \cdot x}{k_1 \cdot W}\right]\right) \cdot W \cdot \left[\frac{16 \cdot x}{k_1 \cdot W}\right] \cdot k_1}{16} \right\rfloor < \frac{k_2 \cdot W}{16}\left(u_2 + \frac{1}{2}\right).$$

Being $n(x)$ a simmetric function, we can focus the attention on the specific case:

$$sign\left(\left[\frac{16 \cdot x}{k_1 \cdot W}\right]\right) = 1,$$

without loss of generality. In this case, we have

$$\frac{k_2 \cdot W}{16}\left(u_2 - \frac{1}{2}\right) \leq \left\lfloor \frac{W \cdot \left[\frac{16 \cdot x}{k_1 \cdot W}\right] \cdot k_1}{16} \right\rfloor < \frac{k_2 \cdot W}{16}\left(u_2 + \frac{1}{2}\right).$$

Now let us make use of the following property: given $a, b \in \mathbb{R}$, we have

$$a \leq \lfloor c \rfloor < b \qquad \Longrightarrow \qquad \lceil a \rceil \leq c < \lceil b \rceil, \tag{6.24}$$

as graphically illustrated in Figure 6.11. Application of (6.24) to our case yields:

$$\left\lceil \frac{k_2 \cdot W}{16}\left(u_2 - \frac{1}{2}\right) \right\rceil \leq \frac{W \cdot \left[\frac{16 \cdot x}{k_1 \cdot W}\right] \cdot k_1}{16} < \left\lceil \frac{k_2 \cdot W}{16}\left(u_2 + \frac{1}{2}\right) \right\rceil$$

$$\frac{16}{W \cdot k_1}\left\lceil \frac{k_2 \cdot W}{16}\left(u_2 - \frac{1}{2}\right) \right\rceil \leq \left[\frac{16 \cdot x}{k_1 \cdot W}\right] < \frac{16}{W \cdot k_1}\left\lceil \frac{k_2 \cdot W}{16}\left(u_2 + \frac{1}{2}\right) \right\rceil.$$

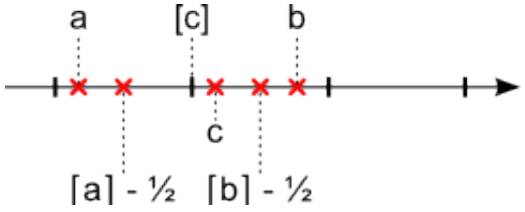


Figure 6.12: Graphical representation of (6.25).

Similarly to the previous, the following property holds for $a, b \in \mathbb{R}$ (see Figure 6.12):

$$a \leq [c] < b \qquad \Longrightarrow \qquad \lceil a \rceil - \frac{1}{2} \leq c < \lceil b \rceil - \frac{1}{2}. \tag{6.25}$$

Using this property, we obtain:

$$\left\lceil \frac{16}{W \cdot k_1} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 - \tfrac{1}{2} \right) \right\rceil \right\rceil - \tfrac{1}{2} \leq \frac{16 \cdot x}{k_1 \cdot W} < \left\lceil \frac{16}{W \cdot k_1} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 + \tfrac{1}{2} \right) \right\rceil \right\rceil - \tfrac{1}{2}$$

$$\underbrace{\frac{k_1 \cdot W}{16} \left(\left\lceil \frac{16}{W \cdot k_1} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 - \frac{1}{2} \right) \right\rceil \right\rceil - \frac{1}{2} \right)}_{L(x)} \leq x < \underbrace{\frac{k_1 \cdot W}{16} \left(\left\lceil \frac{16}{W \cdot k_1} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 + \frac{1}{2} \right) \right\rceil \right\rceil - \frac{1}{2} \right)}_{R(x)}$$

from which we can finally get:

$$\begin{aligned} n(x) = R(x) - L(x) = \frac{k_1 \cdot W}{16} \Bigg(& \left\lceil \frac{16}{k_1 \cdot W} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 + \frac{1}{2} \right) \right\rceil \right\rceil \\ & - \left\lceil \frac{16}{k_1 \cdot W} \left\lceil \frac{k_2 \cdot W}{16} \left(u_2 - \frac{1}{2} \right) \right\rceil \right\rceil \Bigg). \end{aligned} \tag{6.26}$$

In the above equation, k_1 is the only parameter that must be estimated, given that k_2 and the values of W are available from the bitstream. The multiplier k_1 is defined by its relation with the quantizer scale factor used in the first encoding Q_1 (see Section 5.1), yielding to a possible value in the set $\mathcal{K}_1 = \{2\mathrm{Q}_1 : 1 \leq \mathrm{Q}_1 \leq 31\}$.

As we said before, if we assume to have the correct k_1, the histogram of doubly quantized coefficients can be obtained from $\tilde{h}(x)$ as $n(x; k_1) \cdot \tilde{h}(x)$, and

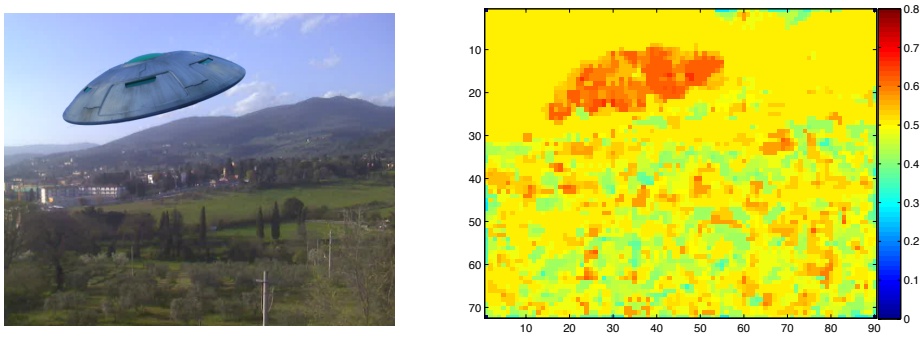


Figure 6.13: An intra-frame tampering (left figure) and the produced probability map (right figure). For showing purposes, a 3×3 median filter has been applied to the map.

we could write the probability distribution of the observed coefficients as the following mixture:

$$p(x; k_1, \alpha) = \alpha \cdot n(x; k_1) \cdot \tilde{h}(x) + (1 - \alpha) \cdot \tilde{h}(x), \tag{6.27}$$

where $\alpha \in [0, 1]$. As suggested in [35], an effective way to get an estimate of k_1 is to iteratively search the value $\hat{k}_1$ that minimizes the difference between the observed histogram $h(x)$ and $p(x; k_1, \alpha)$, choosing the optimal α in the least square sense (formula is given in [35]). With respect to the JPEG case, the minimization is simplified by the fact that the quantization matrix is known, and all the coefficients share the same k_1. Thus, we define the following vector

$$\mathbf{h} = [h_1(-\tfrac{B}{2}) \dots h_1(-1)\ h_1(1) \dots h_1(\tfrac{B}{2})\ h_2(-\tfrac{B}{2}) \dots h_C(\tfrac{B}{2})]^T$$

where $B+1$ is the number of bins of $h(x)$ and C is the number of considered coefficients. Similarly we define $\tilde{\mathbf{h}}$ and $\mathbf{n}$. Then, we can write:

$$\mathbf{p}(k_1, \alpha) = \alpha \cdot \mathbf{n}(k_1) \cdot \tilde{\mathbf{h}} + (1 - \alpha) \cdot \tilde{\mathbf{h}},$$

where " $\cdot$ " denotes component-wise vector product. Finally, $\hat{k}_1$ is obtained as

$$\hat{k}_1 = \arg \min_{k_1 \in \mathcal{K}_1} ||\mathbf{h} - \mathbf{p}(k_1, \alpha)||^2.$$

By using all the coefficients to estimate k_1, a more robust estimation is obtained; this is a crucial benefit, especially if we consider that: i) values in W

are quite high even for small i and j, ii) the spatial resolution of videos is usually lower than that of images; and both these facts reduce the number of DCT coefficients that can be exploited for the estimation. Using $\hat{k}_1$ and the $n(x)$ function defined in (6.26) for the MPEG-2 case, we can compute the probability in (6.17). Finally, the probability that an 8×8 block is tampered is obtained through equation (6.21). Figure 6.13 shows a forged frame along with the probability map generated by the proposed method.

6.3.4 Experimental validation

Experiments have been carried out on a set of videos, selected so to have heterogeneous scenes[7], cropped to a resolution of 720×576 pixels. All videos have been encoded with MPEG-2 VBR, fixed quantizer, using the `FFmpeg` coding software[8].

Experimental validation was carried out as follows (Figure 6.14): the video is first compressed with a quantizer scale factor Q_1; then it is decoded and a square block of 200×200 pixels is replaced with the same content coming from the uncompressed version of the video; finally, the resulting video is re-encoded with a factor Q_2. Using the uncompressed version of the same video as a source for tampered pixels, it is possible to create a forgery that is imperceptible to the eye, thus mimicking the work of an expert.

Given that the performance of VPF do not strongly depend on the size of GOPs, we employed fixed GOP sizes $G_1 = 12$ and $G_2 = 15$ for the first and second compression respectively. Furthermore, we limited ourselves to use P-frames, since GOP estimation in presence of B-frames is not possible with VPF. As shown in [35], forgery localization generally works when the second compression is not as strong as the first one. For this reason, we chose $Q_1 \in \{6, 8, 10, 12\}$, $Q_2 \in \{2, 3, 4, 5\}$, and all the possible combinations between these two sets are used for generating tampered videos. Finally, since the model proposed in Section 6.3.3 has been derived assuming that the fixed quantizer is uniform, the dead-zone of the quantizer implemented in `FFmpeg` was fixed to the interval $[-\Delta/2, \Delta/2]$ (where Δ denotes the quantization step),

[7] Selected videos are: *ducks_take_off*, *in_to_tree*, *old_town_cross*, *park_joy*, *shields*, *sunflower* and *touchdown_pass*, freely available at `http://media.xiph.org/video/derf/`.

[8] `http://www.ffmpeg.org/`

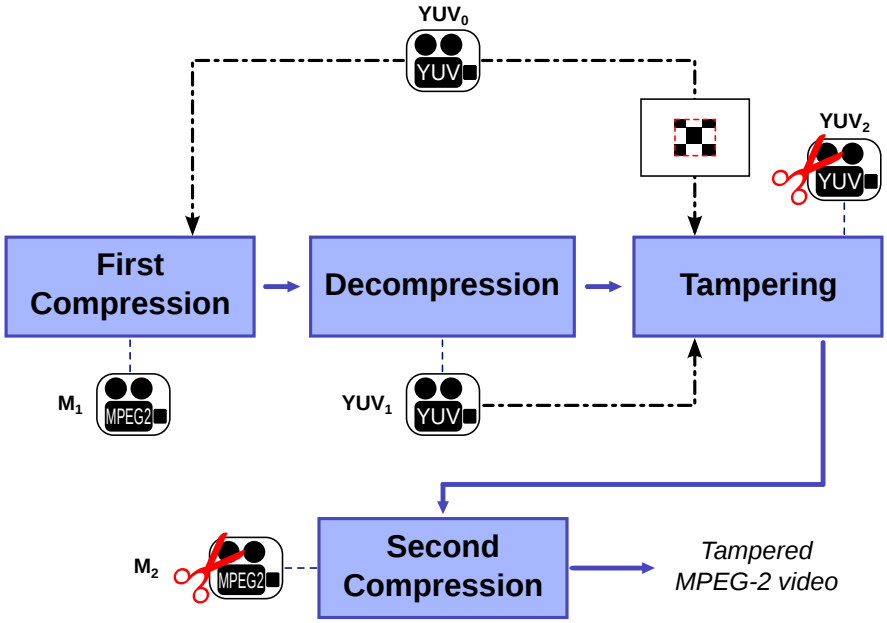


Figure 6.14: Adopted procedure for generating tampered videos.

by setting the parameter `ibias` equal to 128. Note that the model can be easily adjusted to work with different dead-zones.

First of all, we investigated the reliability of the VPF-based GOP estimation in the considered scenario since, in the case of a wrong estimation, the proposed method would fail. The GOP size was retrieved from the available set of tampered videos and the number of exact estimations of G_1 was calculated. The estimation never failed under the considered settings, thus confirming that VPF can be safely used.

After retrieving the GOP size of the first compression (denoted with $\hat{G}_1$) for each tampered video, DQ analysis was carried out, specifically in frames indexed by elements in the set $\mathcal{C}_{\hat{G}_1,15}$, defined in Section 6.3.2. Only the first 5 AC coefficients in the zig-zag ordering were used for the analysis. The probability map produced from each frame was then thresholded and compared to the ground truth mask, allowing us to calculate the true positive and false positive rate; these values were averaged over all videos sharing the

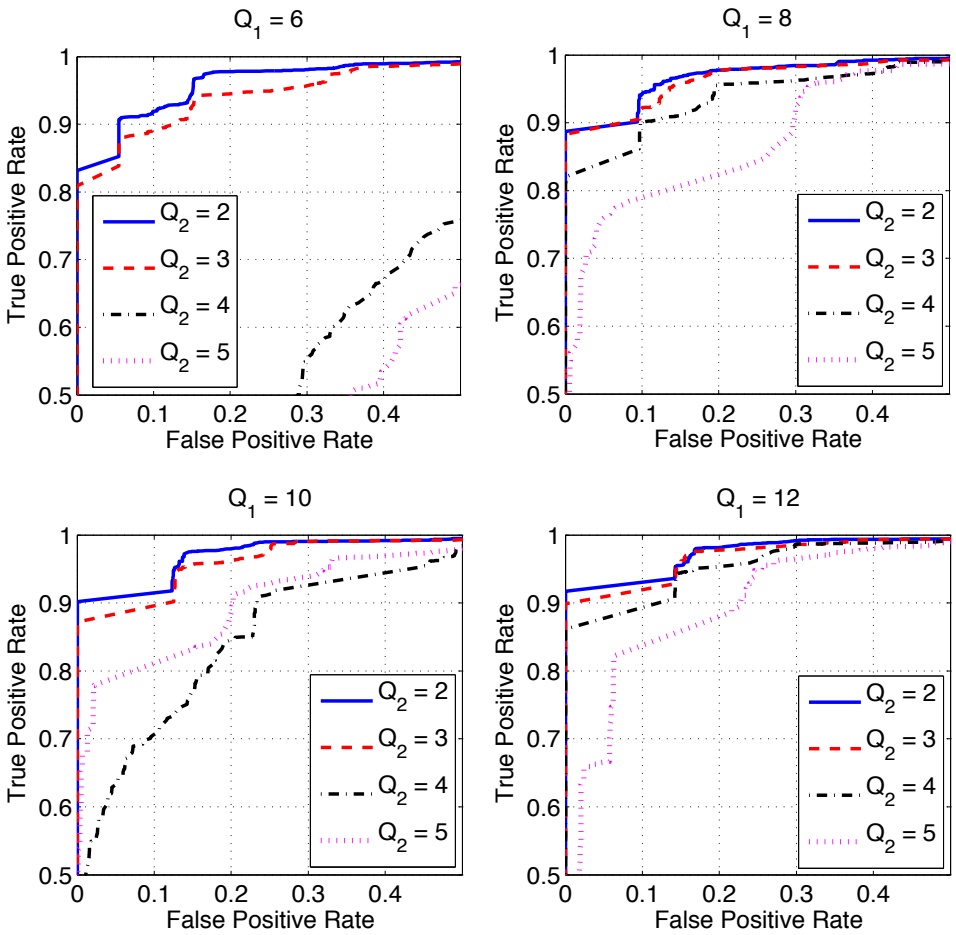


Figure 6.15: ROC curves obtained for the examined combination of Q_1 and Q_2. From top left to bottom right, increasing values of Q_1 are considered, and performance for varying values of Q_2 are plotted (notice that curves have been magnified to improve readability). As expected, lower values for the second compression facilitate the localization.

same combination of Q_1 and Q_2. By varying the threshold, for all the explored combinations of Q_1 and Q_2 ROC curves were obtained (Figure 6.15) and their AUC calculated (Table 6.6).

As the table clearly shows, the localization accuracy obeys the same rules

Q_1 \ Q_2	**2**	**3**	**4**	**5**
6	0.98	0.97	0.68	0.63
8	0.98	0.98	0.96	0.93
10	0.98	0.98	0.91	0.94
12	0.98	0.98	0.97	0.94

Table 6.6: AUC obtained with the proposed method for different combinations of the first and second quantizer scale factors.

holding for images: for a given Q_1, lower values of Q_2 facilitate the localization, because the two components of the mixture are more clearly distinguishable since the peaks in the histogram are better separated. For the same reason, if we fix the value for Q_2, higher values of Q_1 favour the performance of the method. We can conclude, then, that larger differences between Q_1 and Q_2 correspond to higher reliability of the produced map.

Moreover, although this is not visible from the reported results, we can state that the proposed method is not reliable on very uniform regions. The reason is that such regions have a very sparse representation in the DCT domain, making it difficult to tell apart double and single compressed coefficients (they will likely be zero in both cases). Such an information could be exploited to help the analyst: for example, in the output map those regions where the tool is likely to be unreliable could be marked with a specific flag.

Chapter 7

Concluding Remarks

SPLICING detection in digital videos is a very interesting and open field today, also considering the fact that video editing is really one click away: even portable devices allow basic manipulations, like cropping and trimming, that can significantly alter the meaning of the recorded content. The VPF analysis and the derived methods that we introduced in this part of the thesis are an attempt to widen the forensic analyst arsenal for video integrity and authenticity verification. We believe the main strength of the VPF relies in the simplicity of the phenomena being studied, that makes it fit most of the diffused encoding standards (with the noticeable exception of Motion-JPEG coding, where all frames are intra-coded, thus preventing the presence of the VPF). On the other hand, the tools we proposed are undoubtedly open to generalizations and improvements, which are outlined in the rest of this chapter.

7.1 Widening the generality of VPF

Probably the main limitation to the applicability of the VPF-based analysis is that several video editing tools adopt B-frames to encode the produced video, sometimes even as a default option. Throughout our discussion we never considered the impact of B-frames on the presence of the VPF but (as confirmed by some preliminary experimental investigation) such impact is definitely not negligible. This fact is not surprising, since we focus the analysis on prediction of MBs, and in B-frames such a prediction follows a more elaborated path. Therefore, while we can state that the VPF analysis is not adoptable in its current form when the second encoding features B-frames, we can argue that some traces of double encoding are still left in the way MB-s are predicted, though in a different way. This is indeed the first

extension to the VPF we are going to investigate in the future.

As a second point, we noticed that the performance of the VPF analysis are somewhat content dependent. Also in this case, there is a rather intuitive justification for this fact: MBs containing very smooth, untextured content are likely to be predicted differently than "complex" MBs. On the one hand this can be seen as a limitation of the VPF, but on the other hand it opens to the possibility of a more robust study: video frames most of the times contain both uniform and textured regions, so that we could adaptively select a subset of MBs and limit the analysis to them. Moreover, since modern camcorders capture video at very high resolutions - and considering that MBs are not bigger than 16-by-16 pixels - chances of finding MBs with uniform content increase.

Finally, we must admit that the VPF has a rather empirical foundation at the moment, and a deeper theoretical investigation about its originating factors is needed. However, providing a rigorous and general mathematical modeling for MB prediction is hard (if not impossible), because coding standards do not contain indications on how this phase should be implemented, and developers are left completely free.

7.2 Future works on inter- and intra- frame forgery detection

Since the methods we proposed in sections 6.2 and 6.3 rely strongly on the VPF, they also inherit its limitations. Similarly, they will also benefit from the possible improvements to the VPF analysis that were proposed previously in this chapter. Besides that, there are some other, more specific, possible extensions to such methods.

Concerning the proposed inter-frame forgery detection system, the localization of cutting/insertion points is not as precise as that allowed by motion-vector based schemes [62, 63], as it depends on the width of the analysis window: larger windows allow more reliable analysis, but this comes at the price of a lower localization resolution. Moreover, due to its intrinsic nature, the method cannot detect frame manipulations when the attacker removes/inserts a whole GOP, thus re-establishing the exact periodicity of the analyzed signal;

a specific analysis technique should be devised for this specific forgery setting. Finally, the method has not been evaluated in the presence of global processing of the video, e.g. resizing of frames. This is a potential threat to the reliability of the analysis: for example, resolution adjustment is not unfrequent when the video is imported and then exported using an editing software. However, since all frames would undergo the same resampling operation, there is hope that the impact of resizing on MB prediction is not strong, and the same reasoning applies to other kind of processing that are not content-dependent. We are willing to perform a deep experimental investigation to shed light on this aspect.

As to the intra-frame forgery localization method, we had to impose rather heavy requirements: first, we only considered the CBR coding mode, where a fixed quantizer is used throughout the same frame. Generalization to different quantizers is feasible at the cost of a more complex analysis: DQ analysis could be carried separately, grouping MBs based on their quantization factor; nevertheless this would lead to a lower reliability of the localization map, because model parameters would be estimated on a lower number of samples. Moreover, we assumed that no cropping occurred between the two compressions, because this would delete DQ traces: also in this case, it is possible to extend the method so that it searches for traces of non-aligned double quantization (this kind of analysis has been proposed in [41] for images). Another practical constraint regards compression quality: the video should not be re-encoded at lower quality, otherwise DQ traces would not be detected. Compared to previous ones, we believe this requirement is not very limiting: the first encoding is typically performed by the capturing device, whose storage availability is limited, thus calling for strong compression. On the contrary, tampering is usually performed on a PC, where coding time and storage capabilities are not a concern; it is reasonable to think that the user employs less aggressive compression to avoid decreasing the quality of the produced video. Finally, we recall that the only supported codec is MPEG-2, whose popularity is decreasing in favour of more recent coding algorithms such as H.264. Unfortunately, modern coding algorithms employ advanced techniques (like adaptive choice of macro-block size, intra-frame spatial prediction, etc.) that strongly complicate double quantization analysis.

7.3 Conclusions

During our work, and by interacting with people working on the field, we got the understanding that video forensics is still at an embryonal development stage. This is evident both from the theoretical point of view (the amount of mathematical models developed for digital video forensic is incomparable to that existing for images) and from the application point of view, meaning that there are few published tools for video authenticity verification, and no tools at all in the market. On the other hand, if we peek into the field of forensic image and video *analysis* (i.e., the branch of forensic science that tries to enhance the quality of multimedia contents so to improve their intelligibility), we see that videos are far more investigated than images. Thus, we can argue that digital videos are considered of great importance as digital evidences in the forensic environment. Since it is slowly but constantly becoming clear to the forensic community that the integrity and authenticity of digital evidences should be verified before trusting them, we can easily foresee a strong increase in the need for video authentication tools; hopefully, this will further motivate research in this fascinating field.

Part III

Fake Quality and Splicing Detection in MP3 Audio Tracks

Abstract

Audio recordings are one of the most important assets in the forensic environment. Be them obtained by an authorized recording of a conversation, by electronic eavesdropping or by wiretapping, gigabytes of audio recordings are captured everyday to be used in the court. Compared to the visual counterpart, a decent audio manipulation is much easier to realize: this was already true in the analog world, and has become even more true in the digital era. Thus, the combination of high forensic relevance and easiness of manipulation rapidly raised the problem of audio integrity and authenticity verification.

While up to the '90s methods were investigated to authenticate magnetic tape recordings, recently the attention moved to the digital side and nowadays there are several ways to investigate the authenticity of an audio track. However, it is surprising that digital image and audio forensics, despite being born almost simultaneously, followed two totally separated branches: techniques that worked brilliantly well for images were never applied to audio signals and viceversa. One noticeable example is the detection of cut-&-paste based on traces of double compression: while double quantization analysis has been investigated deeply for digital audio, its application to authenticity verification was never attempted.

The contribution of this part of the thesis goes in this direction, and is two-fold: first, we propose a novel approach for the detection of double MP3 encoding, with an emphasis on fake quality detection. Then, similarly to what we did for video forensics, we build on the proposed technique to provide a tool for forgery localization, still under the assumption of MP3 coding. While our studies focused only on MP3 coding, the underlying principles can be extended without problems to different lossy audio compression standards.

Chapter 8

Introduction to Audio Forensics

AUDIO forensics is the discipline studying the acquisition, the analysis and the evaluation of audio recordings that can be used in a court as bodies of evidence [7]. As such, the main goals of audio forensics in its broader definition are:

- assess the authenticity and integrity of a recording;
- process the signal in order to enhance its intelligibility, e.g., for making a recorded conversation more understandable;
- analyze the signal so to extract information from it, like the identity of speakers, the transcription of dialogues, or even information about the recording environment.

Audio forensic can be considered more mature, from an historical point of view, compared to image and video forensic. This is certainly due to the importance that this kind of examination had in some important legal cases dating back to to '60s. It was actually in the development of the United States versus McKeever case, that the judge outlined the first rules for assessing the authenticity of an audio recording. Among the list of conditions (fully explained in [7]), there are requirements about audio integrity ("that the recording has been preserved in a manner that is shown to the court") and authenticity ("that changes, additions, or deletions have not been made in the recording"). Of course, the judge did not provide *methods* for assessing whether the conditions were met.

A few years later, audio forensic played an important role in the *Watergate* scandal, when a panel of audio experts were asked to analyze the authenticity of one of the recorded Nixon's conversations. After analyzing the magnetic

tape, the panel concluded that there were traces of severe manipulations, namely erasures.

Although the mentioned historical cases date back to the analog era, many lessons learned at that time are still valid. It is easy to guess that authenticity verification has become even more difficult with digital audio recordings, since "physical" traces related to magnetic tapes are no longer of help. There are mainly two kinds of manipulations that can mine the authenticity of an audio recording:

- *cut-type*, were part of the recording is erased or replaced;
- *signal processing*, where non-linear effects are applied to the signal or the signal is mixed with extraneous content.

In both cases, it is fundamental to assess whether the track was compressed after editing, because the reliability of some of the most effective techniques for authenticity verification is undermined by lossy coding.

8.1 Previous works in audio forensics

Although audio forensics pursue a wide range of goals, in the scope of this thesis we are mainly interested in those techniques related to authenticity and integrity verification. For this reason, in the following we will only mention the most relevant works targeting this application, with a particular emphasis on those based on the analysis of quantization artifacts.

8.1.1 Techniques based on the Electric Network Frequency

One of the most popular and effective branch of techniques in audio forensics is related to the analysis of Electric Network Frequency (ENF) [66]. ENF is the characteristic frequency of a networked electricity supply transmission system, whose typical nominal values are 50Hz or 60Hz. When a digital recorder is used to capture an audio track, the power line related signal will be "embedded" within the registration. Interestingly, this holds not only for recorders connected to the power supply, but also for battery-powered device as long as they are used in the proximity of electro-magnetic fields emanating from the network.

While electric companies do their best to keep the ENF as close al possible to its nominal value, small fluctuations (in the order of 10^{-2} Hz) are unavoidable. An important property of such fluctuations is that they are consistent throughout the entire network: experimental evidence confirmed that, at a given time, the actual ENF has the same value across Europe (apart from UK, which is not connected), and the same rule holds in different parts of the world. Building on this fact, Grigoras [67] showed that it is possible to date an audio recording, by comparing the observed ENF within the track with a reference database. Moreover, by looking for partial matching and inconsistencies, also authenticity verification can be performed [67]. The main drawback of the above technique is that a reference database is needed, together with a reliable engine for searching possible matches. Recently, a different approach to authenticity verification based on ENF has been proposed by Rodríguez et al. [68], whose idea is to exploit the bare presence of the ENF (not its fluctuations). The audio signal is analyzed with a sliding window, estimating the phase of the ENF-related signal at each time: if the track is manipulated, e.g., by a cut-&-paste attack, the phase of the ENF related sinusoid should show a tell-tale discontinuity. The main obstacles to the use of ENF related techniques probably resides in the fact that many recording devices feature a band-pass filter that removes low frequencies, including 50Hz. In such cases it may still be possible to analyze traces of the ENF at harmonic frequencies, but the analysis becomes much less accurate.

8.1.2 Techniques based on quantization analysis

The audio forensic community also worked significantly on the analysis of the effects of lossy compression, with an emphasis on fake-quality detection, that is when an audio file is recompressed at higher bit-rate to pass off it as a high-quality track. To defeat Fake-Quality MP3, Yang et al. [69] observed that there are many more quantized Modified-DCT (MDCT) coefficients with small values in a single compressed MP3 file than in a fake-quality MP3 file, no matter which bit rate the fake quality MP3 was transcoded from. According to this, a detector was proposed that just measures the number of MDCT coefficients assuming ± 1 values and compares this value to a given threshold T: if it is lower than T, the file is a fake-quality one, otherwise it is a single

compressed one. The same authors in [70] proposed to detect double MP3 compression through the use of support vector machine classifiers with feature vectors formed by the distributions of the first digits of quantized MDCT coefficients; in particular, a global method was proposed, where the statistics on the first significant digit of all quantized MDCT coefficients are taken, and the probability distributions of nine digits are used as features (9 dimensions) for training a SVM. A so called band distribution method is also proposed, where a procedure of band division is added before computing the statistics on the first digits, allowing to increase the performance. To detect double MP3 compression, Liu et al. [71, 72] proposed to extract some features of the MDCT coefficients and use them to train a SVM; more specifically, a set of statistical features of zero MDCT coefficients and non-zero MDCT coefficients from the frequency range as well as individual scale bands are exploited.

As to forgery detection, Yang et al. [73, 74] developed a method for MP3 audio files: based on the observation that tampering breaks the original frame segmentation, frame offsets are used to locate forgeries automatically, allowing to detect common forgeries such as deletion, insertion, substitution, and splicing. However, experimental results are carried out on audio files that have not been re-encoded in MP3 format after the manipulation. Böhme et al. presented in [75] a method to determine the encoder of MP3 data on the basis of statistical features extracted from the data; the work also addresses the classification of MP3 files re-compressed with different encoders, but considering the same value of bit rate for the second compression. Finally, Moehrs et al. [76] considered the inverse decoder problem, where only the uncompressed samples are known to the analyst and the goal is to recover the parameters of a possible previous compression. The same problem is tackled with in a work by D'Alessandro et al. [77], where some properties of the frequency spectrum of the song under analysis are exploited; the same classifier used to differentiate between different MP3 quality levels was also applied to detect transcoded MP3s, but for this scenario the presented experiments are rather limited.

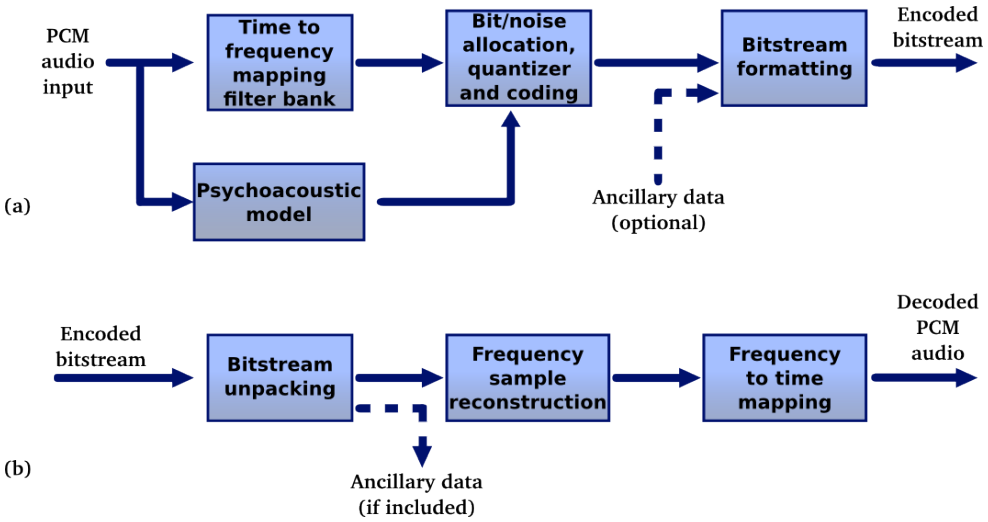


Figure 8.1: High level block diagram of MP3 coding (a) and decoding (b).

8.2 Basics of MP3 audio coding

Audio coding is a very broad field, that of course cannot be reviewed in this thesis. Here we limit ourself to just one coding algorithm, namely MPEG-1 Layer III (abbreviated with MP3 from now on), since it is the one that is treated in our work. Also within this coding scheme, we discuss only those characteristics that are strictly necessary to the understanding of our work, omitting many details.

The basic idea underlying MP3 coding is the same of all perceptive coding schemes: those parts of the signal that are less audible by the human ear can be discarded without a significant perceptual loss. Audio is processed by splitting it into *frames* of 1152 samples per channel. Since the human hearing apparatus has different sensitivity to different frequencies, the first operation that is carried during MP3 coding is the conversion to spectral components, using an analysis filterbank, that generates 32 separate sub-bands (leftmost side of Figure 8.1), each one to be treated separately.

For each band, an adaptive Modified Discrete Cosine Transform (MDCT) is computed. This transform is specifically thought to be computed on adjacent blocks of a signal, where the last half of a block coincides with the

first half of the following block. The size of blocks can vary between short blocks (containing 6 spectral points) and long blocks (18 points); the choice is regulated by a psychoacoustic model. For a long block a total of 576 (32×18) spectral coefficients are generated (192 for short blocks) and this forms a *granule*; two granules together form one MP3 frame.

The following step, that is crucial for the forensic analysis we are going to undertake, is quantization of the spectral coefficients of each band: this is a critical step, since the quantization noise must be kept below the so-called *masking threshold* to guarantee a negligible perceptual impact. To this aim, this step is carried differently for each sub-band and, within each sub-band, for different MDCT coefficients. The impact of quantization is evaluated using a core element of MP3 coding, that is the psychoacoustic model. This model is used to adaptively determine the number of code bits to be allocated to each sub-band, by iteratively searching an acceptable balance between distortion and quantization. The quantization is non-uniform, since smaller steps are used for low valued coefficients. After quantization, the 576 MDCT coefficients are Huffman coded and organized in the output bitstream.

Chapter 9

Double Encoding Detection and Forgery Localization for MP3 Tracks

THIS chapter introduces a novel audio forensic tool for MP3 compressed audio files, based on analyzing the effects of double compression in the statistical properties of quantized MDCT coefficients. The method relies on a single measure derived from the statistics of MDCT coefficients, allowing to apply a simple threshold detector to decide whether a given MP3 file is single compressed or it has been compressed twice, without resorting to SVM classifiers. Moreover, the proposed method is able to derive the bit-rate of the first compression by means of a Nearest Neighbour classifier. The ability of the algorithm to detect double MP3 compressed files remains valid also on tracks of reduced length, allowing its application to the localization of single/double compressed segments within an MP3 audio file. By building on this feature, we develop an authenticity verification method that, under some assumptions, is able to localize spliced regions within MP3 tracks.

The chapter is structured in two main parts: first, the method is presented (Section 9.1) and its application to fake quality detection are discussed, then the extension to forgery localization is explained (Section 9.2). Experimental validation, including comparison with two state of the art methods [70, 72], is the object of Chapter 10.

9.1 Detection and classification of double compression

The core idea of the algorithm is to measure the similarity between the histogram of quantized MDCT coefficients of the MP3 file under analysis, that has possibly undergone a double compression, and the histogram of the MDCT

coefficients computed on a single compressed version of the same file. Intuitively, if the distance between the two distributions is low, we deduce that the file under analysis has not been MP3 encoded twice, viceversa the file will be considered as double compressed.

Obtaining a reliable estimate of the distribution of the single quantized MDCT coefficients from the corresponding quantized or double quantized coefficients appears to be a difficult task. However, it has already been observed in the image forensic literature [78, 8] that the DCT coefficients obtained by applying a slight shift to the grid used for computing the block DCT usually do not exhibit quantization artifacts (calibration technique [8]). In a similar way the distribution of the single compressed MDCT coefficients can be approximated by generating a simulated single compressed signal (referred to as *calibrated* signal from now on), starting from the file under analysis. This can be achieved by decompressing the MP3 file, removing a given number of Pulse Code Modulation (PCM) samples of the decompressed audio and then recompressing the remaining samples to the same compression quality of the signal under analysis.

To demonstrate that the idea is effective, an uncompressed audio track, 4 sec long, has been MP3 compressed at several bit-rates in the set $\{64, 96, 128, 192\}$ kbit/s, and then recompressed to 160 kbit/s, in such a way to obtain 4 double compressed versions, 3 with increasing bit-rate, and one with decreasing bit-rate. Then, the uncompressed file was also compressed once at 160 kbit/s. The above procedure has been applied to each of these MP3 files to obtain a calibrated signal, by removing the first 10 PCM samples to the decompressed signal and recompressing the remaining samples to 160 kbit/s. Then, the histograms of MDCT coefficients extracted from the input original files and from the corresponding simulated single compressed files have been compared.

In Figure 9.1, the histograms corresponding to the MP3 file double compressed at 64 kbit/s and then at 160 kbit/s, and to the calibrated MP3 file compressed once at 160 kbit/s are shown. It is evident that the first histogram exhibits the characteristic pattern of a distribution of coefficients that have undergone a double compression, whereas in the second one these artifacts have been removed, and thus the difference between the two histograms is significant. Similar results are obtained when the first compression was car-

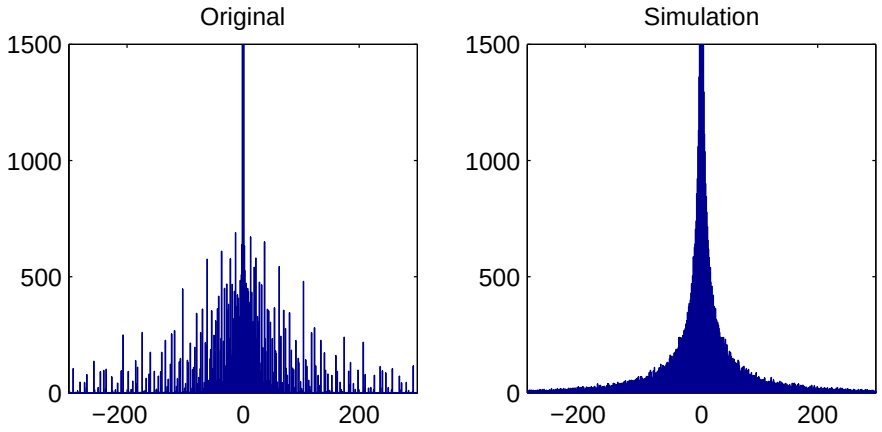


Figure 9.1: Histograms of MDCT coefficients of an MP3 file double compressed at 64 kbit/s and then 160 kbit/s (left), and of the corresponding calibrated signal at 160 kbit/s (right).

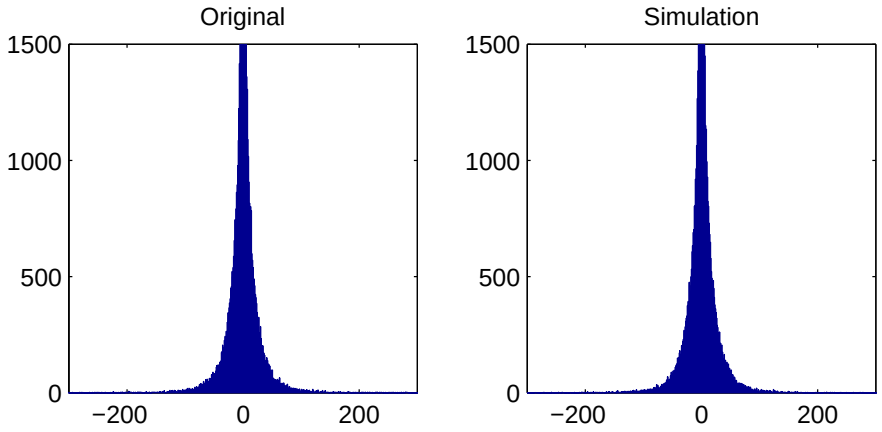


Figure 9.2: Histograms of MDCT coefficients of an MP3 file single compressed at 160 kb/s (left), and of the calibrated signal at 160 kbit/s (right).

ried at higher bit-rates, but still lower than 160 kbit/s, even if the effect of the double quantization becomes smaller.

On the contrary, if the same procedure is applied to a single compressed MP3 file, the histograms of the input file and the corresponding calibrated

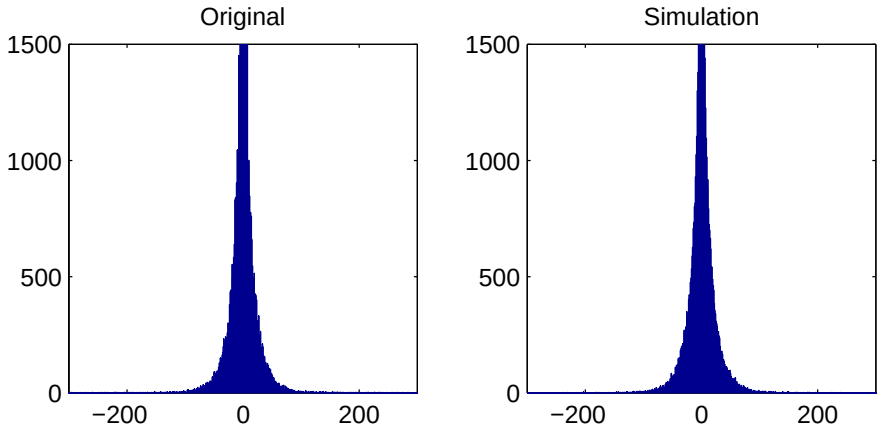


Figure 9.3: Histograms of MDCT coefficients of an MP3 file double compressed at 192 kb/s and then 160 kb/s (left), and of the calibrated signal at 160 kbit/s (right).

one are very similar, as it is shown in Figure 9.2.

A similar situation shows up when a double compression has been applied, but with the first bit-rate (i.e., 192 kbit/s) equal or higher than the second one (i.e., 160 kbit/s), see Figure 9.3: in this case, the histogram of the MP3 file under analysis does not exhibit the double compression artifacts, and the histograms of the two signals are very similar, thus not allowing to detect the double compression. The removal of these artifacts is due to the fact that the second compression is so strong that deletes the traces left by the previous one.

According to the previous analysis, we derived an algorithm composed by the processing blocks illustrated in Figure 9.4: we decompress the MP3 file obtaining a sequence of PCM samples; next, the *Trimming* removes a given number of PCM samples starting from the beginning of the PCM sequence, in such a way that the trimmed sequence is no more aligned with the MP3 frame borders. The *Filterbank + MDCT* block filters the PCM samples and transforms them, achieving a set of unquantized MDCT coefficients. Trimming and recompression allow to remove possible double quantization artifacts, while maintaining the original characteristics of the signal. The *Parameter extraction* block allows to extract from the original MP3 bitstream

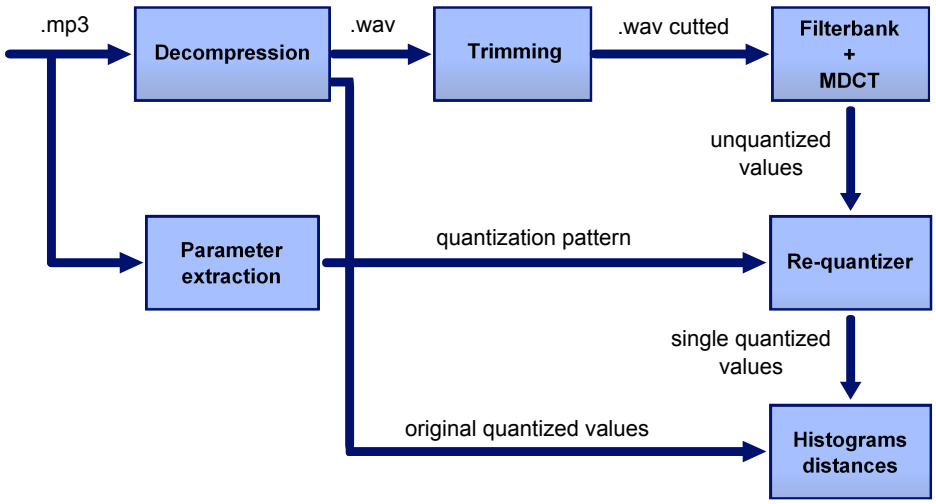


Figure 9.4: Scheme of the proposed method.

the quantization parameters, i.e., the quantization pattern and the original quantization values. The quantization pattern is needed by the *Re-quantizer* to calibrate the signal. In addition, the *Re-quantizer* smooths the sequence of simulated coefficients through a *Laplace Smoothing* [79], a technique used to smooth discrete data (in particular, we adopted the smoothing parameter α equal to 1). This operation aims at filling possible empty bins present in the histogram, and avoiding numerical errors due denominators close to zero in the following computations. The original quantized values and the calibrated values are then compared through the *Histogram distances* block.

By indicating with X and Y, respectively, the histograms of the observed and calibrated MDCT coefficients, we compute a similarity measure. Among the possible measures that can be used to compute the distance between two histograms, we adopted the Chi-square distance $D_\chi(X, Y)$ [80], defined as:

$$D_\chi(X, Y) = \sum_{i=1}^{N} \frac{(X_i - Y_i)^2}{2(X_i + Y_i)} \tag{9.1}$$

where N is the number of bins of the histograms.

The computed distance measure, that will be denoted by D_χ from now on, is the basis of the method that we propose to detect whether an MP3 file

has been single or double compressed.

As it will be shown in Chapter 10, by analyzing D_χ it is possible to retrieve additional information about the compression undergone by the MP3 file, in particular concerning the difference between the second and the first bit-rate ($\Delta = BR2 - BR1$): when such a value is positive, we can classify double compressed MP3 audio tracks according to the first bit-rate. In fact, the values assumed by D_χ range quite differently according to the second bit-rate and Δ, thus allowing to cluster double compressed MP3 audio tracks by applying a Nearest Neighbour classifier.

9.2 Application to forgery localization

Knowledge about double encoding of an MP3 track can also provide evidence of local tampering. Let us assume that a part of an MP3 track has been edited to alter the original audio recording. It is reasonable to assume that the editing operation will also remove the features due to MP3 compression. If such a track is recompressed in MP3 format, then the original part will exhibit the typical artifacts of double compression, whereas the edited part will be very similar to a single compressed track. A similar behavior can be expected when a portion of an audio track is deleted: with a very high probability, such a deletion will introduce a desyncronization of the audio frames; hence, in case of recompression, the audio frames up to the deletion point will be double compressed, whereas the audio frames after the deletion point will appear as single compressed.

In order to exploit the proposed feature D_χ for tampering localization, a straightforward strategy is to divide the analyzed audio track into several short segments and to evaluate D_χ for each segment. In the presence of a genuine audio track, we expect to have similar values on all audio segments. Conversely, in the presence of tampering, the segments corresponding to the manipulated part will behave differently than the original part. In particular, we will assume that a tampered part is characterized by a lower value of D_χ , corresponding to single compressed audio tracks. An example of this behavior is visible in Figure 9.5, where we show the values of the proposed feature for a genuine audio track and an audio track with a local tampering, analyzed by

using audio segments 1/8 seconds long.

In the following, we propose a simple procedure to localize the manipulated parts in MP3 tracks. The first step is to divide the track into several segments and to compute the value of D_χ for each segment. The set of values obtained in such a way are then clustered using the expectation maximization (EM) algorithm [81]. Namely, we assume that the distribution of D_χ can be modeled as a mixture of two approximately Gaussian components, corresponding to the original part and the tampered part. The EM algorithm will produce two clusters characterized by the respective cluster centers, D_1 and D_2, corresponding to the mean value of the Gaussian component representing each cluster.

Let us assume, without loss of generality, that the cluster centers satisfy $D_2 > D_1$. According to the previous analysis, a given audio track will be classified as tampered if the EM algorithm finds two non-empty clusters. Furthermore, when a track is classified as tampered, the audio segments belonging to the cluster with the lower mean, i.e., D_1, will reveal the position of the tampered part. The rationale of the proposed approach is evident by looking at Figure 9.6, showing the distributions of D_χ for a tampered audio track and an original audio track. In the case of a genuine MP3 track, all the values are similar, and the EM algorithm will find a single cluster. Conversely, in the case of a tampered MP3 track, the values form two distinct clusters, corresponding to the original part and the manipulated part, which are well separated by the EM algorithm.

It should be noted that such an approach to turn a double encoding detection method into a forgery localization method is not necessarily restricted to the technique described in Section 9.1. As we will show in Chapter 10, any double encoding detection scheme can be employed, provided that it works on small sized windows.

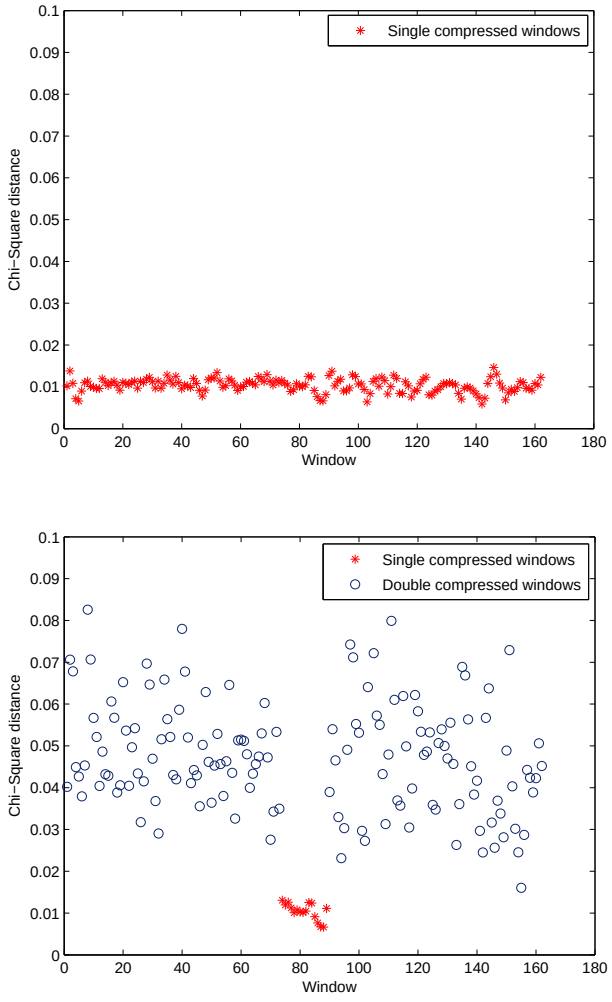


Figure 9.5: Clustering of D_χ values over consecutive temporal windows: (a) genuine audio track, where all segments are single compressed; (b) tampered audio track: the central part of the MP3 track has been edited (mixed with extraneous content) and recompressed; since the editing operation removes traces of prior quantization, the original segments exhibit the artifacts of double compression, whereas the edited ones are similar to a single compressed track.

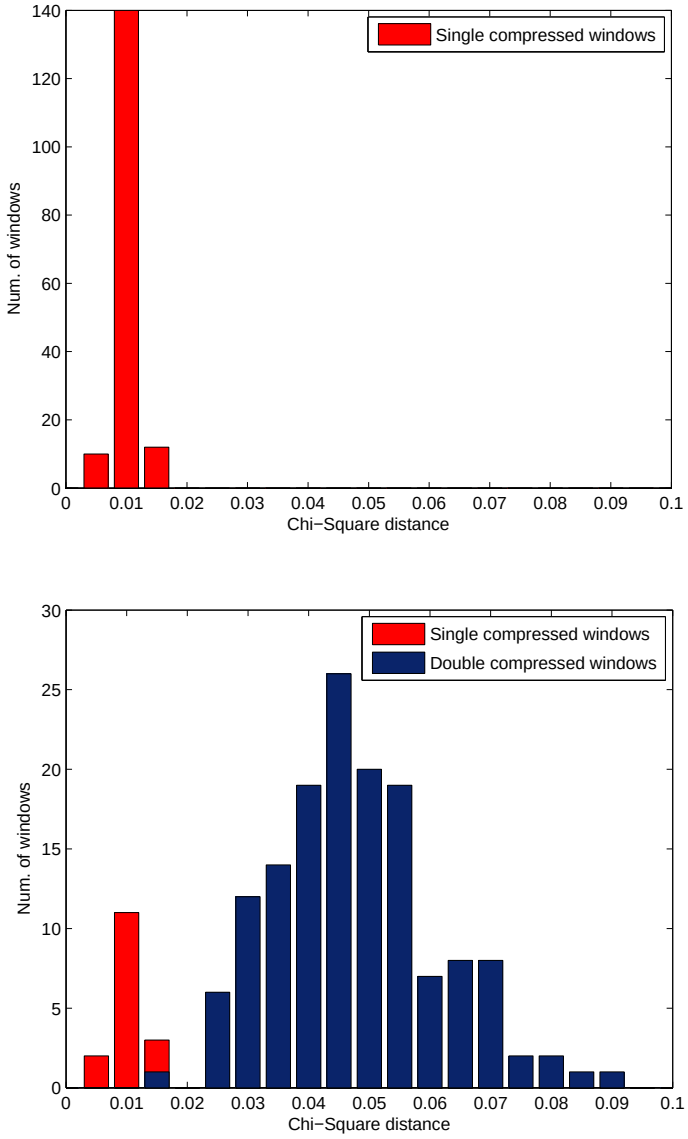


Figure 9.6: Histogram of D_χ values on two analyzed audio tracks: (a) genuine audio track; (b) tampered audio track. The different colors highlight the two Gaussian components found by the EM algorithm.

Chapter 10

Experimental Validation

THIS chapter provides a thorough experimental validation of our method for the forensic analysis of MP3 tracks. To this end, we generated a dataset, as described in Section 10.1, and we compared the performance of our approach to those of two state of the art systems proposed in [71, 72] by Liu, Sung, Qiao and in [70] by Yang, Shi, Huang.

For sake of clarity, we present separately the experimental validation of double compression detection with bit-rate estimation (Section 10.2) and forgery localization 10.3). Finally, we draw some conclusions in Section 10.4, also outlining possible future developments.

10.1 Dataset for the experiments

In order to test our calibration-based method on an heterogeneous dataset, we collected uncompressed audio files coming from different sources. More specifically, we included four different categories of recordings:

- *Music*: royalty free music audio tracks, with 5 different musical styles (jazz, latin, africa, country, funk);
- *Speech*: audio files containing dialogues;
- *Outdoor*: audio files relative to outdoor recordings;
- *Commercial*: files containing dialogues combined with music, as often happens in advertising;

Each category collects about 17 minutes of audio. Throughout all the experiments, we employed the latest release of the `lame` codec (available at

http://lame.sourceforge.net), namely version 3.99.5. This choice was motivated by the widespread adoption of this codec and by the fact that it is an open source project.

10.2 Double compression detection and first compression bit-rate estimation

In order to evaluate the performance of the system we split the available recordings into segments 4 seconds long. Since about 17 minutes of uncompressed audio were available for each category, we obtained 250 uncompressed audio files per category, for a grand-total of 1,000 uncompressed audio files. Each file was compressed, in dual mono, with bit-rate BR1 chosen in $\{64, 96, 128, 160, 192\}$ kbit/s, obtaining $5,000$ single compressed MP3 files. Finally, the files were compressed again using as BR2 one of the previous bit-rates (also the value used in the first compression was considered) obtaining 25,000 MP3 double compressed files. Among these, $10,000$ files have a difference Δ=BR2-BR1 between the second and the first bit-rate which is positive, taking value in $\{32, 64, 96, 128\}$ kbit/s; $10,000$ files have a negative difference Δ taking value in $\{-128, -96, -64, -32\}$ kbit/s and $5,000$ files have $\Delta = 0$. The overall dataset is thus composed by 30,000 MP3 files, including $5,000$ single compressed files.

As a first experiment, we computed the values assumed by D_χ for all the $30,000$ files belonging to the test dataset. The Chi-square distance was calculated by evaluating the MDCT histograms on 2,000 bins, with step size equal to one. In Figure 10.1, the Chi-square distances are visualized: single compressed files and double compressed files with negative or null Δ show a distance D near to zero, whereas the other files have D rather higher than zero. By comparing D with a threshold τ is possible to discriminate the two kinds of signals: double compressed signals with $\Delta > 0$ and the other ones. By adopting a variable threshold τ, we then computed a ROC curve, representing the ability of the detector to separate single compressed from double compressed MP3 files (including $\Delta >$ or ≤ 0). The ROC curve is shown in Figure 10.2(left): it reflects the bimodal distribution of distances of double compressed files (blue and green colored in Figure 10.1) and highlights

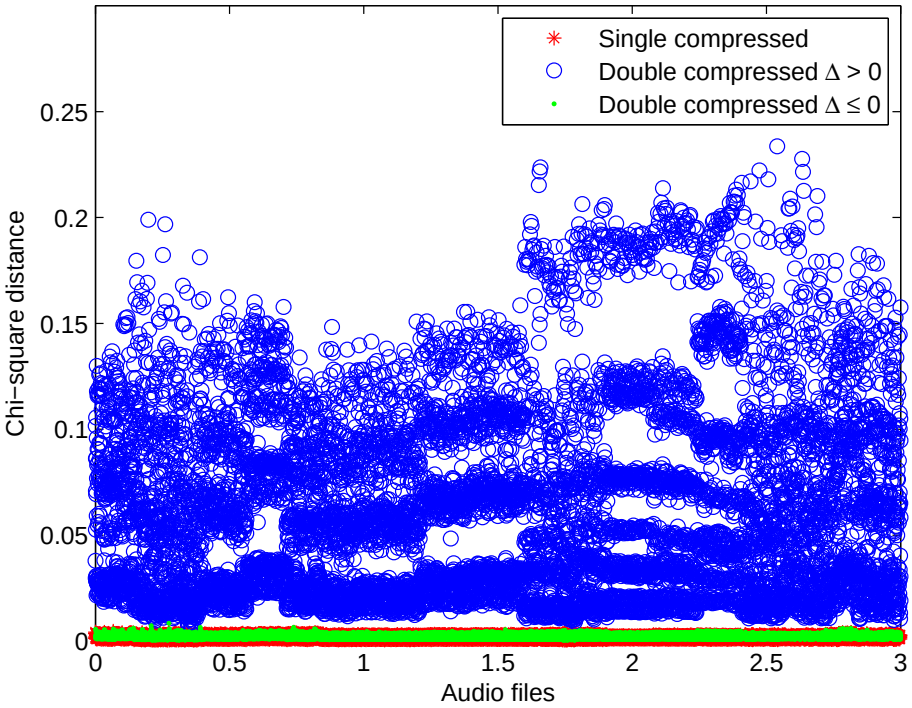


Figure 10.1: Chi-square distances computed for the 30.000 audio segments composing the dataset.

that the detector is able to distinguish only one of the two components (the one with $\Delta > 0$). Let us now separate the ROC curve in a curve corresponding to double compressed signals with positive Δ, and one curve for signals with negative or zero Δ, as shown in Figure 10.2 (right). We can see that when double compressed signals with positive Δ are considered we obtain an almost perfect classifier, while when negative or zero Δ is analyzed, we are close to a random classifier.

As anticipated in Section 9.1, D_χ assumes a large range of values, as it can be clearly observed in Figure 10.1. In order to highlight the relationship between these values and BR2 and Δ, we examined in more detail the 10,000 double compressed signals with positive Δ, plotting their Chi-square distances in Figure 10.3. Such values were plotted according to the different Δ: in

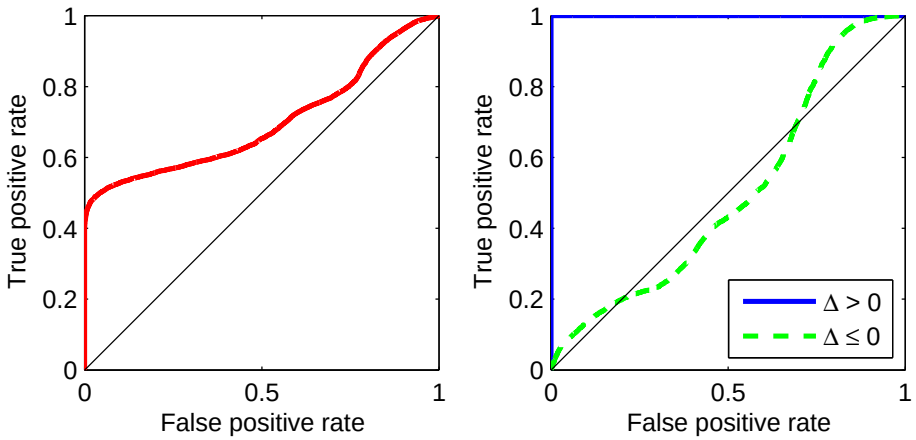


Figure 10.2: ROC curve obtained by varying the threshold τ of the classifier (left). Separation of cases with positive Δ and negative or zero Δ (right)

particular, there are 4,000 files with $\Delta = 32$ (yellow), 3,000 files with $\Delta = 64$ (violet), 2,000 files with $\Delta = 96$ (sky-blue), and finally 1,000 files with $\Delta = 128$ (black), and grouped for different BR2: $\{96, 128, 160, 192\}$. Figure 10.3 shows that the values of Chi-square distance tend to cluster for different Δ factors and different BR2 values (see plotting with same color). On the one hand, increasing Δ values give increasing Chi-square distances. On the other hand, given a value of Δ, for different bit-rate of the second compression we obtain slightly different values of distance. This suggests the possibility of optimizing the detector by considering a specific threshold for each bit-rate of the second compression (a parameter observable from the bitstream). Moreover, the different values of the feature for different Δ can be used to classify the bitrate of the first compression, as detailed in the following.

As to detection, we performed a set of experiments in order to compare the detection accuracy of the proposed scheme with respect to the accuracy of the methods proposed in [71, 72] by Liu, Sung, Qiao (LSQ method) and in [70] by Yang, Shi, Huang (YSH method). The results are shown in Tables 10.1, 10.2 and 10.3 respectively. The classifiers have been trained on 80% of the dataset and tested on the remaining 20%. Results have been averaged over 20 independent trials. In our case, different thresholds have been employed

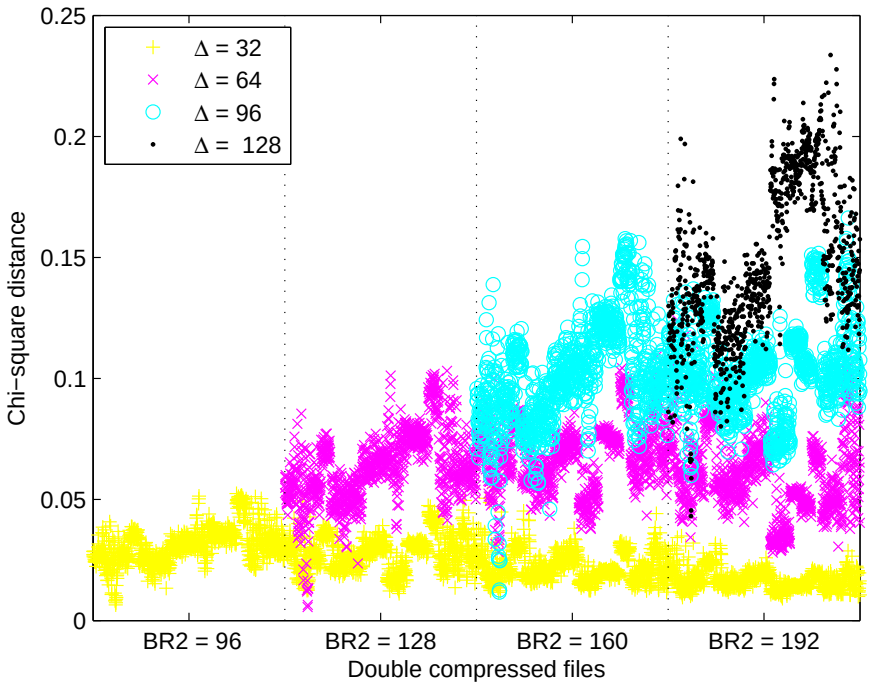


Figure 10.3: Chi-square distances computed for double compressed files with positive Δ, grouped according to both the values assumed by Δ and BR2.

Table 10.1: Detection accuracy of the proposed method for different bit-rates.

BR1 vs BR2	64	96	128	160	192
64	-	99.9	99.9	99.9	99.9
96	49.9	-	99.9	99.9	99.9
128	49.9	49.9	-	99.9	99.9
160	69.5	47.9	49.1	-	96.7
192	56.1	66.4	57.8	67.4	-

for different BR2 values, while for the other methods, different support vector machines (SVMs) have been trained for different BR2 values. Our system achieves nearly optimal performances for all combinations in which $\Delta > 0$. Also the other methods generally achieve good performances for $\Delta > 0$, especially the LSQ method. Conversely, for $\Delta < 0$ the proposed method is not

Table 10.2: Detection accuracy of LSQ method for different bit-rates.

BR1 vs BR2	64	96	128	160	192
64	-	100.0	99.9	99.9	100.0
96	97.7	-	99.5	99.7	100.0
128	96.3	98.4	-	98.1	100.0
160	88.8	98.4	98.4	-	99.9
192	72.9	96.0	95.9	94.7	-

Table 10.3: Detection accuracy of YSH method for different bit-rates.

BR1 vs BR2	64	96	128	160	192
64	-	99.9	99.9	99.9	99.3
96	96.2	-	99.7	99.7	98.8
128	80.2	99.0	-	99.2	98.1
160	84.5	94.1	96.3	-	98.0
192	67.6	88.5	89.9	90.3	-

able to reliably detect double MP3 compression, whereas the other methods have better performances.

All the results shown in Tables 10.1, 10.2 and 10.3, have been achieved considering audio tracks 4 s long. We evaluated the degradation of the performance of the three detectors when the duration of the audio segments is reduced from 4 s to [2, 1, 1/2, 1/4, 1/8, 1/16] s. The reason behind this experiment is that analyzing very small portions of audio potentially opens the door to fine-resolution splicing localization (i.e., detect if part of an audio file has been tampered). Instead of taking all the MDCT coefficients of the 4 seconds long segment, only the coefficients belonging to a subpart of the segment are retained, where the subpart is just one half, one fourth and so on. For BR2=192 kbit/s and BR2=128 kbit/s the detection accuracies (averaged with respect to BR1) have been plotted with varying audio file duration in Figure 10.4 for our method, the LSQ method and the YSH method.

The proposed system and LSQ achieve a nearly constant detection performance up to 1/8 s audio segments, whereas the performance of the YSH method drops for audio segments under 2 s. We achieve very good perfor-

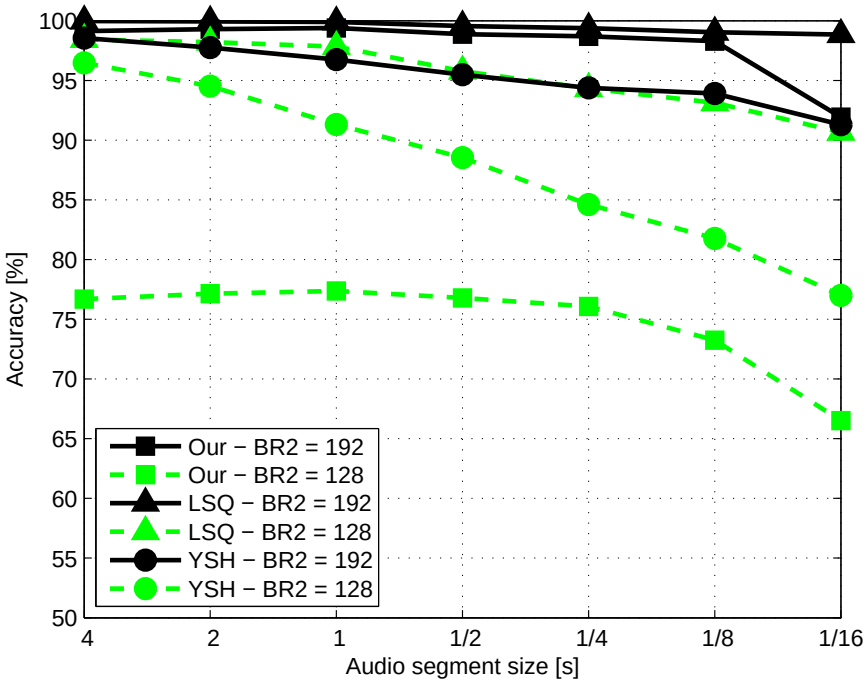


Figure 10.4: Detection accuracy with varying audio file duration for BR2=192 kbit/s and BR2=128 kbit/s.

mance in the case of high quality MP3 files: indeed, for BR2 equal to 192 kbit/s, we have almost perfect classification performance irrespective of the audio segment duration. Conversely, for BR2 equal to 128 kbit/s, even if the performance is only slightly affected by the segment duration, our method remains inferior with respect to the other two.

By taking into account the feature distribution highlighted in Figure 10.3, we then considered the capability of the proposed feature to classify the double compressed signal according to the first compression bit-rate, as BR1=BR2-Δ. A Nearest Neighbour classifier has been adopted for each different BR2 and the corresponding classification accuracy results are shown in Table 10.4 for BR2=192 kbit/s and 10.6 for BR2=128 kbit/s. The rows of the tables represent the actual bit-rate of the first compression, and the columns the values assigned by the classifier. A comment about this experiment is in

Table 10.4: Classification accuracy of the proposed method for BR2 = 192.

Actual vs Pred.	single	192	160	128	96	64
single	81.7	18.3	0.0	0.0	0.0	0.0
192	23.0	77.0	0.0	0.0	0.0	0.0
160	0.0	0.6	99.4	0.0	0.0	0.0
128	0.0	0.0	7.9	81.7	10.4	0.0
96	0.0	0.0	0.0	11.1	77.6	11.3
64	0.0	0.0	0.0	1.0	24.1	74.9

Table 10.5: Classification accuracy of LSQ method for BR2 = 192.

Actual vs Pred.	single	192	160	128	96	64
single	88.0	12.0	0.0	0.0	0.0	0.0
192	12.8	87.2	0.0	0.0	0.0	0.0
160	0.0	0.0	100.0	0.0	0.0	0.0
128	0.0	0.0	0.1	99.9	0.0	0.0
96	0.0	0.0	0.0	0.0	100.0	0.0
64	0.0	0.0	0.0	0.1	0.3	99.6

order: as shown in the previous section, the proposed method will hardly detect double compression for negative or null Δ. Similarly, the output of the classifier on double encoded signals with negative or null Δ is not reliable. In particular, since a single compressed signal can be considered as a signal compressed at infinite quality, the classifier cannot distinguish between single encoded signals and double encoded signals with negative Δ.

For comparison, only the LSQ method is considered, since the YSH method was not proposed for compression classification. The corresponding results obtained on 4 s audio segments for BR2= 192, 128 kbit/s are shown in Tables 10.5 and 10.7 respectively.

As in the case of detection, we evaluated the dependence of classification on audio file duration. The average classification accuracy for different BR2 (i.e. {192, 160, 128, 96, 64} kbit/s) and decreasing audio file duration (i.e. {4, 2, 1, 1/2, 1/4, 1/8, 1/16} s) are shown in Figure 10.5 for both our method (a) and LSQ (b). The accuracy is averaged over every possible BR1 in the

Table 10.6: Classification accuracy of the proposed method for BR2 = 128.

Actual vs Pred.	single	192	160	128	96	64
single	34.0	36.9	24.6	4.5	0.0	0.0
192	29.6	44.1	19.6	6.7	0.0	0.0
160	12.0	1.8	86.2	0.0	0.0	0.0
128	0.3	15.7	0.2	83.8	0.0	0.0
96	0.0	0.0	0.0	3.1	96.6	0.4
64	0.0	0.0	0.0	0.6	7.6	91.7

Table 10.7: Classification accuracy of LSQ method for BR2 = 128.

Actual vs Pred.	single	192	160	128	96	64
single	89.0	5.7	0.8	4.5	0.0	0.0
192	6.1	81.5	10.6	1.8	0.0	0.0
160	1.1	11.0	87.6	0.2	0.0	0.0
128	4.7	2.0	0.1	93.3	0.0	0.0
96	0.0	0.0	0.0	0.1	99.9	0.1
64	0.0	0.0	0.0	0.0	0.0	100.0

dataset, thus providing a fair overall index of classification accuracy, that can be used to compare different methods and show a clear performance trend for different audio segment lengths.

In this scenario, LSQ appears to have a better classification performance, even if our system performs reasonably well for higher bit-rates. It is also worth noting that the performances of both methods suffer only a slight degradation for the shorter audio segments, which suggests good localization capabilities.

10.3 Tampering localization

As explained in Section 9.2, our method for double compression analysis can be used as a tool for audio forgery localization. Although this possibility was not considered in the corresponding papers, previous experiments seem to prove that also the YSH and LSQ methods can be used for such a task, by

analyzing the track using small windows. We thus designed a set of experiments to test the applicability of each of the considered algorithms to forgery localization. Starting from the same 4 categories mentioned at the beginning of this chapter (containing 17 minutes of uncompressed audio each), we split files in segments of 10 seconds, thus obtaining 100 uncompressed file per category for a grand-total of 400 signals. Among these, 240 were chosen at random to build a training set, while the rest were used as the test set. Since the goal is to localize tampered segments within a signal, test signals were created as follows:

1. the signal was MP3 compressed at a bitrate BR1 $\in \{96, 128, 160\}$ kbit/s;
2. the signal was decoded and a portion of 1 second, located at the center of the track, was replaced with the same-positioned samples coming from the uncompressed signal;
3. the resulting track was re-compressed at a bitrate BR1 + Δ, with Δ taking values in {-32, 0, 32} kbit/s.

In such a way, we created a cut-and-paste tampering that is virtually undetectable by a human listener (the pasted content is the same, only without the first compression); this also avoids facilitating the detector by introducing abrupt changes in audio content.

The above procedure creates files where 1/10 of the track is tampered, and the rest is untouched. While this scenario is reasonable for testing localization capabilities, it does not fit well to training a classifier (needed for YSH and LSQ), where a more balanced distribution of positive and negative examples is preferable. Keeping in mind that each signal will be analyzed using small windows, and that each window will be classified as single- or double- encoded, we generated the training dataset as follows:

1. the signal was MP3 compressed at a bitrate BR1 $\in \{96, 128, 160\}$ kbit/s;
2. the signal was decoded and the part of the track between second 5 and 6 was cut and appended at the end of the signal;
3. the resulting track was re-compressed at a bitrate BR1 + Δ, with Δ taking values in {-32, 0, 32} kbit/s.

In such a way, samples in the first half of the track will show traces of double encoding, while samples in the second half will not, due to the induced misalignment of the quantization pattern.

Table 10.8: Tampering localization accuracy obtained by each algorithm for different sizes of the analysis window.

(a) 1/16 s.

	$\Delta = +32$			$\Delta = 0$			$\Delta = -32$		
BR2	OUR	YSH	LSQ	OUR	YSH	LSQ	OUR	YSH	LSQ
64	-	-	-	-	-	-	50.0	64.8	60.5
96	-	-	-	49.0	76.3	57.4	50.0	71.7	56.9
128	93.3	92.3	77.3	49.3	82.2	53.6	50.0	80.5	65.0
160	90.7	92.8	73.9	50.1	84.7	51.0	-	-	-
192	89.8	93.3	93.1	-	-	-	-	-	-

(b) 1/8 s.

	$\Delta = +32$			$\Delta = 0$			$\Delta = -32$		
BR2	OUR	YSH	LSQ	OUR	YSH	LSQ	OUR	YSH	LSQ
64	-	-	-	-	-	-	50.0	67.5	58.0
96	-	-	-	50.0	78.4	60.2	50.0	73.5	57.5
128	89.2	91.7	76.1	50.1	83.1	58.7	50.0	82.4	64.4
160	88.1	91.5	68.7	49.9	86.4	53.3	-	-	-
192	89.2	90.9	91.9	-	-	-	-	-	-

(c) 1/4 s.

	$\Delta = +32$			$\Delta = 0$			$\Delta = -32$		
BR2	OUR	YSH	LSQ	OUR	YSH	LSQ	OUR	YSH	LSQ
64	-	-	-	-	-	-	50.0	73.0	56.9
96	-	-	-	50.0	82.6	61.8	50.0	75.5	56.3
128	91.2	92.6	79.9	50.0	86.8	63.1	50.0	84.2	67.8
160	92.0	95.2	74.2	50.0	89.6	60.3	-	-	-
192	92.0	92.2	93.4	-	-	-	-	-	-

Similarly to what we did in previous experiments, it is of interest to evaluate the localization performance of each algorithm for different sizes of the analysis window: a smaller size causes noisier measurements but higher temporal resolution, allowing the analyst to detect subtle modifications, like turning a "yes" into a "no". The set {1/4, 1/8, 1/16} s was chosen as possible values for the size of the window. When analyzing a signal, the analyst knows both the size of the window he wants to employ and the bitrate of the last compression undergone by the signal. In light of this, the training procedure for LSQ and YSH can be done separately, creating a SVM for each BR2 (in our case, BR2 $\in \{64, 96, 128, 160, 192\}$) and for each window size. We chose RBF kernels and used 5-fold cross validation to determine the best values for $C \in \{2^3, 2^4, \dots 2^{12}\}$ and $\gamma \in \{2^{-4}, 2^{-3}, \dots, 2^5\}$. Concerning our system, we used the training samples to get a good initialization point for the EM algorithm: specifically, we computed the average value of the χ^2 distance obtained for single compressed sequences and double compressed sequences available in the training set, resulting in $\mu_1 = 0.004$ and $\mu_2 = 0.015$ respectively. The algorithm stops when either the log likelihood stabilizes (difference between two iterations lower than 10^{-15}) or a maximum of 500 iterations is reached.

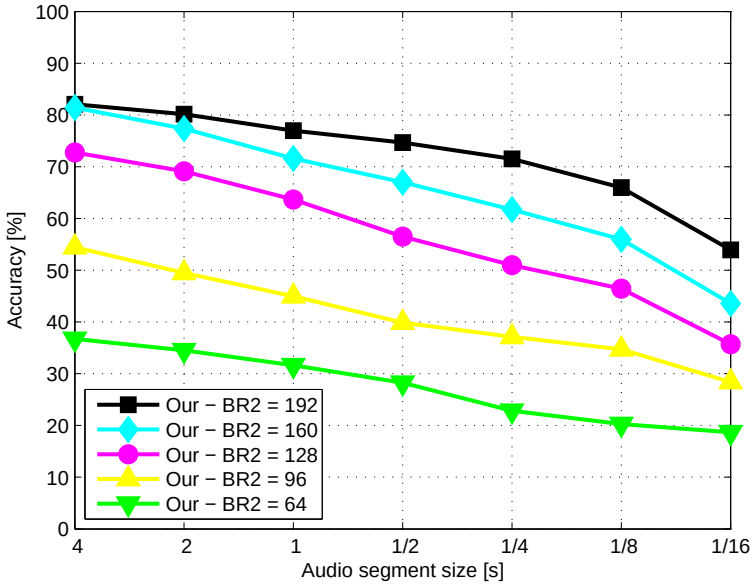
Performance of YSH and LSQ were evaluated as follows: given a test file, the proper SVM was selected based on the observed bitrate and the chosen window size; then the file was decoded and coefficients were classified, moving the analysis window. After repeating the same procedure for all the signals, we computed the true positive (TP) and true negative (TN) probabilities and considered the final accuracy to be

$$\text{ACC} = \frac{TP + TN}{2}.$$

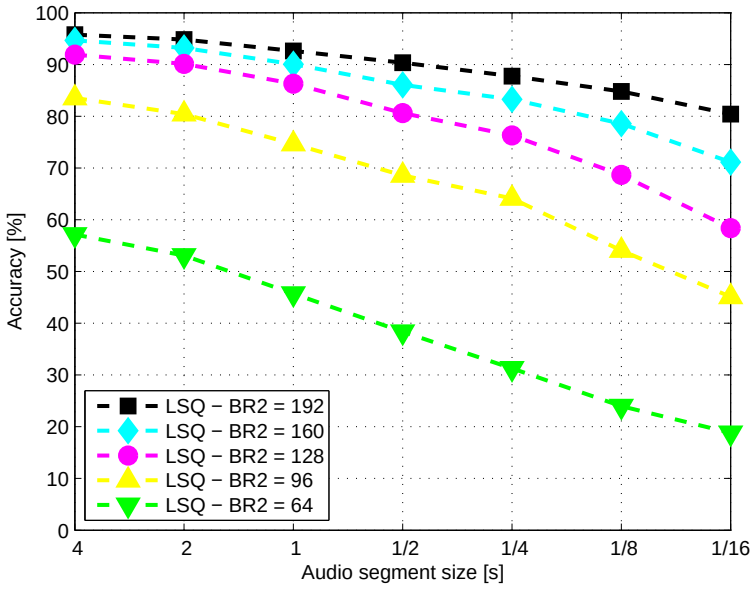
A similar approach was used to evaluate our method. After executing the EM algorithm, if a mixture of two Gaussians was found, we labelled as tampered those windows belonging to the model with lower mean; if only one Gaussian component was found, all the windows were classified as untouched. Finally, the localization accuracy of the algorithm was computed with the same formula described above.

Results are reported in Table 10.8, for different sizes of the analysis window and separated according to the difference between the first and second

compression bitrate (main rows), while the final value of BR2 is given in the columns (this is the only information available to the analyst). Since possible values for BR1 were limited to be in $\{96, 128, 160\}$, some combinations of BR2 and Δ are not explored. As we can see, the proposed system yields comparable performance to the YSH and outperforms LSQ when BR2 is higher than BR1. For null or negative Δs, coherently with previous results, our model is not able to discriminate forged regions.



(a)



(b)

Figure 10.5: Accuracy of the classifiers with varying audio file duration: (a) our; (b) LSQ.

10.4 Conclusions and open issues

In this part of the thesis we presented a method to localize the presence of double compression artifacts in an MP3 audio file, with the aim of uncovering possibly tampered parts. Our algorithm is based on a simple feature measuring the effect of double compression, that allows to decide whether an MP3 track has been compressed once or twice and also to derive the bit-rate of the first compression. In addition, our scheme together with two state-of-the-art methods designed for detecting doubly compressed MP3 files have been applied to analyze short temporal windows, in such a way to allow the localization of tampered portions in an MP3.

The proposed algorithm is very effective when the bit-rate of the second compression is higher than the bit-rate of the first one, but offers limited performance in the opposite case, where it is outperformed by state-of-the-art methods, based on SVM classifiers. On the other hand, we must keep in mind that the proposed approach does not exploit machine learning techniques, as the other schemes do, so its performance are less affected than those of other methods by the choice of a suitable training set.

In our opinion, the main contribution of our work regards the tampering localization scenario. To the best of our knowledge, this is the first time that techniques used for detecting double compression in MP3 audio tracks are used for this purpose. We also point out that the authors in [71, 72] and [70] did not investigate how their methods would perform on very short audio segments. The results we got provide some interesting insights. For example, we see that more complex features, which achieve very good results in the simple detection scenario, may not be well suited for the localization scenario. This is evident for LSQ, whose performance are significantly lower than those achieved in the simple detection scenario. We believe that this is due to the difficulty of providing a good training set for the localization scenario, which negatively affects methods based on machine learning. Moreover, we also see that in the high quality scenario a simple feature based on calibration may obtain performance very close to that of state-of-the-art techniques, without requiring a SVM.

There are some interesting issues that can be considered for further research. For example, the detection and localization accuracies of the different

algorithms may be affected by the actual codec(s) used in the first and second compressions. Since the proposed method is based on a simple histogram distance, we believe that different encoder/decoders should not affect much the performance. Nevertheless, more complex features may be affected by the encoder, especially if the detector is trained on a different encoder. Another aspect is the use of a variable bitrate (VBR) encoding strategy: since our approach does not consider the MDCT coefficients according to the different quantization factors, we think that the different quantization factors induced by VBR will not affect significantly the proposed approach. However, we leave analysis of VBR to future work.

Chapter 11

Other Works: Image Counter-Forensics

BESIDES playing the role of the analyst on different kinds of media, we also took some steps on the opposite side of the battlefield, by collaborating to the development of counter-forensic techniques.

As suggested by the name, counter-forensics aims at concealing the traces introduced by processing tools when the user edits/tampers a multimedia content [82]. Up to a very recent time only *targeted* approaches existed, which aim at deceiving a specific detector: the idea is to exploit knowledge of the forensic algorithm and try to erase the traces it looks for, while preserving perceptual quality. Within this class of methods, we developed a counter-forensic scheme for hiding traces of median filtering [83]. Median filtering detection is an important task in image forensics, since this operator is frequently used both for benign and malicious processing (median filtering is itself a good counter-forensic tool). The basic idea of the counter-forensic technique presented in [83] is to apply a linear filtering to the median filtered image, using a kernel that is automatically searched through an optimization algorithm. The objective function of the optimization aims at maximizing the fidelity between the processed image and the attacked one, while removing footprints searched by the forensic detector. Our system was tested against three median filtering detectors, obtaining good performance both in terms of trace concealment and perceptual quality [83]. However, this approach has two important limitations:

1. it needs to iteratively evaluate the output of the forensic detector for solving the optimization problem;

2. the produced image contains other kinds of artifacts, that could be detected using different (perhaps more sophisticated) forensic tools. For example, one could check whether the image has been filtered.

These limitations are both significant. The first one implies that the forensic detector must be available to the forger (at least as a black-box) and that the produced image is safe only with respect to that specific detector. The second one opens the door to a "cat-and-mouse" game where several iterations of the forensic/counter-forensic loop are carried out. Noticeably, these two limitations are not specific to the method in [83]: they apply to all *targeted* counter-forensic approaches.

As an answer to the above problem, the research community started investigating a different approach to counter-forensic, which aims at designing *universal* CF methods [48]. Instead of concealing the specific footprint searched for by a detector, universal methods aim at making the statistical properties of the image equal or very close to those of unprocessed images. If they succeed, the attacker has the warranty that no detector (based on those statistical properties) can discriminate between attacked and authentic images. Of course, devising universal methods is not easy, so that some assumptions on the complexity of the statistical analysis carried by the analyst have to be made.

We contributed to the development of one of the first methods within the universal class, that is effective against histogram-based detectors [84, 85]. The underlying idea is the following: first the forger processes the image (no restriction are made to the kind of processing), then he modifies the produced image so to make its histogram very close to that of an original image. In order to do so, the proposed counter-forensic scheme works in three phases (illustrated in Figure 11.1):

1. *Histogram retrieval*: given the histogram of the manipulated image, search a similar histogram in a database of original images;

2. *Histogram mapping*: solve an optimization problem to find the best way to modify the histogram, so to bring it as close as possible to the retrieved one, while satisfying some constraints concerning the maximum distortion;

3. *Pixel remapping*: actually change pixels in the image according to the histogram mapping, keeping the perceptual distortion as low as possible.

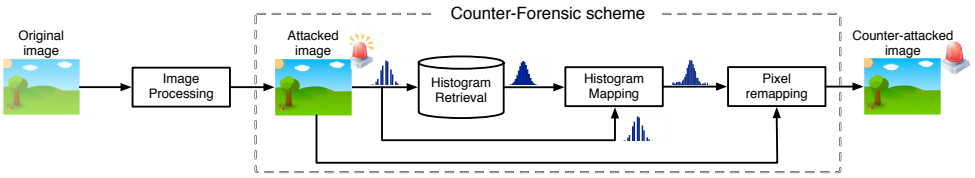


Figure 11.1: A schematic representation of the universal counter forensic approach presented in [84].

Experimental results confirmed that this method successfully deceives detectors whose analysis is based on the image histogram while preserving a high perceptual fidelity: using the method by Stamm et al. [18] as a benchmark, the AUC of the detector drops to about 0.55, while the average SSIM between the image before and after counter-forensic is above 0.98 [84].

The above approach was recently extended to detectors that analyze the histogram of DCT coefficients [86]. Moving from pixels to the DCT domain is not a trivial task because of the perceptual impact of remapping operations in the DCT domain. To tackle with this problem, we proposed to tune the admissible distortion using Watson's model [87], that relates changes in the DCT domain to their perceptual impact.

Despite a good amount of promising results, there is still much to do in universal counter-forensics. For example, our approach against histogram-based detectors cannot be directly extended to deceive detectors that rely on higher order statistics. Moreover, computational complexity makes our method not suitable to combat detectors that *jointly* analyze the histograms extracted from several color channels or DCT frequencies. Nevertheless, the research community is reaching considerable achievements at the theoretical level [48, 88, 89, 90, 91] by combining elements of information theory and game theory. In fact, some of these achievements laid the basis of our works in this field.

Chapter 12

Conclusion

THIS thesis presented several multimedia forensic tools for splicing detection in digital images, video and audio. Besides summarizing our contributions, this final chapter outlines some important open issues that, we believe, should be pursued in the near future, and provides a few remarks on multimedia forensics as a whole.

12.1 Summary

The possibility of blindly investigating whether a digital content represents something that really happened or it is a forged composite is of interest in several fields, prominently forensic investigations. This fact motivated a significant research effort towards the development of blind techniques allowing to detect or even localize forged segments within multimedia contents.

In the first part of the thesis we focused on image forensics. When we began our research activity a considerable number of tools had already been developed, searching for a wide variety of possible manipulations. On the other hand, the forensic analyst can hardly rely on one tool alone, because tools have limited reliability and search for very specific traces. Motivated by this fact, we worked to develop a decision fusion framework tailored to the image forensic scenario. We opted for Dempster-Shafer Theory of evidence as the underlying theory, since it allows to work without prior probabilities and to easily model uncertainty. The most noticeable features of the devised framework are that: i) it allows the analyst to directly specify the known compatibility relationships between forensic traces, and ii) it automatically interprets tool outputs based on background information, so to account for the reliability of tools under the specific working conditions. Experimental results showed that the proposed framework is a valid choice for the analyst,

as it outperforms advanced machine learning methods such as SVMs (under the reasonable hypothesis of a limited training set).

The second part of the thesis moved the attention to video forensics. This branch of research is somewhat delayed compared to image forensics: while there is a large family of tools for detecting double compression, splicing detection has been sparsely investigated. As we learned from image forensics literature, however, double compression analysis can often be used as a tool for splicing detection. Motivated by this fact, we first developed the VPF, a new footprint that can be used to detect double encoding and estimate the size of the GOP used in the first compression. Then, by building on the VPF, we exploited double quantization analysis in order to localize forged regions in frames of MPEG-2 videos. It is interesting to observe how the mentioned technique actually uses "two levels" of double compression analysis: first, it searches for double compression of the sequence as a whole through the VPF; then, on a limited subset of frames, it adapts the double quantization mathematical framework (originally devised for JPEG images) to localize forgeries. Although experimental results are satisfactory, we should not forget that the method relies on a rather narrow set of hypotheses: only MPEG-2 coding is allowed, with a fixed GOP structure both in the first and the second encoding, and with higher coding quality in the second compression.

Still building on the VPF, we also investigated the detection of frame deletion and insertion. We developed a tool allowing to detect this kind of manipulations in a rather reliable way, provided that the second compression is not stronger than the former. While this tool supports all commonly employed coding algorithms (MPEG-2, MPEG-4, H.264), its main limitation is, probably, the limited precision in localizing the point of the stream where the editing took place.

The final part of the thesis dealt with audio forensics. We investigated the analysis of MP3 tracks, with a focus on fake quality detection and forgery localization. Also in this case, we were able to leverage on the heritage of image forensics and steganalysis: we casted the calibration technique to MP3 compression algorithm, thus obtaining the first double encoding detector in audio forensic that is not based on machine learning. Then, mindful of the

rule that a good double compression detector can brilliantly serve as forgery localization tool, we extended our approach in this direction yielding a splicing localization algorithm that can detect manipulations as short as a tenth of a second. Interestingly, it was possible to apply the same extension also to other double compression detectors that were available in the literature.

12.2 Open issues

Before drawing the final remarks, we would like to focus the attention on two topics that have received few attention up to today, namely *multimodal* analysis and *contextual* analysis.

Multimodal analysis is about jointly interpreting information coming from different media to detect anomalies or inconsistencies. The most effective example is the joint analysis of video and audio tracks, that are usually captured together by camcorders. If it is difficult to tamper with a video without leaving noticeable artifacts, it is even more complicated to create a forged audio track that is also consistent with the new frame flow. Effects like reverberation or reflection can help the analyst to detect anomalies by comparing the audio with the environment shown in a video. If we exclude few examples (only the work by Milani et al. [92], to the best of our knowledge), multimodal analysis received little attention in the forensic literature.

As to contextual analysis, it refers to the task of detecting whether a multimedia content is used out of the correct context, so to mislead the user. It is easy to understand that deliberately placing a picture in the wrong position can totally subvert its meaning, or the meaning of surrounding content, even without changing one pixel. We may say that altering the semantic meaning of a picture can be done either by manipulating the picture or by "manipulating the context" wherein the picture is placed. Of course, this kind of investigation sets big challenges, also due to the difficulty of interpreting the semantic meaning of multimedia objects and text. We may consider the existent studies on image phylogeny as a first step in this research direction: given a set of near-duplicate images, phylogeny methods aim at recovering the dependency graph telling which picture originated which [13]. The same idea has been investigated for videos [93]. However, the computational complexity of such

methods already raises a warning about the feasibility of contextual analysis in the general case.

12.3 Final remarks

Multimedia Forensics is composed by branches with significantly different levels of advancement. Limiting our discussion to authenticity verification, audio forensics has the most ancient roots, as it entered the court tens of years ago. Probably, this is due to the fact that tampering with magnetic tapes was much easier than manipulating analog photographies. However, the discovery of the ENF criterion significantly boosted also digital audio authenticity verification. Image forensic received a lot of attention, and today we have tens of different tools, together with many elegant mathematical formulations of topics like multiple quantization or resampling. However there is a concern about practical applicability of image forensic tools, so that only few of them are ready to be used in real world cases today. Finally, video authenticity verification is for sure the less advanced field at the moment. This contrasts with the fact that digital videos have paramount importance in the security field: for example, recordings from surveillance camera are used in many trials, and international terrorism sadly take advantage of video realisticity to upload violent content on the web.

Finally, we spend some words on the main limitations shared by multimedia forensic methods for splicing detection. As long as the literature is concerned, the first enemy of multimedia forensic is counter-forensics, as it aims at erasing the (already fragile) traces left during manipulation. In practice, however, the real enemy is the way digital contents are commonly archived and shared. When an image is uploaded on, say, Flickr or Facebook, it is resized and recompressed by default; something similar happens to a video when it is uploaded on Youtube. Unfortunately, such operations are a very effective, though involuntary, counter-forensic mean. In general, we can say that the main problem for the multimedia forensic analyst is that he has to work on contents whose *integrity* is seldom preserved: for example, no one would fear Flickr as a potential danger to image authenticity, yet it compromises the integrity of the signal by heavily modifying it. The only way for multimedia

forensic to face with this problem is to devise more robust methods, searching for traces that survive these kind of processing. One noticeable example is given by geometrical and physical features (shadows, lighting conditions, perspective consistency); however, such techniques require the manual aid of a clever and patient analyst. Another effective counter-measure is the synergic use of many different tools, hoping that at least some traces of manipulation survive the whole chain linking the forger to the analyst.

Although being aware of all its limits, we believe that multimedia forensics can bring an important contribution in security and justice, so that it is easy to foresee an increasing interest in this topic in the near future. Since the perfect crime does not exist, every little contribution can happen to be fundamental: indeed, the multimedia forensic analyst should never forget that his contribution in a legal case will rarely be the deciding factor (it will more likely be a small part of the complex system in the mind of a judge), yet it may prove important in many situations.

Bibliography

[1] A. Redi, W. Taktak, and J.-L. Dugelay, "Digital image forensics: a booklet for beginners," *Multimedia Tools and Applications*, vol. 51, pp. 133–162, January 2011. [Online]. Available: http://dx.doi.org/10.1007/s11042-010-0620-1 1, 2.2

[2] A. Piva, "An overview on image forensics," *ISRN Signal Processing*, vol. 2013, 2013. [Online]. Available: http://www.hindawi.com/isrn/sp/2013/496701/ 1, 2.1, 2.2

[3] M. Chen, J. Fridrich, M. Goljan, and J. Lukas, "Determining image origin and integrity using sensor noise," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 1, pp. 74–90, 2008. 1, 4.5.1

[4] Z. C. Lin, J. F. He, X. Tang, and C. K. Tang, "Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis," *Pattern Recognition*, vol. 42, no. 11, pp. 2492–2501, November 2009. 1, 4.2.1, 4.5.1, 6.2, 6.3.3

[5] G. Shafer, *A Mathematical Theory of Evidence*. Princeton: Princeton University Press, 1976. 1, 3.1

[6] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, November 2012. 1, 5.2, 5.2.2

[7] R. Maher, "Audio forensic examination," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 84–94, 2009. 1, 8

[8] J. Fridrich, M. Goljan, and D. Hogea, "Steganalysis of JPEG images: Breaking the F5 algorithm," in *Information Hiding*, ser. Lecture Notes in Computer Science, F. Petitcolas, Ed. Springer Berlin Heidelberg, 2003, vol. 2578, pp. 310–323. 1, 9.1

[9] M. Barni and F. Bartolini, *Watermarking systems engineering: enabling digital assets security and other applications*, ser. Signal Processing and Communications Series, M. Dekker, Ed. CRC Press, 2004. 2

[10] N. Khanna, A. K. Mikkilineni, G. T. Chiu, J. P. Allebach, and E. J. Delp, "Forensic classification of imaging sensor types," in *IS&T/SPIE Electronic Imaging 2007*. San Jose, USA: International Society for Optics and Photonics, January 2007, pp. 65 050U–65 050U. 2.1

[11] S. Bayram, H. Sencar, N. Memon, and I. Avcibas, "Source camera identification based on cfa interpolation," in *ICIP 2005, IEEE International Conference on Image Processing*, vol. 3, Genoa, ITA, September 2005, pp. III–69–72. 2.1, 2.2

[12] J. Fridrich, "Digital image forensic using sensor noise," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 26–37, 2009. 2.1, 2.2

[13] Z. Dias, A. Rocha, and S. Goldenstein, "Image phylogeny by minimal spanning trees," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 774–788, 2012. 2.1, 12.2

[14] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, "Image forgery localization via fine-grained analysis of CFA artifacts," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 5, pp. 1566–1577, October 2012. 2.2, 4.5.1

[15] T. Bianchi and A. Piva, "Image forgery localization via block-grained analysis of jpeg artifacts," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1003–1017, June 2012. 2.2, 4.5.1

[16] A. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 758–767, February 2005. 2.2

[17] G. Cao, Y. Zhao, and R. Ni, "Edge-based blur metric for tamper detection," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 20–27, 2010. 2.2

[18] M. Stamm and K. Liu, "Forensic detection of image manipulation using statistical intrinsic fingerprints," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 492–506, September 2010. 2.2, 11

[19] E. Kee, J. O'Brien, and H. Farid, "Exposing photo manipulation with inconsistent shadows," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 28:1–28:12, 2013. 2.2

[20] M. Johnson and H. Farid, "Exposing digital forgeries in complex lighting environments," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 450–461, September 2007. 2.2

[21] H. T. Sencar and N. Memon, *Digital image forensics: There is more to a picture than meets the eye*. Springer, 2012. 2.2

[22] M. Kharrazi, H. T. Sencar, and N. Memon, "Improving steganalysis by fusion techniques: A case study with image steganography," *Transactions on Data Hiding and Multimedia Security I*, pp. 123–137, 2006. 2.3.1, 1

[23] Y.-F. Hsu and S.-F. Chang, "Statistical fusion of multiple cues for image tampering detection," in *Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, October 2008, pp. 1386–1390. 2.3.1, 4.1

[24] G. Chetty and M. Singh, "Nonintrusive image tamper detection based on fuzzy fusion," *International Journal of Computer Science and Network Security*, vol. 10, no. 9, pp. 86–90, September 2010. 2.3.1, 4.1

[25] D. Hu, L. Wang, Y. Zhou, Y. Zhou, X. Jiang, and L. Ma, "Ds evidence theory based digital image trustworthiness evaluation model," in *MINES 2009, International Conference on Multimedia Information Networking and Security*, vol. 1. Hubei, CHN: IEEE, November 2009, pp. 85–89. 2.3.1, 4.1

[26] S. Bayram, B. Sankur, N. Memon, and İ. Avcıbaş, "Image manipulation detection," *Journal of Electronic Imaging*, vol. 15, no. 4, pp. 041 102–041 102, 2006. 2.3.1, 4.1

[27] P. Zhang and X. Kong, "Detecting image tampering using feature fusion," in *ARES 2009, International Conference on Availability, Reliability and Security*. Fukuoka, JP: IEEE, March 2009, pp. 335–340. 2.3.1, 4.1

[28] Z.-W. Sun, H. Li, and Z.-C. Ji, "Fusion image steganalysis based on Dempster-Shafer evidence theory," *Control and Decision*, vol. 26, no. 8, pp. 1192–1196, 2011. 2.3.1

[29] M. Barni and A. Costanzo, "A fuzzy approach to deal with uncertainty in image forensics," *Signal Processing: Image Communication*, vol. 27, no. 9, pp. 998–1010, 2012. 2.3.1, 3.3, 3.3

[30] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," *Annals of Mathematical Statistics*, vol. 38, pp. 325–339, 1967. 3.1

[31] R. Yager, "Aggregating non-independent Dempster-Shafer belief structures," in *IPMU 2008, International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Malaga, ES, June 2008, pp. 289–297. 3.1.2

[32] A. Benavoli, L. Chisci, B. Ristic, A. Farina, and A. Graziano, *Reasoning under uncertainty: from Bayesian to Valuation Based Systems. Application to target classification and threat evaluation.* Rome, ITA: SELEX Sistemi Integrati, 2007. 3.1.2

[33] L. A. Zadeh, "Review of a mathematical theory of evidence," *AI magazine*, vol. 5, no. 3, p. 81, 1984. 3.1.2

[34] A. C. Doyle, *The sign of four.* Lippincott's Monthly Magazine, 1980. 3.1.2

[35] T. Bianchi, A. De Rosa, and A. Piva, "Improved DCT coefficient analysis for forgery localization in JPEG images," in *ICASSP 2011, IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, CZ, May 2011. 3.3, 3.3, 3.4, 4.2.1, 4.5.1, 4.15, 6.2, 6.3.3, 6.3.3, 6.3.3, 6.3.3, 6.3.3, 6.3.3, 6.3.3, 6.3.4

[36] M. Fontani, T. Bianchi, A. De Rosa, A. Piva, and M. Barni, "A Dempster-Shafer framework for decision fusion in image forensics," in *WIFS 2011, IEEE International Workshop on Information Forensics and Security*, Foz do Iguaçu, BR, December 2011, pp. 1–6. 3.3

[37] M. Fontani, T. Bianchi, A. De Rosa, A. Piva, and M. Barni, "A Framework for Decision Fusion in Image Forensics Based on Dempster-Shafer Theory of Evidence," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 4, pp. 593–607, 2013. 3.3

[38] T. Denoeux, "A k-nearest neighbor classification rule based on Dempster-Shafer theory," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 5, pp. 804–813, 1995. 3.3.1

[39] L. Ceriani and P. Verme, "The origins of the Gini index: extracts from Variabilità e Mutabilità (1912) by Corrado Gini," *Journal of Economic Inequality*, vol. 10, no. 3, pp. 421–443, September 2012. 3.3.2

[40] W. Luo, Z. Qu, J. Huang, and G. Qiu, "A novel method for detecting cropped and recompressed image block," in *ICASSP 2007, IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, Honolulu, USA, April 2007, pp. II–217 –II–220. 4.2.1, 6.2

[41] T. Bianchi and A. Piva, "Detection of non-aligned double JPEG compression with estimation of primary compression parameters," in *ICIP 2011, IEEE International Conference on Image Processing*, Brussels, BE, September 2011, pp. 1929 –1932. 4.2.1, 4.2.2, 7.2

[42] H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 1, pp. 154–160, 2009. 4.2.1, 4.2.2, 4.4.1

[43] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 4.2.2

[44] M. Barni, A. Costanzo, and L. Sabatini, "Identification of cut & paste tampering by means of double-JPEG detection and image segmentation," in *ISCAS 2010, IEEE International Symposium on Circuits and Systems*, Paris, FR, May 2010, pp. 1687–1690. 4.5.1

[45] A. Swaminathan, M. Wu, and K. J. R. Liu, "Digital image forensics via intrinsic fingerprints," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 1, pp. 101–117, 2008. 4.5.1

[46] R. Böhme and M. Kirchner, "Counter-forensics: Attacking image forensics," in *Digital Image Forensics*, H. T. Sencar and N. Memon, Eds. Springer New York, 2013, pp. 327–366. [Online]. Available: http://dx.doi.org/10.1007/978-1-4614-0757-7_12 4.5.2

[47] M. Fontani, A. Bonchi, A. Piva, and M. Barni, "Countering anti-forensics by means of data fusion," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, February 2014, pp. 90 280Z–90 280Z. 4.5.2

[48] M. Barni and F. Perez-Gonzalez, "Coping with the enemy: Advances in adversary-aware signal processing," in *ICASSP 2013, IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, CA, May 2013, pp. 8682–8686. 4.5.2, 11, 11

[49] A. C. Bovik, *Handbook of image and video processing*. Academic Press, 2010. 5.1

[50] ISO, "Information technology - generic coding of moving pictures and associated audio information - part 2: Video," International Organization for Standardization, Geneva, Switzerland, ISO/IEC IS 13818-2, 2008. 5.1, 6.3.3

[51] ISO, "Information technology - coding of audio-visual objects - part 2: Visual," International Organization for Standardization, Geneva, Switzerland, ISO/IEC IS 14496-2, 2009. 5.1

[52] ISO, "Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding (AVC)," International Organization for Standardization, Geneva, Switzerland, ISO/IEC IS 14496-10, 2010. 5.1

[53] Y. Su, W. Nie, and C. Zhang, "A frame tampering detection algorithm for MPEG videos," in *IEEE Joint International Information Technology and Artificial Intelligence Conference*, vol. 2, Chongqing, CHN, August 2011, pp. 461–464. 5.2.1, 5.2.2

[54] D. Liao, R. Yang, H. Liu, J. Li, and J. Huang, "Double H.264/AVC compression detection using quantized nonzero AC coefficients," in *IS&T/SPIE Electronic Imaging 2011*, San Francisco, USA, February 2011, pp. 78 800Q–78 800Q–10. [Online]. Available: +http://dx.doi.org/10.1117/12.876566 5.2.1

[55] S. Milani, P. Bestagini, M. Tagliasacchi, and S. Tubaro, "Multiple compression detection for video sequences," in *MMSP 2012, IEEE International Workshop on Multimedia Signal Processing*, Banff, CA, September 2012, pp. 112–117. 5.2.1

[56] J. Xu, Y. Su, and Q. Liu, "Detection of double MPEG-2 compression based on distributions of DCT coefficients," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 27, no. 01, p. 1354001, 2013. 5.2.1, 5.2.2

[57] X. Jiang, W. Wang, T. Sun, Y. Shi, and S. Wang, "Detection of double compression in MPEG-4 videos based on Markov statistics," *IEEE Signal Processing Letters*, vol. 20, no. 5, pp. 447–450, 2013. 5.2.1

[58] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, June 2010. 5.2.1

[59] W. Luo, M. Wu, and J. Huang, "MPEG recompression detection based on block artifacts," in *IS&T/SPIE Electronic Imaging 2008*, San Jose, USA, February 2008, pp. 68 190X–68 190X–12. [Online]. Available: +http://dx.doi.org/10.1117/12.767112 5.2.1

[60] J. Xu, Y. Su, and X. You, "Detection of video transcoding for digital forensics," in *ICALIP 2012, International Conference on Audio, Language and Image Processing*, Shanghai, CHN, July 2012, pp. 160–164. 5.2.1

[61] P. Bestagini, A. Allam, S. Milani, M. Tagliasacchi, and S. Tubaro, "Video codec identification," in *ICASSP 2012, IEEE International Conference on Acoustics, Speech and Signal Processing*, Kyoto, JP, March 2012, pp. 2257–2260. 5.2.1, 6.1

[62] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double MPEG compression," in *MM&Sec 2006, ACM Multimedia and Security Workshop*, Geneva, CHE, September 2006, pp. 37–47. 5.2.2, 7.2

[63] M. Stamm, W. Lin, and K. Liu, "Temporal forensics and anti-forensics for motion compensated video," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1315–1329, 2012. 5.2.2, 6.2, 7.2

[64] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double quantization," in *MMSEC 2009, ACM Multimedia and Security Workshop*. New York, USA: ACM, September 2009, pp. 39–48. 5.2.2

[65] T. Pevny and J. Fridrich, "Estimation of primary quantization matrix for steganalysis of double-compressed JPEG images," in *IS&T/SPIE Electronic Imaging 2008*, vol. 6819, San Jose, USA, February 2008, pp. 681 911–681 911–13. [Online]. Available: http://dx.doi.org/10.1117/12.759155 6.3.3

[66] C. Grigoras, "Digital audio recording analysis - the electric network frequency criterion," *International Journal of Speech Language and the Law*, vol. 12, no. 1, pp. 63–76, 2005. 8.1.1

[67] C. Grigoras, "Applications of ENF criterion in forensic audio, video, computer and telecommunication analysis," *Forensic science international*, vol. 167, no. 2, pp. 136–145, 2007. 8.1.1

[68] D. P. N. Rodríguez, J. A. Apolinário, and L. W. P. Biscainho, "Audio authenticity: Detecting ENF discontinuity with high precision phase analysis," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 534–543, 2010. 8.1.1

[69] R. Yang, Y. Q. Shi, and J. Huang, "Defeating fake-quality MP3," in *MMSEC 2009, ACM Multimedia and Security Workshop*, Princeton, USA, September 2009, pp. 117–124. 8.1.2

[70] R. Yang, Y. Q. Shi, and J. Huang, "Detecting double compression of audio signal," in *IS&T-SPIE Conference on Media Forensics and Security*, San Jose, CA, January 2010. 8.1.2, 9, 10, 10.2, 10.4

[71] Q. Liu, A. Sung, and M. Qiao, "Detection of double MP3 compression," *Cognitive Computation*, vol. 2, pp. 291–296, 2010. 8.1.2, 10, 10.2, 10.4

[72] M. Qiao, A. H. Sung, and Q. Liu, "Revealing real quality of double compressed MP3 audio," in *MM2010, International conference on Multimedia*. Florence, ITA: ACM, 2010, pp. 1011–1014. [Online]. Available: http://doi.acm.org/10.1145/1873951.1874137 8.1.2, 9, 10, 10.2, 10.4

[73] R. Yang, Z. Qu, and J. Huang, "Detecting digital audio forgeries by checking frame offsets," in *MMSEC 2008, ACM Multimedia and Security Workshop*, Oxford, UK, September 2008, pp. 21–26. 8.1.2

[74] R. Yang, Z. Qu, and J. Huang, "Exposing MP3 audio forgeries using frame offsets," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 8, no. 2S, pp. 35:1–35:20, September 2012. [Online]. Available: http://doi.acm.org/10.1145/2344436.2344441 8.1.2

[75] R. Böhme and A. Westfeld, "Feature-based encoder classification of compressed audio streams," *Multimedia Systems*, vol. 11, no. 2, pp. 108–120, 2005. [Online]. Available: http://dx.doi.org/10.1007/s00530-005-0195-2 8.1.2

[76] S. Moehrs, J. Herre, and R. Geiger, "Analysing decompressed audio with the inverse decoder - towards an operative algorithm," in *Audio Engineering Society Convention 112*, 4 2002. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=11346 8.1.2

[77] B. D'Alessandro and Y. Q. Shi, "MP3 bit rate quality detection through frequency spectrum analysis," in *MMSEC 2009, ACM Workshop on Multimedia and Security*. Princeton, USA: ACM, Setember 2009, pp. 57–62. 8.1.2

[78] J. Lukáš and J. Fridrich, "Estimation of primary quantization matrix in double compressed JPEG images," in *Digital Forensic Research Workshop*, Cleveland, USA, August 2003, pp. 5–8. 9.1

[79] C. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, United Kingdom: Cambridge University Press, 2008. 9.1

[80] G. Snecdecor and W. Cochran, *Statistical Methods*, ser. Statistical Methods. NJ, USA: John Wiley & Sons, 1991, no. v. 276. 9.1

[81] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B 39*, pp. 1–38, 1977. 9.2

[82] M. Kirchner and R. Böhme, "Tamper hiding: Defeating image forensics," in *Information Hiding*. Springer, 2007, pp. 326–341. 11

[83] M. Fontani and M. Barni, "Hiding traces of median filtering in digital images," in *EUSIPCO 2012, European Signal Processing Conference*, Bucharest, ROU, August 2012, pp. 1239–1243. 11, 11

[84] M. Barni, M. Fontani, and B. Tondi, "A universal technique to hide traces of histogram-based image manipulations," in *MMSEC 2012, ACM Multimedia and Security Workshop*. Coventry, UK: ACM, September 2012, pp. 97–104. [Online]. Available: http://doi.acm.org/10.1145/2361407.2361424 11, 11.1, 11

[85] M. Barni, M. Fontani, and B. Tondi, "A universal attack against histogram-based image forensics," *International Journal of Digital Crime and Forensics*, vol. 5, no. 3, pp. 35–52, 2013. 11

[86] M. Barni, M. Fontani, and B. Tondi, "Universal Counterforensics of Multiple Compressed JPEG Images," in *IWDW 2014, IEEE International Workshop on Digital-Forensics and Watermarking*, Taiwan, TW, October 2014. 11

[87] A. B. Watson, "DCT quantization matrices visually optimized for individual images," in *IS&T/SPIE's Symposium on Electronic Imaging: Science and Technology*. San Jose, USA: International Society for Optics and Photonics, September 1993, pp. 202–216. 11

[88] M. Barni and B. Tondi, "The security margin: A measure of source distinguishability under adversarial conditions," in *GlobalSIP 2013, IEEE Global Conference on Signal and Information Processing*, Austin, USA, December 2013, pp. 225–228. 11

[89] M. Barni and B. Tondi, "Binary hypothesis testing game with training data," *arXiv preprint arXiv:1304.2172*, 2013. 11

[90] F. Balado, "The role of permutation coding in minimum-distortion perfect counterforensics," in *ICASSP 2014, IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, ITA, May 2014, pp. 6240–6244. 11

[91] C. Pasquini, P. Comesana-Alfaro, F. Perez-Gonzalez, and G. Boato, "Transportation-theoretic image counterforensics to first significant digit histogram forensics," in *ICASSP 2014, IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, ITA, May 2014, pp. 2699–2703. 11

[92] S. Milani, P. F. Piazza, P. Bestagini, and S. Tubaro, "Audio tampering detection using multimodal features," in *ICASSP 2014, IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, ITA, May 2014, pp. 4563–4567. 12.2

[93] Z. Dias, A. Rocha, and S. Goldenstein, "Video phylogeny: Recovering near-duplicate video relationships," in *WIFS 2011, IEEE International Workshop on Information Forensics and Security*, Foz do Iguaçu, BRA, 2011, pp. 1–6. 12.2

Visual and audio contents always played a key role in communications, because of their immediacy and presumed objectivity. This has become even more true in the digital era, and today it is common to have multimedia contents stand as proof of events. Digital contents, however, are also very easy to manipulate, thus calling for analysis methods devoted to uncover their processing history. Multimedia forensics is the science trying to answer questions about the past of a given image, audio or video file, questions like “which was the recording device?", or “is the content authentic?". In particular, authenticity assessment is a crucial task in many contexts, and it usually consists in determining whether the investigated object has been artificially created by splicing together different contents.

In this thesis we address the problem of splicing detection in the three main media: image, video and audio. Since a fair amount of image splicing detection tools are available today, we contribute to image forensics by developing a comprehensive decision fusion framework, allowing to intelligently merge the output of different algorithms. On the other hand, authenticity verification of digital videos is a rather unexplored field: we thus contribute by introducing a novel video forensic footprint, called Variation of Prediction Footprint, and we show how it can be used to detect double video encoding as well as removal, insertion and manipulation of frames. Finally, we tackle the problem of fake quality and forgery detection in MP3 compressed audio tracks.

The Ph.D. School of Information Engineering of the University of Siena is a school aiming at educating scholars in a number of fields of research in the Information Engineering area. The Ph.D. School of Information Engineering is part of the Santa Chiara High School of the University of Siena. A Scientific Committee of external experts recognized Ph.D. Schools belonging to Santa Chiara as excellent, according to their degree of internationalization, their research, and educational activities.