Universidade Estadual de Campinas
Instituto de Computação

Anselmo Castelo Branco Ferreira

Multi-Analysis Techniques for Digital Image Forensics

Técnicas de Multi-Análise para Investigação Forense de Documentos Digitais

CAMPINAS
2016

Anselmo Castelo Branco Ferreira

Multi-Analysis Techniques for Digital Image Forensics

Técnicas de Multi-Análise para Investigação Forense de Documentos Digitais

Tese apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação.

Dissertation presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Supervisor/Orientador: Prof. Dr. Anderson de Rezende Rocha
Co-supervisor/Coorientador: Prof. Dr. Jefersson Alex dos Santos

Este exemplar corresponde à versão final da Tese defendida por Anselmo Castelo Branco Ferreira e orientada pelo Prof. Dr. Anderson de Rezende Rocha.

CAMPINAS
2016

Agência(s) de fomento e nº(s) de processo(s): CNPq, 870358/1997-9; CAPES, 99999.002341/2015-08

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

F413m

Ferreira, Anselmo Castelo Branco, 1982-
Multi-analysis techniques for digital image forensics / Anselmo Castelo Branco Ferreira. – Campinas, SP : [s.n.], 2016.

Orientador: Anderson de Rezende Rocha.
Coorientador: Jefersson Alex dos Santos.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Análise forense de imagens digitais. 2. Aprendizado de máquina. 3. Processamento de imagens- Técnicas digitais. I. Rocha, Anderson de Rezende,1980-. II. Santos, Jefersson Alex,1984-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Técnicas de multi-análise para investigação forense de documentos digitais
Palavras-chave em inglês:
Digital image forensics
Machine learning
Image processing - Digital techniques
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Anderson de Rezende Rocha [Orientador]
Ricardo Dahab
Marco Aurélio Amaral Henriques
Erickson Rangel do Nascimento
Nina Sumiko Tomita Hirata
Data de defesa: 31-05-2016
Programa de Pós-Graduação: Ciência da Computação

Universidade Estadual de Campinas
Instituto de Computação

Anselmo Castelo Branco Ferreira

Multi-Analysis Techniques for Digital Image Forensics

Técnicas de Multi-Análise para Investigação Forense de Documentos Digitais

Banca Examinadora:

- Prof. Dr. Anderson de Rezende Rocha (Presidente)
  IC/UNICAMP
- Prof. Dr. Ricardo Dahab
  IC/UNICAMP
- Prof. Dr. Marco Aurélio Amaral Henriques
  FEEC/UNICAMP
- Prof. Dr. Erickson Rangel Nascimento
  DCC/UFMG
- Profa. Dra. Nina Sumiko Tomita Hirata
  IME/USP

A ata da defesa com as respectivas assinaturas dos membros da banca encontra-se no processo de vida acadêmica do aluno.

Campinas, 30 de maio de 2016

Dedicatory

To my family Tatiana, Vinicius and Sir Giuseppe the *criceto*, the Italian mini hero that showed us that it doesn't matter your size, but what you can do.

A picture is worth a thousand lies (Unknown Author)

Acknowledgments

Dante Allighieri once wrote in the "Divine Comedy" his adventure of self knowledgment, which takes him from hell to paradise. That could never be possible without the help of his always-present friend Virgilio. So, here I am... with my Phd thesis, for which I want to thank all my Virgilios.

Firstly I want to thank my parents-in-law Tieko Sugahara and Yochimassa Sugahara for being my shield against my all kinds of problems. Thank you Vera Magalhaes for helping us and always trying to make our life fun when the worst possible storm comes. I also want to thank my father Amilcar Ferreira for everything. Finally, I want to thank my pets Ping, Pax, Francesco and Giuseppe (in memorian), they gave me lots of fun moments that alleviated my "Phd Stress".

This thesis would never be possible without the help of my army of collaborators. I want to thank my Reasoning on Complex Data Laboratory fellows: Giuliano Pinheiro, Carlos Alfaro, John Edgard Vargas, Pablo Arroyo, Siovani Felipussi and Luiz Navarro. I also want to thank my other Reasoning at Complex data fellows: Ramon Pires, Tiago Carvalho, Filipe Costa, Fabio Faria and Allan Pinto. Thank you all for being part of my academic and personal life.

My research was supported in Brazil by the National Counsel of Technological and Scientific Development (CNPq), so I want to say thanks for the financial support all along. My country also gave me the great opportunity to improve my knowledge, my life experience and moments in Italy. So I want to thank the Brazilian Coordination for the Improvement of Higher Education Personnel (CAPES) PDSE program (99999.002341/2015-08) for supporting my sandwich period. I also want to thank lots of great people of deal whom I knew at Polimi. They are Luca Baroffio, Luca Bondi, Paolo Bestagini, Roman Fedorov, Ilio Catalio, Eleonora Ciceri, Felix Salazar, Eric Umuhoza and Carlo Bernaschina. Also, I want to thank my Italian advisor Dr. Stefano Tubaro for making me a better researcher by always questioning me about my results. I wish the best for all of you.

I also want to thank Evelyn Brandao for helping my family and me to have somewhere to stay in Campinas in our way back to Brazil. Thanks also to our friends Leticia Moreira and Maira Trentin for always caring about my family, no matter where we are and for giving to us great moments in Campinas. You gave peace enough to us to keep on moving. Thank you.

A long time ago I decided to follow the academic life. But I was also very disappointed for not being chosen in Phd. selection procedures. But two particular people appeared in time in my life and changed everything: my advisors Anderson Rocha and Jefersson Alex dos Santos. Anderson always believed in me and gave me the opportunities to collaborate in the research of my future fellows, even when I was still not enrolled in PhD. program. Anderson, I know your story and I am really proud to work and being advised by you. You are the guy whom I want to see when I look at the mirror. Thank you professor Jefersson for all your advices in our present works and also your ideas for future work.

Finally, thank you so much my love Tatiana for your support and for solving all our problems while I was working in my research. Thank you also for giving the reason of my life: our beloved Vinicius. I know you once dreamed about this and I will write exactly this here Tatiana: thank you for believing in me.

Resumo

O desenvolvimento de abordagens para autenticar e indicar a fonte de documentos questionados tem atraído a atenção da comunidade científica nos últimos anos, atenção esta causada, principalmente, pela enorme quantidade de informação disponível hoje em dia para as pessoas comuns, tais como vídeos e imagens, e que podem ser facilmente alteradas para forjar o seu significado. Além disso, materiais impressos são criados diariamente para falsificar documentos como dinheiro, cláusulas contratuais e também são usados para distribuir pornografia infantil. Soluções propostas na literatura frequentemente exploram diferentes ramos de pesquisa, tais como detecção de manipulação de imagem, atribuição de origem, esteganálise, detecção de pornografia, entre outros, por meio da investigação de artefatos característicos nestas imagens chamados de inconsistências estruturais. No entanto, para investigar um determinado artefato para uma determinada aplicação forense, normalmente diferentes abordagens são criadas. Nesta tese, propomos novos algoritmos que realizam análise forense de documentos digitais focados em diferentes aplicações, mas com base em uma idéia principal, chamada de *multi-análise*. Esta nova abordagem para o desenvolvimento de algoritmos de análise forense de documentos digitais leva em conta a análise de diversos cenários na imagem de entrada, tais como a análise de multi-direcionalidade, multi-perturbações, múltiplas resoluções entre outros. Esses cenários investigativos podem ser aplicados em qualquer fase da investigação de uma imagem questionada como, por exemplo, pré-processamento, descrição e classificação. Mostramos através de uma série extensa de experimentos que as soluções propostas para atribuição de impressora a laser, detecção de filtragem de imagens e detecção de manipulação por cópia e colagem são eficazes quando comparadas com os seus homólogos da literatura e que a abordagem proposta nesta tese pode ser a base de diversos outros algoritmos de análise forense de documentos digitais no futuro.

Abstract

The development of approaches to authenticating and pinpointing the source of questioned documents attracted the attention of the research community in recent years, mostly because of the huge amount of information available today to ordinary people, such as videos and images, which can be easily tampered with to in order to produce deceitful information. Moreover, printed materials are daily created to forge documents such as currency, contractual clauses, and also are used to distribute child pornography photos. Solutions proposed in the literature often explore different branches of research, such as image manipulation detection, source attribution, steganalysis, pornography detection among others, by investigating characteristic artifacts in these images called structural inconsistencies. However, to investigate a given artifact for a given digital image forensic application, normally a very different approach is created. In this thesis, we aim at proposing new digital image forensic algorithms focused on different applications, but based on a core idea, called the *multi-analysis*. This new approach to create digital image forensic algorithms takes into account the analysis of several scenarios for the input image, such as the analysis of multi-directionality, multiple perturbations, multiple-resolutions, among others. These investigative scenarios can be applied in any step of the image investigation, such as pre-processing, description and classification. We show through an extensive series of experiments that the proposed solutions for laser printer attribution, image filtering detection and copy-move detection are efficient when compared with their literature counterparts and the approach proposed in this thesis can be the root of several other digital image forensic algorithms in the future.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

The modern press creation and technological revolution of the last century multiplied the volume of information produced by media companies. Additionally, the cheap access of ordinary people to document generator devices such as cameras, scanners and printers, coupled with fast Internet connections, are now making it very easy for anybody to upload digital documents, such as images and videos, to social networks present in the World Wide Web. To understand the dimension of this fact, the video social network Youtube stores 300 hours of video every minute [6], Instagram had 20 billion images uploaded until 2014 [7] and Facebook users send 136,000 photos every minute [8]. Therefore, the term *Information Age* of these present days cannot be more appropriate.

However, the high availability of such data has been raising several questions about misconduct in news companies, science and also in ordinary people's communications. Image editing software such as GIMP and Photoshop easily allow users to tamper with images, changing their semantics and creating false impressions about the facts depicted. For example, this can be made to defame politicians, change a scientific experiment result and fool insurance companies.

To show how tampered images can deceive people, an experiment was performed by Sacchi *et al.* [9]. They showed pictures of two public gatherings in China and Italy edited to make the scenes more dramatic and asked a group of volunteers about the facts depicted in the images. The authors of the study report that most of the volunteers affirmed that those events were bigger and more violent than they really were. This behavior is now yielding policies about information in images, which are supposed to be worth a thousand words. As an example, in France, there are efforts to force advertisements to warn people if manipulated images are used [10].

Manipulated documents are also commonly found in science, a field in which integrity should be the paramount. In 2004, the South Korean researcher Hwang Woo Suk published an article in the renowned scientific journal Science, which included a series of composed images depicting stem cell colonies [11]. In 2007, the researcher R. Michael from University of Missouri and his colleagues removed their article from the same journal, after image tampering was revealed [12]. According to the Office of Research Integrity, an agency that monitors scientific publications in the United States, in 1996, there were 6% of scientific publications that involved suspected image tampering; in 2005, this percentage increased to 44% [13].

Not only tampered with, but also pristine documents are of ethical interest in the information age. The users apparent anonymity in the world wide web allows for the upload of millions of images depicting child pornography, illegal drugs and animal abuse. According to the British institute Internet Watch Foundation, in 2014, the impressive number of 31,443 reports of child pornography websites were confirmed, an 136% increase if compared to the same period of 2013 [14]. Moreover, statistics show that pirated books led to a loss of 500 million dollars in royalties for book authors [15]. Finally, it is also worth mentioning that document generators such as printers can be used to print documents for criminal purposes, such as child pornography photos, fake documents, currency, passports, drug traffic accounting, additional contractual clauses not present originally, and life threatening letters. Hence, it is paramount for law enforcement and civilian agencies to have tools to detect misconducts such as tampered and child pornography photos and also establish the ownership (source) of questioned documents.

The scientific community has made efforts to identify and recognize the author of questioned documents. The research field called Digital Image Forensics aims at detecting the authenticity and integrity of digital content, thereby restoring its trustworthiness. Digital image forensics is an emerging research area, yet there are still several aspects to improve upon, as the creativity of forgers, image editors performance and the development of counter-forensics are limitless.

Digital Image Forensics is usually performed using active [16, 17, 18] or passive methods [19, 20, 21]. Active methods read the hidden information present in the document to indicate its owner and/or authenticity, embedded in its creation step. This information is extracted and compared with the reference data about its authenticity and can be used to verify whether the image was tampered or not in forensic authentications, although it cannot show the exact type of tampering the document has undergone. The problem with this approach is the fact that image editing software and document generator manufacturers are still reluctant to create a standard for image authentication in documents, as their clients want privacy when dealing with images and generating documents. In addition, it is obvious that criminals will not use any kind of digital markings when creating any kind of document.

In passive methods it is assumed that no embedded information was inserted in the digital image and its own structure is used in the analysis. These methods are regarded as the right direction to follow in digital image forensics investigations, as they are more in line with what we deal with in the real world. Based on which kind of tampering operation was applied to the image, passive digital image forensics methods are focused on two scenarios. The first one is related to source attribution [22, 23] in which a document, digital or digitalized, tampered or not, has its source (type and model) revealed. Specific features of the image acquisition devices present in the image can be used, such as noise, texture and geometric distortions. The second scenario largely used in digital image forensics is image tampering detection [24, 25], which aims at detecting inconsistency in the digital image creation, pointing whether the image is tampered or not. Figure 1.1 shows the subdivision of digital image forensics.

According to Figure 1.1, tampering detection focuses on detecting operations involving, but not limited to, four kinds of image manipulation: retouching, splicing, copy-move and

computer generated images. The first one is used to enhance the image quality, where only certain features of the image are highlighted such as light conditions, among others [25]. The last three are used to fool the observers about the fact depicted in the image, using methods to modify and combine pixels. Splicing forgery consists in creating an image by the union of at least two different pictures: a host image and another one, called *alien*. To eliminate possible visual artifacts when joining two different images, the forger usually blurs and feathers the edges of the alien image [26, 27]. Copy-move tampering is the replication of the same object or background or removal of image elements [28]. Finally, computer generated images are synthetic objects from images or videos, adding textures, colors and illumination, approximating them to their real representations, creating fake scenes visually similar to real ones [29].

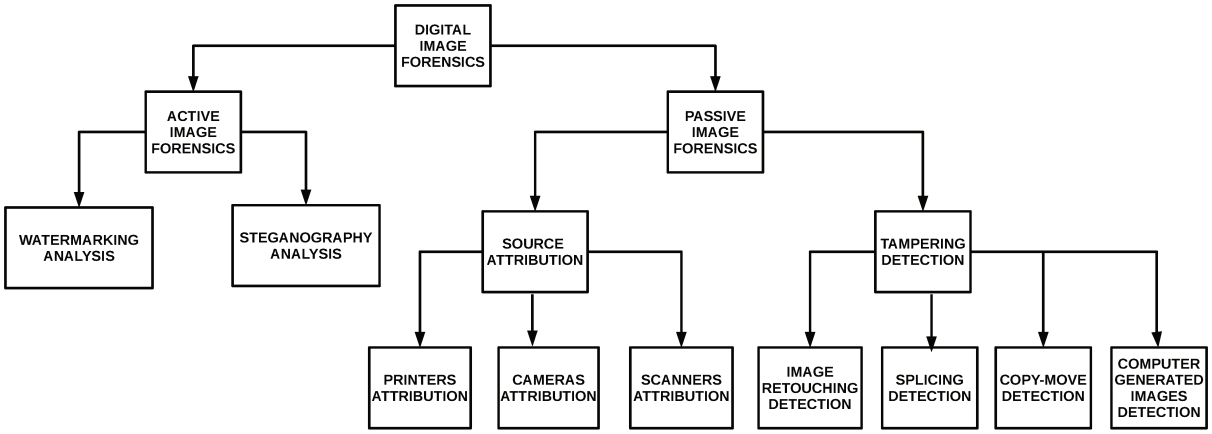


Figure 1.1: Digital Image Forensic Ramifications.

Several techniques from digital image processing can be used in the process of image tampering. They can be found in editors such as GIMP and Photoshop. Some examples are Edge Feathering, Sharpness Edition, Illumination Matching, Lazy Snapping [30, 31], Poisson Matching [32, 33, 34], Structural Propagation [35, 36] among others. Finally, counter-forensics techniques can be applied to erase invisible, but detectable artifacts by machine learning and computer vision approaches [37, 38, 39].

In summary, approaches proposed for digital image forensics in the literature until now work by investigating inconsistencies and telltale signs in images for pinpointing the authorship and authenticating a digital document. For example, for tampering detection there are techniques that investigate lighting inconsistencies [40, 41, 42], JPEG recompression artifacts [43, 44, 45] and edge/region artificial blurring [46, 47]. For source attribution, there are approaches that investigate geometric distortions [48, 49], noise [50, 51, 52], textures [53, 54] among others. As there are different artifacts to investigate, a common behavior found in the methods proposed in the literature is that they use a completely different workflow in different forensic scenarios/applications.

In this thesis we solve three different problems in digital image forensics. To develop these solutions we present a novel approach, called *Multi-Analysis*, which consists on the analysis of several scenarios that a Digital Image Forensic investigator must consider. We envision that different combinations of these scenarios can be used to solve other

problems in Digital Image Forensics with potential of good results. The proposed approach can be used in all digital image processing steps, such as pre-processing, description and classification and involves the analysis of results of several operations (scenarios) applied upon a questioned image, such as:

1. **Multiple Perturbations:** a suspected artifact is artificially inserted progressively in the image and the impact of such operation is methodically measured.
2. **Multiple Scales:** image pyramidal transformations in the image or multi-scale descriptors are used in the investigation to emphasize different aspects of a suspect image.
3. **Multiple Directions:** data are not considered in isolation but in terms of a given neighborhood. The relationship present in a given neighborhood (e.g., pixels neighborhood) can pinpoint additional unseen telltale signs.
4. **Multiple Representations:** several transformations of the same input are analyzed individually in parallel to search for specific artifacts.
5. **Multiple Data:** Different regions of interest are extracted from the raw data and are considered in the analysis.

The multi-analysis scenarios take into account the random behavior of structures from digital images of forensic interest and are important for designing more robust techniques in Digital Image Forensics. The multiple perturbation is useful to investigate if a tampering (structural inconsistency) artifact is present or not in the image; multiple resolution by pyramidal decomposition of images is useful to have approaches resilient to noise and resizing issues, as the decomposition removes the noise by using Gaussian filtering and generate samples with multiple sizes; multiple resolution by multi-scale descriptors is done to generate descriptors that take into account multiple filters in the image tampering procedure; Multi-directionality analysis is important to avoid misclassifications and false hits by classifying artifacts taking into account the neighborhood behavior; Multiple representations pre-process images in a way to better highlight the artifacts, investigating them in parallel and taking into account the complementarity of each individual investigation by aggregating them at the end. Finally, multiple data representations, also known as data augmentation, can provide additional clues in a given task, such as unseen patterns still not present in the analysis, and also contribute with new training examples for a classification problem in need of additional training samples. These scenarios proposed for Multi-Analysis can be used together or in isolation for the authenticity investigation or source attribution of a questioned document.

The scenarios proposed in Multi-Analysis can also be applied to any traditional digital image processing step. For example, multiple perturbation pre-processes the image by disturbing it so it can be used as input for any given statistic calculation for artifact searching. Moreover, Multiple Directions can be considered for the description of a given artifact and also for a classification of a given pixel and so on. By suggesting the use of these operations that can be used in any image processing step and in different applications, we aim at proposing in this thesis a new form of designing Digital Image Forensic algorithms.

1.1 Hypothesis

Our hypothesis is that, due to the complexity of several digital image forensic problems, the *Multi-Analysis* approaches can provide important clues in several applications, such as tampering detection and source attribution that otherwise would be invisible or non-directly accessible.

1.2 Objectives

The objectives of this thesis are:

1. To propose digital image forensic approaches for source attribution, in special for laser printer attribution.
2. To propose digital image forensic approaches for image tampering detection, in special for image retouching detection and copy-move detection.

1.3 Contributions

The main contributions of this thesis for the Digital Image Forensics field are:

1. A new approach, the *multi-analysis*, which is composed of several scenarios that can be combined to generate several digital image forensic techniques, are applied in three image forensic problems and have potential to be applied in others.
2. Four new digital image forensic approaches applied in laser printer attribution, median filtering detection and copy-move detection.
3. New datasets of digital image retouching and laser printer attribution. The latter is the first one in the literature that contains digitalized documents acquired by different scanners.

1.4 Thesis Roadmap

The remaining of this thesis is organized as follows: In chapter 2, we discuss the literature solutions for solving the problems we are aiming at with the multi-analysis proposed in this thesis. In chapter 3, we propose several solutions for laser printer attribution using handcrafted features based on multi-analysis techniques applied in the pre-processing and descriptions steps of image classification using multi-directionality and multiple resolution scenarios. In chapter 4, we propose the first deep learning solution, as far as we know, for laser printer attribution by multi-analysis approaches in the pre-processing step, using multiple representations of multiple data. In chapter 5, we propose a solution for image retouching detection that takes advantage of multi-scale multiple progressive perturbations done in the pre-processing step of image classification. In chapter 6, we propose a classification procedure using our multi-analysis rationale in a novel way to fuse

different copy-move detection algorithms, using multiple directionality and multi-scale scenarios. Finally, chapter 7 concludes this work, discussing the main findings of this thesis and their implications as well as pointing to our future investigative directions.

Part I

Related Work

Chapter 2

Related Work

In this thesis, we aim at using *multi-analysis* approaches to solve diverse problems in the Digital Image Forensic Scenario, such as laser printer attribution, image filtering detection and copy-move detection. In this chapter, we discuss these applications and show what is proposed in the literature to deal with them. As one of the proposed methods in this thesis involves fusion of classifiers, we also discuss some existing methods in the literature for that in this chapter.

2.1 Laser Printer Attribution

Identifying the source of a printed document involves two strategies: the first, known as finding the *extrinsic signatures*, is an active procedure and involves embedding a signature on the printed page. This is done by modifying the document before it is sent to the printer or by encoding identification information, such as the device's serial number. The second and most used way of identifying the source printer is finding a structural inconsistency called the *intrinsic signatures*. This is a passive strategy which is used on a scanned version of the document. It requires an understanding and modeling of the device mechanism to find clues in the printing pattern that are present on the scanned image. Most techniques applied to identification of laser printers take into account an artifact commonly caused by the printer manufacturing process: the *banding*.

To understand the printer artifacts present in printed material and how it can be detectable for laser printer attribution, the Laser Printer (LP) process must be known first. We discuss this in section 2.1.1 and then in section 2.1.2 we discuss literature solutions for identifying the source of a laser printed document.

2.1.1 How Laser Printers Work

Laser printers basically use the attraction of opposite electrical charges in the printing process. The main component of the LP system is a revolving drum or cylinder. This assembly is made of photo-conductive material, which is discharged by light photons of a laser beam. As described by Chiang et al [55], Laser Printers works in six steps:

1. **Charge:** the revolving drum that rotates at a constant angular velocity is positively

charged by a roller or wire having electrical current moving through it.

2. **Exposition:** as the drum revolves, the printer uses a laser beam reflected by a mirror to discharge certain points on the drum, which will be the letters and images to be printed.

3. **Development:** after the pattern has been created on the drum, the printer coats these areas with positively charged ink (or toner) particles.

4. **Transferring:** the printing is done by moving the positive toner particles on the drum to a sheet of paper negatively charged, which moves on a belt below it.

5. **Fusion:** a fuser uses pressure and heat to fuse toner onto the paper.

6. **Cleaning:** to print the next page, a blade cleans the drum to eliminate any residual toner.

Figure 2.1 depicts how LPs work.

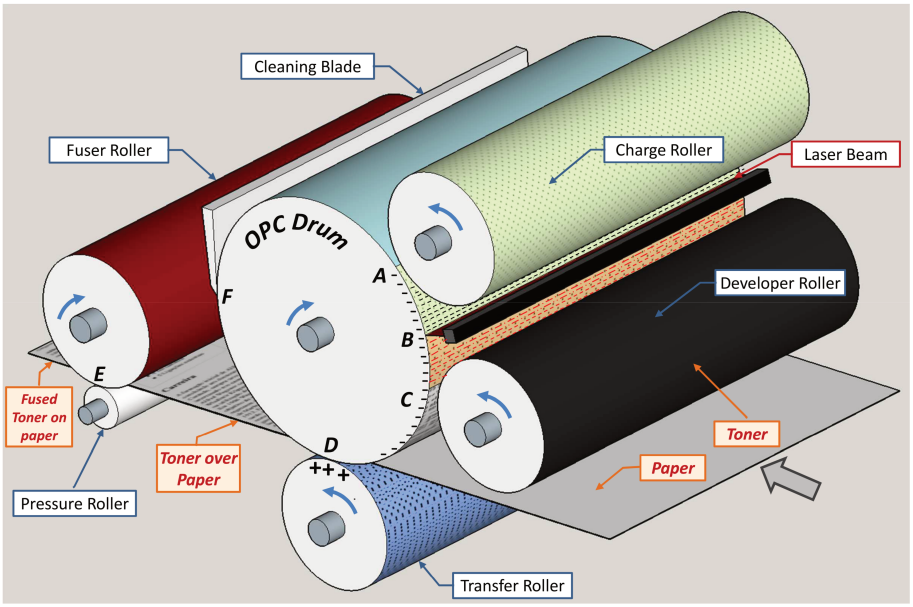


Figure 2.1: Steps of LP workflow: (A) charging, (B) exposure, (C) development, (D) transfer, (E) fusing, (F) cleaning.

In black and white printed documents, colors are represented by grayscale using standard conversion formulas to preserve visual perception characteristics, such as luminance. As laser printers have only one ink that is darkest black, grayscale intermediate tonalities are achieved using density variation from black and white small areas (above human eyes resolution) using halftones. Halftones are an old printing technique consisting in black dots with different diameters over a white surface, which creates a grayscale visual illusion. Common halftone algorithms are error diffusion [56] and clustered dot halftone [57].

As laser printers are electromechanical devices with moving parts, there are many small physical differences on LPs such as motor drifting and gear precision that can be seen on printed pages. These informations patterns can be used as intrinsic signatures of

these devices. The *banding* [58, 59] is an artifact detectable on scanned printed images that can be used to identify the source printer. It is defined as nonuniform light and dark lines perpendicular to direction in which the paper moves through the printer.

Different printing devices have almost unique banding frequencies, depending on model and brand. To recognize this property, several techniques proposed in the printer attribution literature analyze the frequency domain of one dimensional signal of large halftone regions of the document. Studying the Fourier transform of the printed material can be useful to identify the frequencies at which printers work. But those features are only detected at higher resolutions, where variations on distances of halftones can be measured properly. In text documents, whereby only the black color is visible, the absence of halftone areas makes it difficult to perform the Fourier analysis of a signal. In this case, the banding can be seen as textures in specific characters and happens because of toner variations in the *development* stage of the LPs process. This variation is caused by electromechanical imperfections in LPs.

We discuss in the next subsection techniques in the literature which aims at identifying the source printer of documents, using these intrinsic signatures or by extrinsic signatures, which can be understood as visible or invisible watermarks on the printed paper.

2.1.2 Existing Solutions for Laser Printer Attribution

Although our focus here is on discussing Computer Vision approaches for investigating intrinsic or extrinsic signatures for laser printer attribution, they are not the only way to identify the laser printer source of a document. Investigation methods of questioned documents also include physical, microscopical and chemical techniques [60]. Physical marks due to traction mechanisms, traces of toner spread on the paper and electrostatic drum defects create patterns, which can identify specific laser printer devices. On the other hand, chemical components of toner, analyzed by chemical methods such as spectroscopy [61, 62] and x-ray [63] provide information about toner manufacturer and also can be used for comparison with seized evidence materials. Microscopy can also show some patterns on the toner fusion and letter borders.

Some of these methods are destructive, as they require the use of samples extracted from documents on destructive experiments. Another aspect of those methods is that they normally require special laboratory devices, equipment, and also experts to prepare, manipulate and analyze the samples. This does not happen with the same extent with Computer Vision-based techniques, which require only a scanned version of the document and little supervision.

The Computer Vision-based approaches are focused on finding two kinds of signatures in the printed paper, the first one, the *extrinsic signatures*, is already embedded on the printed page and shows identification information, such as the device's serial number. Although it is not the focus of this thesis discussing the search of extrinsic artifacts, examples of such approaches are present in the works of Gaubatz and Simske [64] and Simske et al. [65] on recognizing security deterrents.

The second signature to be found in laser printer attribution are the *intrinsic signatures*, which can be searched in a scanned version of the document. Several computer vision

techniques focused on finding these signatures use similar approaches. Some of them are *halftone-based* [48, 49] and are applied only in color documents, which often contain images. Other techniques are *texture-based* and are applied on text documents [66, 67, 68, 53, 69, 70], in which halftones are not present. There are other techniques which aim at identifying the printer noise [50, 71, 72], among others. Most of approaches can be divided in laser printer attribution of color documents (containing images) and text-only-documents. We discuss both of them in the following subsections. Although this section gives a guided tour on solutions available in the literature for forensic printer attribution, the reader may also want to refer to [55, 58, 73, 74] to find other methodologies and review works.

Solutions for Color Documents

In the literature, the approaches focused on identifying the source printer of color documents (i.e., documents containing images) commonly involve the investigation of intrinsic signatures in the noise, statistics of the transformed image, geometric distortions or in the textures of halftones.

Ryu et al. [75] proposed the analysis of very high-resolution scanned images through histograms of Hough Transform angles in CMYK color channels, generating a feature vector of printing patterns for each document printed by a given printer. The detection occurs by correlating this pattern with a reference created for each printer.

Lee et al. [50, 71] also used the CMYK color space to detect the source of the printer, but in this case the noise was analyzed and the band K was discarded. The Wiener filter was applied over the CMY image and the subtraction between the image and the filtered image yields the noise. Then, five gray-level co-occurence matrix statistics [2] calculations are performed over this noisy image and are used as feature vectors used to feed a machine learning classifier. The analysis of specific color channels was also performed by Choi et al. [76] with Wavelet Transforms on RGB and CMYK color channels and Tsai et al. [77], with Wavelet Transforms and feature selection on RGB images.

Elkasrawi and Shafait [72] also used the noise pattern to identify the printer even with common-resolution scans (400dpi), but their feature vector is based on the work of Khanna et al. [78], in which statistics in the row and column directions of the image are calculated. The filtering of the area is also done differently, with the aid of the Otsu's threshold [79].

Wu et al. [49] used geometric distortions to identify the laser printer source of documents. They first model a projective transformation using the center of characters and the whole scanned image in TIFF version. Then, by solving this model with least squares and singular value decomposition outliers removal, pieces of the model parameters are used as geometric signatures used to feed a machine learning classifier. Bulan et al. [48] also used geometric distortions, but in a different manner. Firstly, geometric signatures are extracted by estimating where the dots in halftone are in training scanned documents of a given set of printers. Then, by correlation, the halftone points in a test document are linked to their source.

Kim and Lee [80] use the halftone patterns for laser printer identification, acquiring images by photography, instead of scanning. First, the image is preprocessed to eliminate

illumination variability, using for that each channel in the CMY domain. Then, a set of 15 halftone texture features are extracted in the discrete Fourier transform (DFT) and are used to feed a Machine learning classifier. This work was extended in [81] using the Curvelet Transform and correlation-based attribution. Finally, Wu et al. [82] create printer models, composed of distance and angles of halftone dots. Euclidean distances allied with K-means clustering help in the final printer identification.

Literature Solutions for Text Documents

For text documents, most of the approaches for printer attribution uses the texture and noise in the printed letters to find the extrinsic signatures of the banding, common to different printers.

Ali et al. [66] used the one dimension pixel values of letters "i" as features after it suffers dimensionality reduction by Principal Component Analysis. Then a Gaussian mixture model machine learning classifier is used for the source attribution.

Mikkilineni et al. [53] proposed the use of texture descriptors based on statistics of gray level co-occurrence matrices to identify the source of text documents. A set of letters "e", which is the most used letter in English texts [83], is chosen to be the data extracted from the documents for classification. Then, 22 statistics of gray-level co-occurrence matrices are extracted and used as input to a previously trained 5-nearest neighbors classifier, with the majority voting of the classified letters of a document defining the final source of the document. In future papers of the same authors a Support Vector Machines classifier was used [69] and in another future work [70], clustering and Euclidean distance were used to identify unknown sources of documents. Jiang et al. [84] proposed the extraction of feature vectors based on Benford's law. The features extracted were the first digit probability distribution of Discrete Cosine Transform coefficients from multi-size blocks.

Kee and Farid [85] proposed to use reference characters and reconstruction error to identify the source of text documents. Firstly, with a reference "e" character of each printer, the search of similar ones from the same printer are done in a training step by template matching. These letters are then used to build the printer profile, useful for printer attribution later on. This profile is firstly built by preprocessing the letters with histogram normalization and registration with the reference letter of the printer. Then the mean character is calculated and the top p eigenvectors from Principal Component Analysis [86] are calculated on the aligned characters, yielding the printer profile. Given a test document, its letters "e" are extracted and each available printer's profile is used to calculate a *reconstruction error*. The smallest mean error identifies the source.

Schreyer [87] used statistical features in the noise image, in the discrete cosine transformed image and in the multi-resolution wavelet transformed image, using them as feature vectors of machine learning classifiers. Mazzela and Marquis [88] studied text and dot-quality objective measurements to differentiate printed outputs.

Finally, other authors have focused on analyzing the attribution problem for other languages. Tsai et al. [89, 67] combined features from statistics of gray level co-occurrence matrices and sub-bands of wavelet transform for laser printer source of Chinese printed documents. As with English language, a specific symbol of Chinese language is chosen for

classification. Tsai et al. [90] extended upon this method by using important statistical features from Gray Level Co-occurrence Matrix, Discrete Wavelet Transform (DWT), Spatial filter, Wiener filter and Gabor filter to identify the source of Japanese printed documents.

Most of the literature methods presented thus far are limited in several ways. First, they are application-focused. In other words, they are applied on documents with text or documents with images. The second limitation is that they are applied only on text databases with the same font style and size. These particularities are not always useful when real-world documents, such as contractual clauses, are investigated. These documents usually have letters with different sizes, configurations (italic, bold, etc.), styles and also can contain figures. Another limitation of most of these techniques is the lack of a public benchmark for comparison. In this thesis we show how to properly deal with these problems using our multi-analysis solutions. This will be discussed in Chapters 3 and 4.

2.2 Image Filtering Detection

The smoothing is a common operation made by image forgers to eliminate some visual artifacts in the tampered image. It can be used to make people younger than they really are, for example, and also to eliminate some tampering artifacts detectable to other digital image forensic approaches. In this thesis, we focus on *multi-analysis* solutions for median filtering detection. In subsection 2.2.1, we discuss how the median filtering is useful for image tampering and the structural inconsistency it generates in the tampered image. Finally, in subsection 2.2.2, we report some works in the literature aimed at detecting this filtering operation.

2.2.1 Median Filtering and Streaking Artifacts

The median filtering is a basic low-pass filter aimed at removing some undesirable high-frequency artifacts, such as noise. Extracting the noise from images is useful for image processing applications that are focused on detecting edges, for example. One particular problem with most of the low-pass filters is the fact that they remove most of high-frequency artifacts, including the edges, which is undesirable for some applications. However, the median filter was proven to be the best to this task [91] because, for some levels of Gaussian noise, the median filter is better than other kinds of filters (such as the Gaussian filter) at removing noise and preserving edges at the same time for a given fixed window size. The median filter is a non-linear filter and does not rely on a convolution in the filtering process (a given window just slides and changes pixel values) [92].

The operation of median filtering in digital images is done by sliding an M mask with size $n \times n$ over the two-dimensional input signal I. This mask defines the size of the neighborhood around each pixel of the image used to perform the filtering. Then, for each image pixel, its value and the ones from the neighborhood (defined by the mask) are sorted. If n is odd, the median value of pixels in the mask is used to change the pixel value located in the center of the mask. If n is even, the median can be the mean of the two center values. For multi-channel images (such as RGB images), each channel is filtered

separately. Problems of lack of values in the boundaries can be solved by padding the image with zeros, using values from the opposite horizontal or vertical or simply avoiding the boundaries in the filtering process.

Because of its inherent property of removing noise while preserving edges, the median filtering can be used to tamper with images in several ways. For example, it can be used for image smoothing [93]. The low pass characteristic of the median filtering removes noise and some other undesirable high-frequency imperfections in faces such as nonuniform light distribution, scratches, blackheads and pimples etc., giving the visual impression of smooth faces.

Another use of image median filtering is hiding traces of image tampering investigated by forensic techniques, working as an anti-forensic approach. This technique can be used to fool forensic techniques such as the one from Johnson and Farid [94] and its extension by Saboia et al. [95]. These techniques detect image manipulation by means of the eye specular highlights in images containing people. In a composite image with two or more people that came from different photos, these techniques identify the forgery by estimating, for each eye in the image, the direction of the light source, viewer (camera) and the normal surface based on specular highlights present on the eye. Inconsistencies of light directions are detected by these techniques. One way to fool this technique is blurring one eye with median filtering and replacing all eyes in the image with the tampered eye.

The median filtering is also used in others anti-forensic techniques such as the one proposed by Stamm et al [39]. This approach can remove blocking artifacts from a previously JPEG-compressed image. The authors found that, by lightly smoothing the image followed by adding low-power white Gaussian noise, it is possible to remove statistical traces of JPEG blocking artifacts. The smoothing is performed by median filtering.

Finally, the median filtering can also be used to hide traces of re-sampling in images detected by the forensic approach proposed by Popescu and Farid [96]. They proposed an Expectation Maximization technique which detects, in an interpolated signal, periodic samples and relations between them. As re-sampling operations use interpolation to reduce or increase the image dimensions, the proposed technique finds the re-sampling by analyzing the unique Fourier signal magnitude of a probability map, generated by Expectation-Maximization steps. Median filter can play an interesting role here, as noted by Kirchner and Bohme [97]: as the proposed technique assumes a linear interpolation of pixels, a non-linear filter (such as the median filter) can destroy these re-sampling artifacts, making the Fourier signal magnitude of the median blurred image the same as the one from a non-blurred image.

A previous study of the median filtering effects gives interesting artifacts for forensic examiners who tries to identify it in digital images. Bovik [98] observed that the median filtering artifacts are identified as equal or nearly equal neighboring pixels, which create a visual impression with no correlation. Bovik called this effect *streaking artifacts*. As the image changes smoothly pixel-wise, when the $n \times n$ filter changes a value and goes to the horizontal/vertical neighbor pixel, only n new pixel values are considered in the filter mask and are used in the neighboring area of the next pixel. Hence, the probability of the previously changed pixel being the median value of the neighboring area of its neighbor pixel is high. This yields horizontal and vertical streaks in the image, and this effect is

referred to *streaking artifacts*.

Given the high use of median filtering operations for image tampering and to anti-forensic applications, several filtering detection techniques were proposed in the literature. We shall discuss each of them in the next subsection.

2.2.2 Existing Solutions for Median Filtering Detection

Kirchner and Fridrich [99] proposed the first method targeting the identification of median filtering in digital images. For uncompressed images such as PNG and TIFF images, they proposed a method based on a histogram of differences between an image and its version translated one pixel. In median filtered images, the *streaking artifacts* refer to pixels with the same value. The ratio of bin related to zero value and the adjacent bins are higher in median filtered images than this same ratio in pristine (non-filtered) images. To detect the median filtering, the authors proposed the use of an ad-hoc threshold. For compressed images, such as JPEG, the authors proposed the classification, by a machine learning classifier, of feature vectors based on Subtractive Pixel Adjacency Matrix (SPAM). The problem with the proposed approaches is the fact that SPAM is vulnerable when JPEG compression is applied to images before compression. Also, cross-dataset experiments were not considered.

Cao et al [100] investigated the streaking artifacts in uncompressed images by analyzing pixel neighborhoods. The proposed method firstly computes pixel differences with neighbors in the top and in the right, resulting two matrices of differences. These matrices are then binarized and create the binarized neighboring differences row-wise and another with neighboring differences column-wise. As the pixels in median filtered images have similar values in a given neighborhood (due to the streaking artifacts), the variance in a squared region around each pixel is low. A map of pixel variances in a neighborhood is then built and also binarized. These three matrices are used to calculate features that, when summarized, yields a measure that is used to identify the median filtered images. The drawback of this approach is the fact that it has difficulties to discriminate between median filtering and other kinds of smoothing operators. Also, cross-dataset experiments were not considered.

Yuan [101] noticed dependencies in overlapped neighboring blocks of pixels in median filtered images. The author states that these dependencies identify the median filtering and proposed a set of five metrics to be calculated per pixel in each $s \times s$ non-overlapped blocks, yielding $s \times s$ five dimensional vectors per block, one five dimensional vector per pixel in this block. The mean of the five metrics per corresponding pixel is calculated in all blocks, yielding $s \times s$ five final dimensional vectors. These vectors can be combined as a feature vector to train a machine learning classifier to identify the median filtering. The limitation of this approach happens when looking for local median filtering on images with low quality factors JPEG compression. Cross dataset experiments were not considered..

Chen and Ni [102] noticed that, compared with images not blurred by median filtering, the median filtered images exhibit characteristic traces around edges (neighborhood correlation, noise suppression and good edge preservation). These fingerprints are characterized through an Edge Based Prediction Matrix (EBPM) containing the estimated

prediction coefficients of neighborhood among different edge regions in images. Firstly, the image is divided in $B \times B$ overlapped blocks. For each block, the horizontal and vertical gradient features are calculated and the blocks are classified according to these gradient values. Then EBPMs are calculated on the three kinds of blocks separately, yielding $3 \times (B^2 - 1)$ prediction coefficients that are concatenated to yield a feature vector used in a machine learning classifier. The results reported are promising but were done only in one uncompressed dataset. Cross dataset experiments were not considered.

Chen et al. [103, 104] stated that median filtered images inevitably exhibit distinctive statistical artifacts in the difference domain. They explored the cumulative distribution function of the first and second order pixels differences of non-filtered, median filtered and linear filtered images as fingerprints to construct the global probability feature set (GPF). They also used the local correlations between different adjacent image difference pairs to construct the local correlation feature set (LCF) and proposed an approach, which yields 56 features used by an SVM classifier (44 based on GPF and 12 based on LCF). Limitations of this approach happen on images with decreasing JPEG quality factor and cross-dataset experiments were not considered.

Kang et al. [105, 106] used a novel approach to identify the median filtering based on the difference between a disturbed (median filtered) image and the input image, instead of just analyzing the input image. This artifact is investigated because the authors want to remove the interference from the image content, such as edge and texture. The median filter used to disturb the image uses a 3×3 mask and this image is called the Median Filtered Residual (MFR). This image is used as input to an autoregressive model in row and column-wise, yielding 10 coefficients used to feed an SVM classifier. The authors report good results but cross-dataset experiments were not considered.

Zhang et al [107] proposed a novel local texture descriptor to detect median filtered images: the second-order local ternary pattern (LTP). The nth order LTP operator is a set of matrices (one per direction) which encode the n-order pixel differences using a 3-valued code (-1,0 and 1). They extract a feature vector based on the LTP matrices in the following steps: firstly, the LTPs are calculated for each pixel in four neighboring directions by coding the difference between eight pairs of central and neighbor pixels in a 3×3 area and their neighbors in that direction. Each pixel has four matrices with eight binary values, coding its neighborhood in a 3×3 area. The LTP is then divided in two LTP matrices, considering the positive and negative pixel difference halves. The binary values in each direction are converted to decimal and histograms per positive/negative matrices are then concatenated to form the final feature vector with 2,048 dimensions (256 bins and 4 directions for positive values, 256 bins and 4 directions for negative values are $1,024 + 1,024 = 2,048$ dimensions). The results are promising but the proposed approach was compared only against two approaches from the literature. Also, cross-dataset experiments were not considered.

Other limitation of most of the proposed approaches for this task is the fact that they don't know a priori the size of window used to propagate the streaking in the filtered image. In this thesis we show how to properly deal with these problems using our multi-analysis solutions. This will be discussed in chapter 5.

2.3 Copy-Move Forgery Detection

One of the most common and also most powerful image forgery operation consists of copying and moving parts of the image as to conceal existing elements or, sometimes, to duplicate others. Although this operation is easy to perform, detecting it is challenging, as it consists of combined transformations, such as compression, illumination changes, resizing, rotation, etc. The structural inconsistency found in copy-move images is an unusual amount of similar pixels in the image, and several literature approaches deal with this problems investigating this structural inconsistency.

In a recent paper, Christlein et al. [1] stated a workflow of common copy-move forgery detection approaches (Figure 2.2). Firstly, the image is pre-processed. This can be, for instance, the merging of color channels to generate a grayscale image that can work properly for a given approach. Then, the feature extraction can be done in two kinds of data. For blocks, the image is tiled in overlapping blocks of size $b \times b$. If using image keypoints, then a keypoint extraction algorithm like SIFT or SURF is applied over the image, yielding points of interest which will be analyzed by the proposed techniques. All these data are represented by feature vectors F_i (for blocks, raw pixel values, block statistics or transformed pixels are used as F_i. For keypoints, SIFT and SURF vectors can be used) and the search for similar feature vectors is done. Then, the filtering of feature vectors not close enough is performed to eliminate false positives. Finally, a post processing is done to guarantee that a meaningful detection is performed.

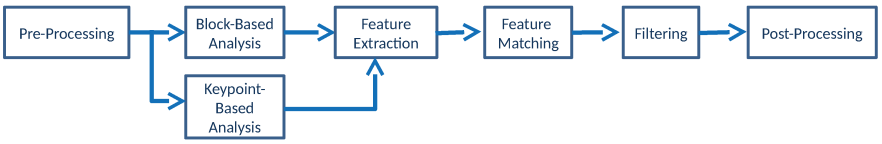


Figure 2.2: Common worflow for copy move detection approaches according to Christlein et al [1]

The differences due to the computational cost, particularities and the detection result make the differences between block-based and keypoint-based methods very important. We will start in the next subsections to discuss approaches proposed using each of them separately.

2.3.1 Block-Based Copy Move Detection

The simplest form of searching for similar patches in the image is using the raw pixel values in blocks as depicted in Figure 2.3 and it was firstly proposed by Fridrich et al [108]. The proposed *Exact Match* is done by using a square sliding window to collect image blocks, which are stored in a matrix. After that, the lexicographic sorting is applied in the matrix, and two consecutive identical lines are considered as cloned regions. The problem with this approach is the fact that lossy compression, such as the JPEG compression scheme can hide some pixel details in similar areas and this can confuse the detection. To deal with this issue, the authors proposed the *Robust Match*. The *Discrete Cosine Transform* (DCT) is applied to each block and the same process is repeated. The problem with this

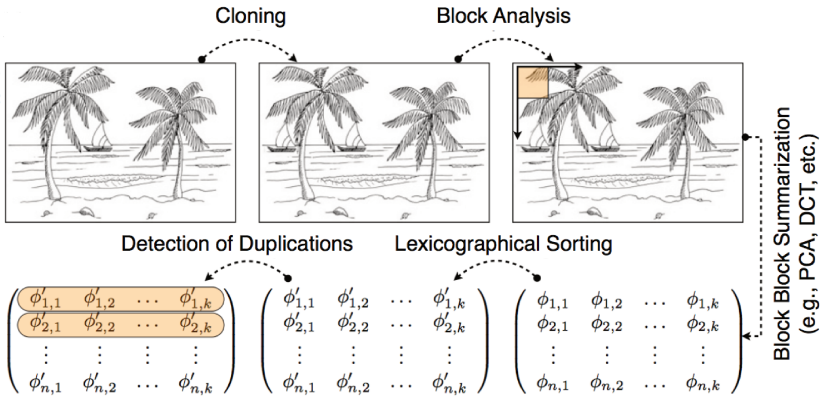


Figure 2.3: Block-Based Copy-Move detection. In this approach, overlapping or non-overlapping blocks are captured by sliding windows in the image. The data can be captured as image pixels or after transformations of the image. These data are stored in a matrix and similar blocks are searched by lexicographical sorting and similarities thresholding.

approach is the fact that it is not able to find forgeries in scenarios such as Gaussian noise, rotation and resizing.

Popescu and Farid [109] used the Principal Component Analysis (PCA) over the raw pixel values in the blocks, reducing the dimensionality of the blocks. This increases the efficacy of the method when scenarios such as JPEG compression and Gaussian noise addition happens, but cannot work properly on scenarios with rotation, resizing and flipping.

Luo [110] extracted features based on pixel intensities in each RGB channel and on some directional information, from each block. The authors reported good results for JPEG compression, additive Gaussian noise and Gaussian blurring. But this approach is not able to find copy-move operations when image geometric transformations are applied.

Mahdian and Saic [111] proposed the matching of blocks over feature vectors built with 24 blur moment invariants on each block of pixels. These 24 moments are calculated in each channel of RGB images, totaling a 72-dimensional feature vector. The authors employed the PCA in this work to reduce the dimensionality of the feature vectors. The detection can be done even in the presence of blurring, additive Gaussian noise and JPEG compression of the image, however, the detection in image geometric transformations was not considered in the experiments.

Zhang [112] proposed the matching on blocks after they are transformed by the *Discrete Wavelet Transform* (DWT). The author showed the robustness of the proposed method only under JPEG compression and blurring. Li [113] also used DWT with *Singular Value Decomposition* (SVD) to generate a more robust block representation. This approach was effective in the cases of JPEG compression with high quality factors. Kang and Wei [114] also used SVD to extract feature vectors from blocks. The authors stated that SVD has algebraic and geometric invariance and it is invariant to noise. However, rotation and resizing of cloned parts were not reported in the experiments of this last work.

Ryu [115] proposed to calculate Zernike moments, which are invariant to rotations on the cloned parts, on overlapping blocks to detect copy-move operations. Such work was

extended later in [116], with a better block matching procedure. However, resizing was not treated by this method. Bravo-Solorio and Nandi [117] took the correlation coefficient of the *Fourier Transform* to find similar blocks of pixels in log-polar form. A filtering approach was used later to discard blocks which the entropy is lower than a threshold. The experiments showed robustness to deal with flipping and simple copy-move operations, but there are problems when dealing with resizing and rotation.

Bashar [118] used the block representation by *Discrete Wavelet Transform* or by *Kernel Principal Component Analysis*. Features extracted from blocks are organized into a matrix and the lexicographical sorting and thresholds are used to detect possibly tampered regions. This approach cannot handle geometric operations (*e.g.*, resizing and shearing), as reported by the authors. Bayram [119] proposed to employ *Fourier-Mellin Transform* (FMT) to describe the blocks and *Counting Bloom Filters* to sort them. The method was able to identify clonings under JPEG compression with great quality factors, resizing and rotation only in limited conditions.

Wang [120] proposed the use of circular sliding windows with *Gaussian Pyramid Decomposition* [121] to describe image blocks. The method showed good results for rotation, blurring, JPEG compression and horizontal flipping transformations, but was intolerant to resizing operations. Later, Wang [122] applied the same Gaussian Pyramid Decomposition and proposed to calculate the Hu moments for the block of pixels. Experiment results showed robustness to Gaussian noise, JPEG compression and blurrings. However, other operations were not detected by the proposed method, such as a 90° rotation, and a horizontal flipping case. Lin [123] used a 9-dimensional feature vector containing information in some specific regions of the block to represent them. Also, the author used the *Radix-Sort* algorithm, which offers a linear time sorting. The experiments showed robustness to JPEG compression and Gaussian noise only.

Ardizzone and Mazzola [124] used spatial domain approach called *Bit-Plane Analysis* to detect copy-move forgeries. This method is based on the decomposition of bit-plane slices and the encoding of the bit blocks with respective ASCII values. This process produces high accuracy in reasonable time, however it was not robust to post-processing operations. Barnes [125] developed a randomized approach to detect similar blocks of pixels in an image, based on the *PatchMatch Algorithm* [126]. The authors showed only a couple of simple experiments in their paper.

2.3.2 Keypoint-Based Copy Move Detection

The advantage of using keypoints to detect copy-move is its reasonable invariance to geometric transformations such as rotations and resizing and also to noise and lighting adjustments. Basically this set of approaches work by firstly detecting keypoints in the image, and then finding similarities on the representing vectors of these keypoints. Figure 2.4 shows an example of this branch of copy-move detection techniques.

With that in mind, Huang [127] proposed an approach based on similarity search of Scale-Invariant Features Transform (SIFT) keypoints descriptors. This method does not work well for finding small duplicated regions. Pan and Lyu [128, 129] also employed these descriptors in an almost similar manner and the experiments results show effectiveness

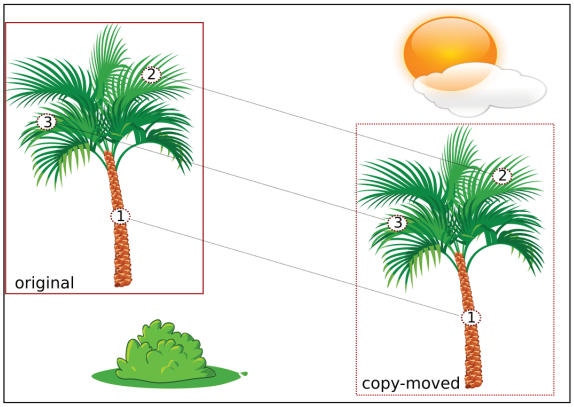


Figure 2.4: Block-Based Copy-Move detection. In this approach, overlapping or non-overlapping blocks are captured by sliding windows in the image. The data can be captured as image pixels or after transformations of the image. These data are stored in a matrix and similar blocks are searched by lexicographical sorting and similarities thresholding.

for finding copy-move operations when images suffer JPEG compression, Gaussian noises, rotation and resizing, but are not effective when other mixed operations on images are performed.

Amerini [130] performed the analysis of SIFT correspondences by means of an hierarchical clustering procedure, which was proven to be useful to estimate the geometric transformation on the replicated segments. This reduces the false positive rates and can detect segments replicated multiple times. The approach was examined in scenarios comprising rotations, symmetric/assymetric resizings, JPEG compression and other operations. High true positive rates were reported but transformations used were very limited.

Xu [131] used the *Speeded-Up Robust Features* (SURF) keypoint descriptors algorithm as keypoints to locate copy-move regions. The authors randomly divide the SURF keypoints set in the image into two clusters, finding the nearest neighbors in these groups and repeating the process for each sub-group, until all of them become unitary. The authors reported experiments with rotation, resizing and blurring segments along with white Gaussian noise insertion. Combinations of these operations were also considered. However, such method may have difficulties in locating cloned homogeneous segments, as they could not provide enough keypoints for a proper analysis.

Shivakumar and Baboo [3] proposed a methodology in which SURF keypoints are extracted and stored in a Kd-tree. This data structure is used to improve and facilitate the search for keypoint matchings (nearest neighbors), and, according to the authors, to generate lower false negative rates. The paper lacks a deep experimental validation, though, by only presenting visual results of the detection in two cases of rotation, resizing, and Gaussian noise independently.

One recent technique used points of interest and blocks of pixels at the same time to detect tampering. Proposed by Silva et al [4], this technique is applied over the HSV color channel version of the suspected image. Firstly a set of SURF keypoints are detected in the image. Then the authors perform the matching of keypoints by a policy known as the *Nearest Neighbor Distance Ratio* (NNDR) policy [132]. A clustering procedure is done to yield a set of areas called source and another called destination, which will be examined

separately and are considered as good cloning candidates. These correspondent regions are delimited by the spatial distribution of their keypoints and rectangular windows are used to bound them. Then the image suffers the Gaussian Pyramidal Decomposition and the sets of source and destination areas are analyzed individually in each scale. To do this, a circular sliding window such as the one used by Wang [120] is applied only on correspondent regions found previously, then a binary detection map is generated per scale. Finally, a voting Scheme is used on the multiscale binary maps to generate a final output image. This approach was validated on two complex datasets, but the results were only promising when uncompressed images are used.

2.4 Fusion of Classifiers

With all the pros and cons of each method discussed previously, one could wonder how to combine some of them in order to obtain a more resilient forgery detector or printer attribution algorithm. Using the example of copy-move detection, patch-based approaches are good at detecting slight illumination changes, but are unable to detect image transformations such as rotation and resizing of copied portions of an image. The keypoint-based approaches, on the other hand, are good at detecting some scaling and orientation changes and are partially invariant to illumination changes and affine transformations, while having problems with both small tampered and homogeneous regions. In this vein, the ideal strategy would be exploring the best of both worlds. In the remaining of this section, we discuss some common fusion approaches proposed in the literature for performing fusion of classifiers.

2.4.1 Majority Voting

This scheme considers only the most likely class provided by each classifier and chooses the most frequent class label within the output set. A variant of majority voting is the weighted majority voting, which multiplies each vote by a weight before the actual voting.

2.4.2 Threshold voting

This voting technique considers a threshold to decide whether the example belongs to the positive class, according to the sum of positive outputs of the combined classifiers. For example, while majority voting considers three out of five votes for deciding upon an outcome of a 2-class problem, threshold voting may arbitrarily choose two as the minimum necessary number of votes for a given class of interest.

2.4.3 Bayesian fusion

Another form of combining different classifiers was proposed by Xu et al.. [133] and aims at combining K multi-class classifiers with a Bayesian approach, assuming each classifier is independent of the other ones. Firstly, for each k binary pixel-based forgery detectors, a

confusion matrix is constructed

$$M_k = \begin{pmatrix} F_{00}^{(k)} & F_{01}^{(k)} \\ F_{10}^{(k)} & F_{11}^{(k)} \end{pmatrix}, \tag{2.1}$$

where F_{ij} is the number of pixels where the detector k misclassified a pixel belonging to class i as belonging to class j. The diagonal contains the correctly classified cases. These confusion matrices are used to calculate the conditional probability that a pixel x belongs to class i, provided that there is an observation on the output of the forgery detector k, predicting that it belongs to class j in Equation 2.2.

$$P(x \in c_i | \epsilon_k(x) = j) = \frac{M_{ij}^{(k)}}{\sum_{i=0}^{1} M_{ij}^{(k)}}, i \in \{0, 1\}. \tag{2.2}$$

Finally, we approximate the probability that a pixel is actually a forgery given K observations on the classifiers we are combining:

$$P(x \in c_1) = \frac{\prod_{k=1}^{K} P(x \in c_1 | \epsilon_k(x) = j_k)}{\sum_{i=0}^{1} \prod_{k=1}^{K} P(x \in c_i | \epsilon_k(x) = j_k)}. \tag{2.3}$$

The probability of a pixel belonging to the forgery class is calculated by Equation 2.3, using both the conditional probability derived from the confusion matrices in Equation 2.2 and the K-dimensional vector of observations of the detectors outputs for this pixel.

2.4.4 Behavior Knowledge Space

An issue with the Bayesian combination is that it assumes that the decisions of the classifiers are independent. Behavior-Knowledege Space (BKS) [134] was developed to avoid this assumption and derives the information from a knowledge space, which records the decision of all classifiers on each learned sample.

The BKS method is a trainable combination scheme that seeks to estimate the a posteriori probabilities by computing the frequency of each class for every possible set of classifier decisions, based on a given training set. BKS builds a lookup table that matches the final classification result with each combination of classifier outputs. For each combination of outputs in the lookup table, it associates the most often class label to it, giving a specific classifier decision $D_1, ..., D_K$ from K individual classifiers. The posterior probability $P(c_i | D_1, ..., D_K)$ of class c_i is computed as follows:

$$P(c_i | D_1, ..., D_K) = \frac{N(c_i, D_1, ..., D_K)}{\sum_{i=1}^{|C|} N(c_i, D_1, ..., D_K)}, \tag{2.4}$$

where $|C|$ is the number os classes and $N(c_i, D_1, ..., D_K)$ counts the frequency of class c_i for the classifier combination output $\{D_1, ..., D_K\}$. If K is the number of combined classifiers, then BKS requires estimates of $|C|^K$ a posteriori probabilities.

In order to perform the combination of classifiers using the BKS method, we need to build a lookup table based on observations on the training dataset. We should be aware

that the amount of possible entries for a set of K binary detectors is 2^K, making it difficult that all possible cases are covered when K is large. This poses a serious problem, given that the set of points in the testing environment can include some of those that lack of an entry in the Behaviour Knowledge Space. Also, to classify the pixel, the neighborhood behavior in BKS fusion is not taken into account. Our multi-analysis solutions to apply BKS classifiers fusion aimed at copy-move forgery detection are discussed in chapter 6.

Part II

Multi-Analysis Solutions for Source Attribution

Chapter 3

Handcrafted Solutions for Printer Source Attribution

In this chapter, we deal with the source attribution of printed documents by proposing a family of handcrafted *multi-analysis* approaches. These approaches use multi-directional and multiscale procedures and act in the pre-processing and description steps of the input data. These techniques were created by feature engineering after a microscopic investigation of the structural inconsistency called *banding* present in printed materials.

3.1 Motivation

The massive use of printers is now giving rise to questions about authenticity of printed documents. Today, unknown contractual terms can be added easily and a forged correspondence can be linked to an innocent. Also, documents related to crimes such as child pornography photos, fake travel tickets, terrorist plots, fake money, pirated copies of books and illegal drug selling accounting are constantly printed everywhere. Identifying the source printer of these documents is an important clue to pinpoint their owner.

In this chapter, we propose three solutions based on *multi-analysis* aimed at the identification of the source printer of a document that explore these intrinsic signatures. The proposed solutions do not need very high resolution digital versions of documents and take into account that this problem requires multidirectional and multiscale analysis, because of different printing patterns yielded by different manufacturing processes. The proposed solutions described in this chapter are:

1. Two descriptors based on multidirectional and multiscale properties of texture micro patterns. These descriptors are applied in text letters or regions of interest. These descriptors are focused on the inner part of printed letters.

2. Another descriptor, here described as the Convolution Texture Gradient Filter (CTGF). The CTGF is built as a histogram of low-level gradient filtered textures. We use filters of one or more scales, which are focused on filtering inner and outer parts of printed letters and figures.

3. The application of descriptors on segments of a document, called frames. With this approach, we can identify the printing source of a document even if parts of it are unavailable or with problems. If the whole document is available, we can use this approach allied with fusion strategies, which provides even more reliable results.

We perform experiments in a well documented printed document benchmark, which is a very difficult one containing different letter sizes, styles and figures. The dataset was created within the scope of this work and is freely available though FigShare[1] along with the source code of the proposed methods available on GitHub[2]. Finally, we show that the presented techniques are very competitive and have important properties when compared to others in the literature.

We organize the remaining of this chapter in four sections. Section 3.2 discusses our approaches for laser printer attribution; Section 3.3 shows the setup we use in this chapter for validating the proposed methods and the existing counterparts in the literature, while Section 3.4 shows the performed experiments and results. Finally, Section 3.5 concludes this chapter with our final considerations and proposals for future work.

3.2 Proposed Approaches for Laser Printer Attribution

The techniques proposed in this chapter were originated by a series of microscope analyses of printed documents. We investigated pictures (of same position on original document) of three letters from three documents, printed by different printers (they can be seen in Figure 3.1). Although borders are more irregular and show more differences between printed characters, even on characters of the same printer, it is noticeable that inside the letters there are micro textures with different sizes and directions.

This investigation enforces our hypothesis that multidirectional and multiscale texture analyses are useful to identify the source printer. In documents with different font configurations, sizes, styles, and figures, the printing patterns are spread over different directions. Hence, the contributions of this chapter for laser printer attribution are:

1. The analysis of multidirectional texture patterns captured through Gray-level Co-occurrence matrices, which is a set of statistics calculated over eight gray-level Co-occurrence matrices, each one representing one texture direction

2. The multidirectional and multiscale approach, applied again over gray-level co-occurrence matrices. These two first approaches are applied inside the printing material (*e.g.*, letters), which is the area where the micro texture pattern is spread.

3. The Convolution Texture Gradient Filter (CTGF). This descriptor is created as histograms of filtered printing patterns over low-gradient areas. These areas are located commonly inside the printing material and close to the borders. We also extend this proposed approach to take advantage of multiscale filters, which increases the printing pattern investigated area. We use these low-gradient areas because they

[1]http://dx.doi.org/10.6084/m9.figshare.1263501
[2]https://github.com/anselmoferreira/printer_forensics_source_code

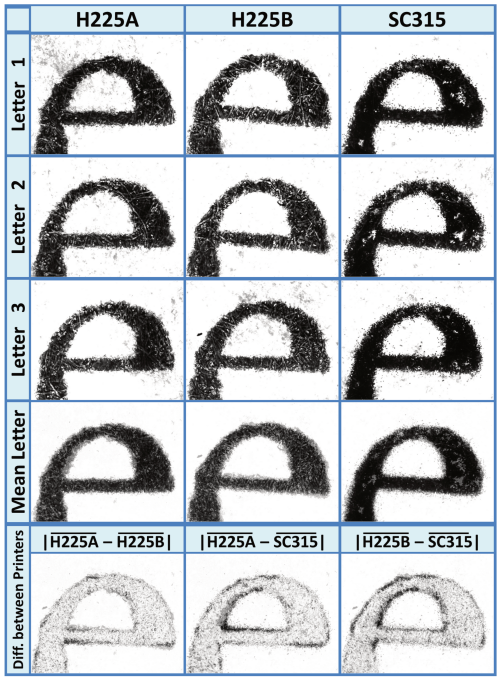


Figure 3.1: Microscope images of three letters in three documents. The last row shows that the main differences are on the borders and some areas inside the letters (with low gradient). These are the regions which we aim at proper characterizing with the Convolution Gradient Texture Filter.

are intentionally created by printer firmwares to create visual effects not perceptible by the human eye. Investigating the pattern used by firmwares of different printers is useful to identify the source printer.

In the next subsections, we discuss the proposed methods in greater details.

3.2.1 Texture Micro Patterns via Multidirectional Gray-Level Co-Occurrence Matrices

Our first solution is based on statistics of Gray-Level Co-Occurrence Matrices (GLCM), a well known micro-texture descriptor. Proposed by Haaralick et al. [2], these matrices are built by calculating how often two neighbor pixels i and j occur in a given direction and offset. Each direction will define a GLCM: West/East (0°), Southwest/Northeast (45°), South/North (90°) and Southeast/Northwest (135°). Figure 3.2 depicts these directions.

After each of these four matrices are built, a set of statistics can be calculated to describe these textures. The original paper proposes 14 measures, such as the angular second moment, contrast, correlation, sum average and so on. For each measure, there are four values. The authors proposed to use the mean and range of these four values and, finally, a 28-d feature vector is used to describe the image.

Several GLCM variations have been proposed in the literature for printer attribution. We discuss the ones proposed by Mikkilineni *et al* [68, 53, 69]. In these papers, the GLCM is calculated over a set of characters extracted from the printed document. The GLCM is calculated just in the pixels in a Region of Interest (ROI), which is the printed area of

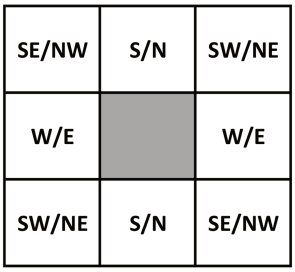


Figure 3.2: Neighboring directions used to build the four gray level co-occurrence matrices proposed by Haaralick et al. [2]: West/East (0°), Southwest/Northeast (45°), South/North (90°) and Southeast/Northwest (135°).

the rectangular region containing the letter. In these works, an offset (distance) of two pixels was used to build only one GLCM. The direction used in this case was only the pixels in the bottom side (270°). After that, 20 statistical features are calculated from the GLCM and two new metrics are proposed: the variance and entropy of pixel values in the ROI. At the end, 22 features are used in machine learning classifiers to identify the source printer. For more details, we refer the reader to Appendix A.1.

Differently from Mikkilineni et al's variation and the original GLCM, in this chapter, we start by extending the basic GLCMs to incorporate eight angles (directions) in each pixel's neighborhood, using the original image scale. The eight GLCMs are built in the following neighboring directions: East (0°), Northeast (45°), North (90°) Northwest (135°), West (180°), Southwest (225°), South (270°) and Southeast (315°). After these matrices are built, we extract the same 22 statistical measures per GLCM proposed in Mikkilineni's et al [68, 53, 69] approach. Hence, a $22 \times 8 = 176$-d feature vector is used to feed a machine learning classifier able to identify the source printer.

The multi-analysis procedure performed herein takes into account several directions (multidirectionality) represented by each GLCM, instead of few directions used by previous techniques in the literature. The rationale to do this procedure is explained by the fact that the banding artifact has an irregular behavior, as Figure 3.1 illustrates. This can be better captured by considering more directions in the texture analysis. This approach occurs in the description phase of the input questioned image, as the statistics (description) are calculated on multiple directional matrices. Figure 3.3 shows the neighboring directions used to build the proposed GLCMs.

3.2.2 Texture Micro Patterns via multiscale Multidirectional Gray-Level Co-Occurrence Matrices

Another GLCM variation proposed in this chapter bears from the idea that multiple scales of a suspected document might spread uniquely the texture found in the original scale of each printed document. We propose a GLCM texture descriptor based on Gaussian Pyramidal Decomposition of images. The *Multiscale Multidirectional Gray Level Co-occurrence Matrices* are built in the same way as the Multidirectional GLCM presented before. The difference is the use of multiple scales of the original image in a Gaussian Pyramidal Decomposition. Using s scales, an $176 \times s$ feature vector is created.

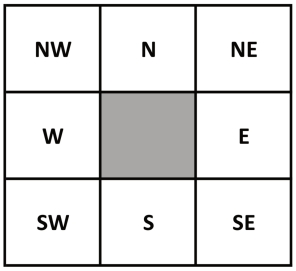


Figure 3.3: Proposed Multidirectional GLCM. We used statistics over eight matrices as printer texture signatures. Each matrix represents eight possible directions on each pixel's neighborhood: East (0°), Northeast (45°), North (90°) Northwest (135°), West (180°), Southwest (225°), South (270°) and Southeast (315°).

This approach is, in part, inspired by Siqueira et al.'s work [135], where Multiscale GLCM descriptors are proposed. The core differences of our method and theirs are: (1) we do not use dimensionality reduction in feature vectors in order to preserve texture micro patterns found in different scales; (2) the data does not need to be normalized as in their work; (3) we use more directions (eight) in the GLCM construction in order to capture more subtleties of texture micro-patterns; (4) we consider $s = 4$ scales with a very particular configuration: two downscaled versions, one upscaled version and the original scale of the image. We chose to downscale with Gaussian pyramidal decomposition because this process applies Gaussian filtering, which is known as a low-pass filter. Areas with low frequency, which are the areas with significant difference between letters from different printers (see Figure 3.1) are highlighted in the downscaled versions, and this way they are important for better laser printer discrimination. Figure 3.4 illustrates how the multidirectional and multiscale GLCMs source printer detector works. Afterwards, a classifier can be trained to identify a particular printer based on the texture of the printed material.

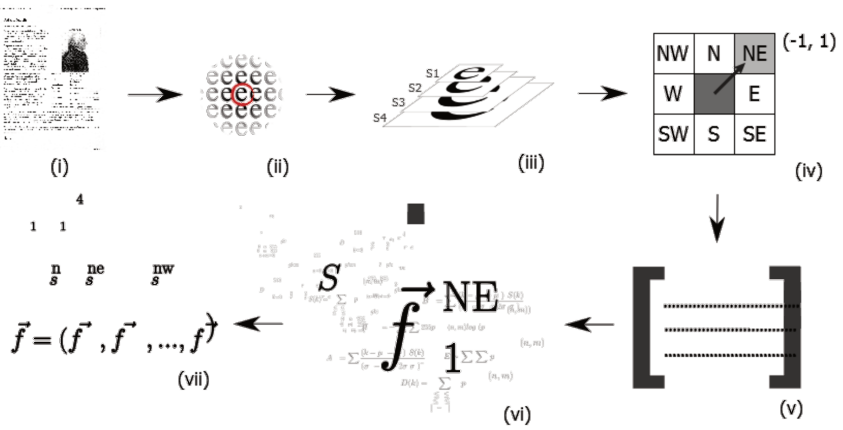


Figure 3.4: Proposed Multiscale and Multidirectional GLCM. (i) scanned document; (ii) character extraction; (iii) Gaussian pyramidal decomposition; (iv) directions used in the multidirectional approach; (v) GLCMs construction in each direction at each scale; (vi) GLCMs statistical features extracted per scale and direction; (vii) final feature vector comprising all statistics extracted across different scales and directions.

The GLCM approaches presented here are applied to inner area of printed text. These are the areas with multiple directions and scales micro texture behavior shown in Figure 3.1. The Gaussian filter of a pyramidal decomposition will emphasize these inner areas by filtering just low frequency components at each scale. We believe that the analysis of these low-level components yields a better printer attribution approach because, as can be seen on the last row of Figure 3.1, inner areas with low gradient have potential to describe printing sources, as there are differences in these low frequency areas among different printers.

The multi-analysis procedure performed herein takes into account several directions (multidirectionality) and several scales (multi-scale) represented on each GLCM, instead of few directions and single scale used in the literature. The rationale to do this procedure is explained by the fact that the banding artifact has an irregular behavior and multiple sizes, as Figure 3.1 illustrates. This behavior can be better captured by considering more directions and scales in the texture analysis. The multiscale approach occurs in the pre-processing phase of image processing and the multi-directionality is, as discussed before, done in the description phase.

3.2.3 Texture Micro Patterns via Convolution Texture Gradient Filter

Textures on almost flat areas (with small gradient value) are intentionally generated by the printer firmware by combining near pixel values below human eye resolution. These textures are created to give tonalities impression, smoothing of borders, shadows, roughness, gradient or glossy effect. On the other hand, effects of mechanical parts can produce texture patterns near printer resolution, such as motor drift, gear backlash, laser focus, mirror imperfections, drum surface defects, among others.

Our third approach for laser printer attribution relies on the analysis of these low-level gradient areas. The proposed descriptor, the Convolution Texture Gradient Filter (CTGF), aims at describing the texture of low-gradient areas. Given a labeled training set of documents, CTGF learns a set of $n \times n$ pixel patterns (texture) in low-gradient areas that appear more frequently in a given printer, but not in others.

Given the scanned printed document S with size $r \times c$, the following transformations (summarized in Figure 3.5) generate the feature vector of the proposed technique used for the learning and attribution process.

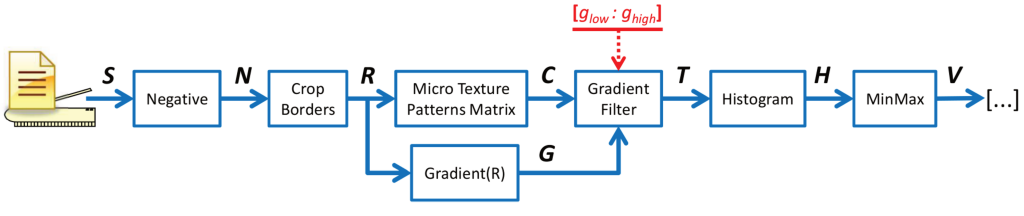


Figure 3.5: Proposed solution for laser printer attribution using the Convolution Texture Gradient Filter.

Negative As a pre-processing step, the image pixels in S are inverted. Thus, values close to zero will mean white pixels and 255, black pixels. This is made for convenience in the algorithm operations and yields a negative image N.

Crop borders In order to eliminate scanning noise at the image borders generated by external light, folding, among others, the negative image N is cropped, eliminating 6% of pixels in each border. This percentage in a letter paper document (216 $\times$ 279 mm), for example, corresponds to 12.96 mm $\times$ 16.74 mm margins, which covers areas with no printed information in typical documents. The negative cropped image is now denoted as matrix R. We still consider the dimensions of matrix R as $r \times c$ for convenience.

Micro Texture Patterns Matrix Textures with $n \times n$ neighbor pixels contained in R are then represented by two properties, which can be computed in parallel: their sum and maximum gradient between the central pixel and its neighbors. Although those two properties do not identify specific textures, they group textures of interest and allow filtering printer signatures. The convolution of R with an $n \times n$ matrix full of ones O results in the micro texture patterns matrix C

$$C = R * \mathrm{O} \tag{3.1}$$

where $*$ is the discrete convolution operator and, for the case where $n = 3$

$$\mathrm{O} = \begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix}_{3\times 3} \tag{3.2}$$

Therefore,

$$C(i,j) = \begin{cases} 0 \quad \text{if } i = 1 \text{ or } i = r \text{ or } \ j = 1 \ \text{ or } \ j = c, \\ R(i-1:i+1, j-1:j+1) * \mathrm{O}, \ \text{otherwise} \end{cases} \tag{3.3}$$

where $0 \leq C(i,j) \leq 255 \times 3^2$.

Gradient (R) In this step, we calculate the gradient of each pixel in R in a 3×3 area centered at the pixel to create the matrix of gradients G. The difference of two pixels x and y is calculated as

$$d_{x,y} = |x - y|. \tag{3.4}$$

Given the matrix R calculated previously, the gradient matrix G is calculated as

$$G(i,j) = \begin{cases} 0 & \text{if } i = 1 \text{ or } i = r \\ & \text{or } j = 1 \text{ or } j = c \\ \max\limits_{\substack{i-1 \leq p \leq i+1 \\ j-1 \leq q \leq j+1}} (d_{R(i,j),R(p,q)}) & \text{otherwise} \end{cases} \tag{3.5}$$

where $0 \leq G(i,j) \leq 255$.

Gradient filter With gradients (G) and pixel sums (C), we filter the textures with gradients of interest. Two parameters (g_{low} and g_{high}) define the range of gradient range of textures that identify discriminant features for printer signature. Such parameters are selected from a training set of documents per suspected printer (which we will discuss in section 6.3.4) for maximum results on the learning process. The matrix T of texture codes (sums) is then created by filtering textures that are not in the defined range. Those textures are the discriminant positions in the printed document. T is calculated according to Equation 3.6.

$$T(i,j) = \begin{cases} C(i,j) & \text{if } g_{low} \leq G(i,j) \leq g_{max} \\ 0 & \text{otherwise} \end{cases} . \tag{3.6}$$

Histogram Counting the number of positions for each texture in T from one (zero represents a position with no considered texture and is not used in the histogram) to $255 \times n^2$ generates the histogram vector with $255 \times n^2$ bins, as shown in Equation 3.7.

$$\mathbf{H} = \text{Histogram}(T, 1 : 255 \times n^2). \tag{3.7}$$

MinMax the final feature vector $\mathbf{V}$, which represents the histogram of low-level gradient textures that a printer prints in the document is generated by applying a MinMax normalization on the histogram $\mathbf{H}$, scaling the components to the interval $[0, 1]$, as Equation 3.8 shows.

$$\begin{aligned} u &= Min_{\mathbf{H}(j)}(\mathbf{H}), \\ v &= Max_{\mathbf{H}(j)}(\mathbf{H}), \\ \mathbf{V}(j) &= \frac{\mathbf{H}(j) - u}{v - u}. \end{aligned} \tag{3.8}$$

As the final feature vectors are histograms of sums of pixels, they have $255 \times n^2$ dimensions, where n is the dimension of a squared sliding window used to calculate the texture.

This new method is based on $n \times n$ neighboring textures working with two basic properties: (1) sum of pixels; and (2) gradient filtering. The sum of pixels, obtained by a convolution with an $n \times n$ kernel of ones, measures the grayscale tone related to the visual impression of this region. The gradient is used to separate flat areas on text and images from the borders, as edge pixels have larger gradient than the interior of letters and background areas. Although those two properties cannot uniquely identify textures, they group textures of interest when used together, and also allow filtering printer signatures. Figure 3.6 depicts how texture values vary for the same text and picture printed on different printers.

The multi-analysis procedure performed herein considers several directions (multidirectionality) in the filtering (textures used in the histogram are created by filters that consider all neighbors in the filtering procedure) and, if several filter sizes are used, there are several scales (multi-scale) filters. The reason for that is that the multidirectional

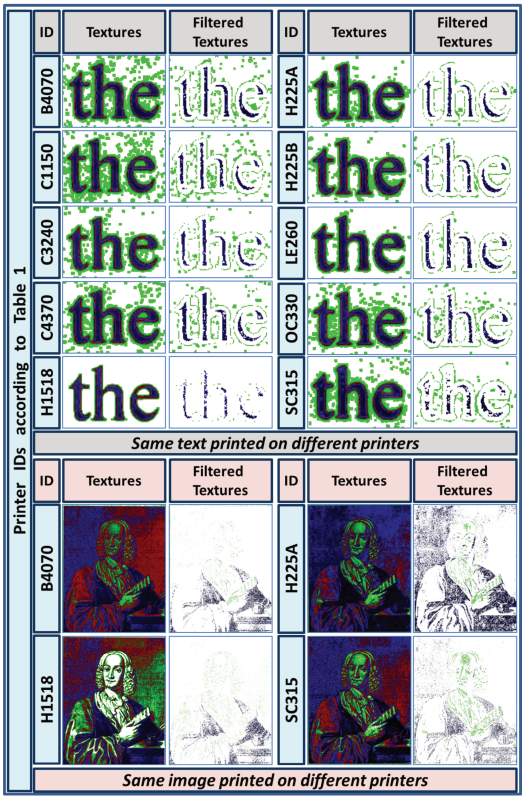


Figure 3.6: Textures and filtered textures by gradient filter of (a) text and (b) image from different printers. White=0, Green [1:765], Red=[766:1530] and Blue=[1531:2295].

convolution explores texture patterns with gradients of interest that can better represent the banding, as the pre-processed image used for the histogram construction considers the banding that can happen in all directions. The multi-analysis here is considered in the description step, as the pixels of the transformed image are used in the histogram (feature vector) that feeds the machine learning classifier at the end.

3.3 Experimental Setup

In this section, we present the dataset and methodology used in the experiments. We discuss the experimental scenario, which parts of scanned documents are used in our investigation, metrics and how we implement the proposed methods and the state-of-the-art methods used for performance comparison.

3.3.1 Dataset

To validate the proposed techniques and compare them to the ones from the literature, we decided to use a dataset projected and built to provide instances of scanned documents as close as possible to a real situation. The databases used in prior works are limited in some way because they always consider fonts of same size and style, some of them have only text or only figures and some expect that the scanned documents are available in very high resolutions. Hence, the datasets in some prior works do not consider the case

where the original document is not available to be scanned in very high resolution. This can possibly affect these approaches performance. In addition, for the works we surveyed, the used datasets are not readily available for download hardening comparisons using the same setups.

The aforementioned issues do not happen in the proposed dataset. We printed all or some of the 120 documents on ten LPs (showed on Table 3.1) in standard resolutions with Chamex white letter paper on $75g/m^2$ granularity, yielding 1,184 TIFF images. These images are printable versions of Wikipedia documents converted to pdf with one, two or three pages and contain different letter sizes, fonts and figures. These documents were later scanned by a reference scanner (Plustek SO PL2546) at 600 dpi resolution and are separated by two factors: Language (English or Portuguese) and Figures (With or Without).

#	**Printer ID**	**Manufacturer**	**Laser Printer Model**	**Number of Printed Documents**
1	B4070	Brother	HL-4070CDW	120
2	C1150	Canon	D1150	116
3	C3240	Canon	MF3240	120
4	C4370	Canon	MF4370DN	120
5	H1518	Hewlett Packard	CP1518	120
6	H225A	Hewlett Packard	CP2025A	119
7	H225B	Hewlett Packard	CP2025B	110
8	LE260	Lexmark	E260DN	119
9	OC330	OKI Data	C330DN	120
10	SC315	Samsung	CLP315	120
			Total	**1,184**

Table 3.1: Printers and number of documents per printer used in the experiments.

3.3.2 Methodology

In this section, we discuss the background of the experimental methodology outlining the used regions of interest considered in each document, the metrics used, and implementation details about each method.

The techniques used in the experiments follow the pipeline presented in Figure 3.7. Classifiers are trained with feature vectors yielded by different description techniques after the documents are printed and scanned at 600 *dpi*. Given one scanned printed document for testing, the classifier predicts its class. We have used the Support Vector Machines Classifier [136] with linear kernel in this process.

We used the one against one implementation of Support Vector Machines for multiclass problems. This approach works by building a set of $\frac{c(c-1)}{2}$ binary classifiers, where c is the number of available classes. Each of these classifiers will use data from each unique pair of classes. Then, at the end of the classification step, a voting strategy is performed. Each result of each binary classifier is considered a vote and the class with the maximum number of votes will be the classification of the given sample.

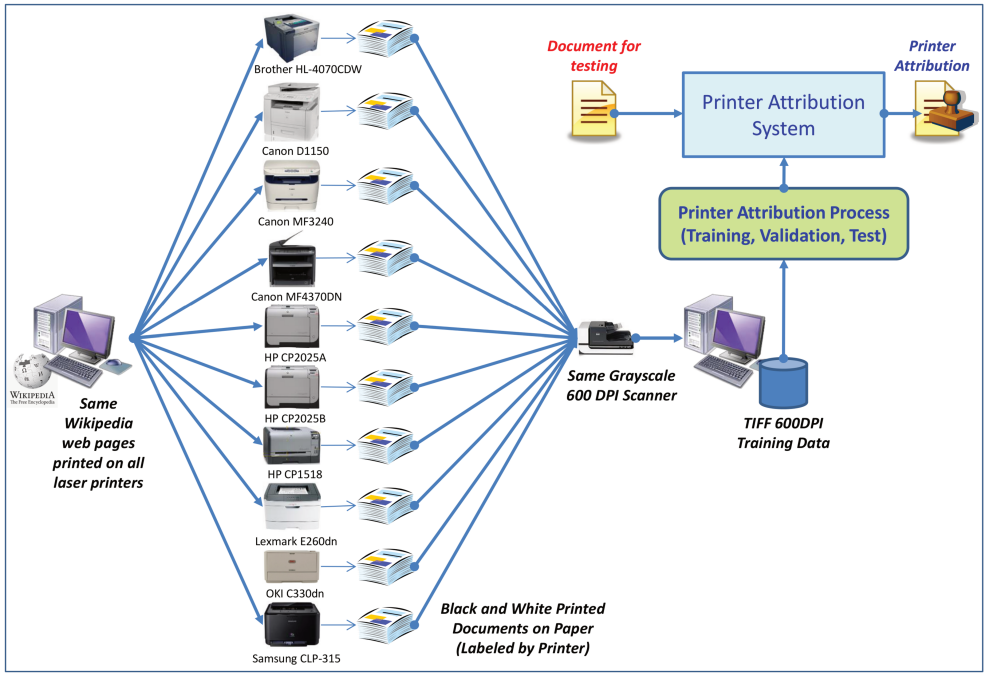


Figure 3.7: Workflow used in the experiments. Firstly, the documents are printed by different printers and scanned. After that, a classifier is trained on feature vectors created through the different description techniques. Given an investigated document, the classifier will predict its class based on the trained models.

3.3.3 Sampling Approaches

To study the effects of document sampling on the printer attribution process, we will discuss here the analysis on different areas of documents. It includes characters, frames and the whole document.

Characters As texture analysis implies in the investigation of printed areas and their interactions with paper borders, we start by extracting specific letters from the digital versions of documents. These letter images will be the dataset used in the experiments. To extract all letters from a document, we first implemented an algorithm that searches for connected components in graphs. Using thresholding and considering the neighboring pixels as graph nodes connected by neighbors, we extract useful areas on the image to capture how the letters look like.

To distinguish and separate the characters with higher occurrence (with same size and same font), we used a descriptor similar to Local Binary Patterns [137]. This descriptor separates a black and white version of the given image on slices of a superposed imaginary circle, describing them by counting the number of white and black pixels on each slice. The final descriptor is the counting of black and white pixels or ratios as black and white pixels density among others. This algorithm, when fed with a reference character as input, can separate with high hit rates the letters from the rest of extracted connected components. For this part, we focused on letters “e” as it is the most common letter in English documents and were also used before in the literature [85, 68, 53, 69]. The final letter dataset has 245,000 extracted characters.

To classify the source of a given investigated printed document using this approach, the letters "e" are firstly extracted. Then, each letter is classified by a printer attribution method (e.g., the ones discussed in this chapter), and a majority voting is applied in the end. The occurrence of each labeled class is counted on these letters and the most voted class will define the class of a document.

Frames A letter paper, scanned in grayscale at 600dpi without compression, produces a very large file with approximately 31 MB of size, corresponding to about 5K by 6.6K pixels, even after discarding 6% on each border of the document corresponding to blank margins carrying external light scanning noise. After cropping, the remaining number of pixels is still very large (about 4.4K by 5.8K pixels). There are also areas inside the document that are completely blank (without printed ink). As those areas do not contribute with information about the printer, it is useful to split the large document in smaller samples, which maintain printer characteristics and can generate more feature vectors for the training and testing learning process.

In previous works [85, 68, 53, 69], character samplings were proposed to capture texture behavior on printed documents. Letters "e", which is the most used letter in English texts, are extracted in each document. A mask of a template letter "e" is used to scan, compare and cut its copies from the document, capturing its pixels. The typical letter "e" in documents are inside an area of 40×50 pixels. This process is normally time consuming and not very accurate.

In this chapter, we propose to use chunks of letters in regions of interest from a document, which we call *frames*. Frames are rectangular areas inside the document that have sufficient printed material to keep the characteristics of a printed document. The process used to obtain frames from the cropped images with 4.4K by 5.8k pixels consists of dividing them in five columns by six rows of frames, resulting in about 900 by 980 pixels corresponding to 37 mm (1.5") by 43 mm (1.6").

In order to avoid frames that do not contain enough printed areas, we state that the minimum accepted ratio between dark pixels (black and dark gray) and blank ones (blank and light gray) should be 0.02. This process eliminates frames that are completely blank or have only a few printed symbols on it. For reference, Figure 3.8 shows the same document sampled by frame and character.

Document To consider the scenario with just one frame and to evaluate the methods when not using any kind of voting scheme, we have also proposed an approach based on the whole document. Although there are several ways of describing a document using only one feature vector (e.g., each character GLCM can be accumulated to yield a single GLCM, from which one single feature vector can be extracted), we decided to apply the texture descriptors on the whole document, similar to some state-of-the-art techniques [87, 84, 72].

3.3.4 Metrics and Statistics

We adopt 5×2 cross validation protocol. Using this approach, five replications of the two-fold cross-validation protocol are performed. In each one, a set $\boldsymbol{X}$ is randomly

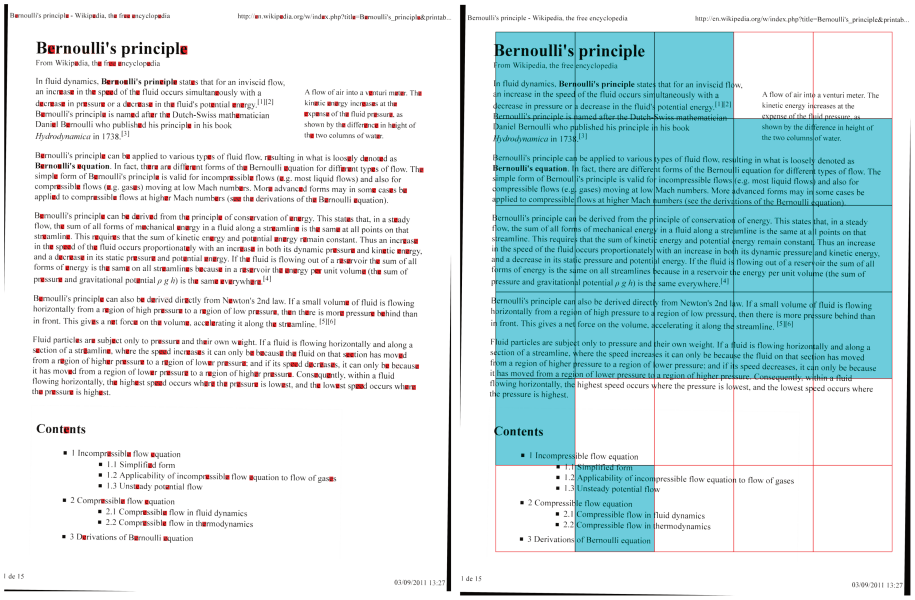


Figure 3.8: Letter (left) and frame (right) sampling from a scanned document. The red areas identify the extracted letters while the cyan areas identify the extracted frames.

divided into $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ and a classifier is trained with $\boldsymbol{X}_1$ and tested on $\boldsymbol{X}_2$. Thereafter, training/testing sets are switched and the process repeated. There are $5 \times 2 = 10$ different executions in the end. This is considered an optimal benchmarking protocol for learning algorithms [138].

We use a set of known metrics to assess all the algorithms performance using the above cross validation approach. In a multi-class problem with c classes, a confusion matrix M is built with c rows and c columns on each round of the 5×2 cross validation. Main diagonal values of M will show the right hits for each class. Other values are false hits

$$accuracy = \frac{\sum_{i=1}^{c} M(i,i)}{\sum_{i=1}^{c} \sum_{j=1}^{c} M(i,j)} \tag{3.9}$$

The precision of a given class (in this case, a printer) i, is defined as the fraction of events where the classifier *correctly* classified i out of all instances classified as being from that class

$$\boldsymbol{Precision}(i) = \frac{M(i,i)}{\sum_{j=1}^{C} M(j,i)} \tag{3.10}$$

The recall of a given class i is the fraction of events where the classifier correctly classified i out of all instances of that class

$$\boldsymbol{Recall}(i) = \frac{M(i,i)}{\sum_{j=1}^{C} M(i,j)} \tag{3.11}$$

The *f-measure* of a given class i considers both the precision and recall in the analysis. It can be interpreted as the harmonic mean of precision and recall, where it reaches its

best value at 1 and worst score at 0

$$\boldsymbol{f}(i) = 2 \cdot \frac{\boldsymbol{Precision}(i) \cdot \boldsymbol{Recall}(\mathrm{i})}{\boldsymbol{Precision}(i) + \boldsymbol{Recall}(i)} \tag{3.12}$$

We perform a series of statistical tests to define if the results are statistically significant. First, we confirm if all techniques are statistically different (also known as pre-test). If they are, we check the techniques pairwise to define which ones are statistically different when compared to other (also known as post-test). Each of these steps usually involves a statistical test and a confidence level for the test. Here we consider a confidence level of 95% for each test. As pre-test, we consider the Friedmann test. This test is non-parametric and is used to determine if subjects change significantly across occasions and conditions. To compare the techniques pairwise (also known as multi-compare approach), we use the Tukey-Kramer approach (also known as Honestly Significant Difference (HSD).

3.3.5 Baselines

We compare our proposed techniques against four state-of-the-art methods (presented in section 2.1.2) and also against two well-known texture descriptors widely used in content-based image retrieval applications.

The first state-of-the-art technique uses Gray-Level-Co-Occurrence Matrices (which we call GLCM) applied to laser printer attribution, proposed by Mikkilineni et al. [68, 53]. This technique describes the neighborhood behavior of pixels in a two-dimensional histogram given an offset, yielding one GLCM in which 22 statistics are calculated. The original GLCM of Mikkilineni et al. [68, 53] uses an offset of $dr = 2$ and $dc = 0$ (dr stands for the offset in the rows while dc stands for the offset in the columns). In our implementation, we used dr in the interval $1 \leq dr \leq 3$ and we found the best as $dr = 1$. This is explainable because the Regions of Interest in our database are smaller than the ones in [68, 53] approach. Although this technique was originally proposed to operate on characters, we also evaluate its performance on documents and frames directly. The 22 statistics extracted from the GLCM are discussed individually on Appendix A and are also used in our proposed GLCM variations.

The second considered method is based on statistics of Discrete Wavelet Transform (which we call DWT_STATS) from color bands applied to laser printer attribution, proposed by Choi et al. [76]. In this implementation, 39 statistical features are extracted from the HH Discrete Wavelet Transform sub-band per image. This approach is also applied document-, character-, and frame-wise.

The third method evaluated was the statistics of printer noise (which we call NOISE_STATS) in the row and column direction by Elkasrawi and Shafait [72]. This technique, based on a previous work of Khanna et al. [78] on scanners, works by first filtering the printed area with Otsu's threshold [79]. By binarizing the image with this threshold, the authors compute the median gray-level for the foreground as well as the median gray-level for the background pixels. Hence, a clean image is generated by only having gray-level values of all foreground and background pixels set to the median foreground and background calculated. The noise image is then obtained by subtracting the

original image from the clean image. The mean of rows and columns of this noise reference image is calculated, yielding two vectors: the correlation between each row of the noise image and the average of all columns, as well as the correlation between each column and the average of all rows. Finally, a set of 15 statistics are calculated over these vectors. This approach is evaluated document-, character-, and frame-wise.

The last state-of-the-art method implemented was the technique proposed by Kee and Farid [85] (which we call RECONST_ERROR). This technique has three steps: pre-processing, printer profiling and source identification. In the pre-processing step, the authors first choose a reference character (they chose the letter 'e'). Then, same letters are searched by template matching, preprocessed by histogram normalization and registered with the reference letter. In the printer profiling step, the mean character $\bar{c}$ per printer is calculated and Principal Component Analysis (PCA) [86] is performed on these aligned characters per printer. The printer profile are the PCA top p eigenvectors e_i, $i \in [1, p]$ and the mean character. In the source identification step, a test document is given, its letters 'e' are extracted and preprocessed the same way. These letters are used with the top p eigenvectors and mean character per printer to calculate a *reconstruction error* of each printer. The smallest mean error will identify the source of a printed document. This is the only method in the literature that does not use a known machine learning classifier. Therefore, we consider, in the 5×2 cross validation, the printer profiling phase as the training and the source identification as the testing step. This approach is only applied on characters.

In addition to the state-of-the-art methods considered herein, we also assess two well-known texture descriptors widely used in the literature. The first one is the Local Binary Patterns (LBP) [137] and the Histogram of Gradients [139] (HOG). The LBP is a histogram of eight-neighboring pixel relations. HOG consists in histograms of gradient orientations in localized regions (rectangular or circular) of an image. We chose these descriptors because they can be regarded as multidirectional descriptors.

3.3.6 Implementation Aspects of the Proposed Methods

We first consider the two proposed GLCM variations: the multidirectional and multidirectional and multiscale ones. We also implement four CTGF variations, three exploring 3×3, 5×5, and 7×7 filter sizes and a multiscale one exploring all the previous filter sizes.

The multidirectional GLCM hereinafter referred to as GLCM_MD consists of 22 statistics calculated on each GLCM built using one neighboring pixel offset over eight directions, as described in section 3.2.1. The final feature vector has $22 \times 8 = 176$ dimensions. The multidirectional/multiscale (GLCM_MDMS), in turn, consider four scales of the image Gaussian pyramidal decomposition: the original scale, two down-scales and one up-scale. The final feature vector lies in the $176 \times 4 = 704 - d$ space.

CTGF is built as described on section 3.2.3 and yields feature vectors with $3^2 \times 255 = 2,295$ ($n = 3$), $5^2 \times 255 = 6,375$ ($n = 5$) and $7^2 \times 255 = 12,495$ ($n = 7$) dimensions. We also evaluate a combined approach, in which we consider the different scales in a combined form creating what we call the *Multiscale CTGF* (hereinafter referred to as CTGF_MDMS), with $2,295 + 6,375 + 12,495 = 21,165$ dimensions. These feature vectors

undergo dimensionality reduction on each filter window size as we shall discuss later in this chapter.

Finally, we test the complementarity of the proposed methods by fusing the feature vectors from the CTGF using the 3×3 mask and GLCM_MDMS, creating what we call the CTGF_GLCM_MDMS.

3.4 Results and Discussion

We now turn our attention to the actual experiments and results. We start with a study on dimensionality reduction for the CTGF method as one could wonder if all its features are really necessary for attribution. Then, we present the experiments for all methods considered herein followed by a proper statistical analysis of the results.

3.4.1 Convolution Texture Gradient Filter Parameters and Dimensionality Reduction

The main parameters of CTGF method are (g_{low}, g_{high}), which are defined during training. For that we consider the 5×2 cross validation protocol discussed in section 4.4.2.

We performed two experiment configurations: keeping g_{low}=1 and varying g_{high} from 1 to 128 and varying g_{low} from 1 to 128 and keeping g_{high}=254. This way, in the first interval, we consider intervals with low and medium gradient values (from g_{low}=1 to g_{high}, where $1 \geq g_{high} \geq 128$), and in the second interval we will consider intervals with low, medium and big gradient values (from $1 \geq g_{low} \geq 128$ to g_{high}=254). This configuration was also done to verify what happens in one gradient interval and also in the rest of the interval, as can be seen in the results shown on Figure 3.9. For this experiment, we used the one-against one multiclass SVM with linear kernel and the CTGF filter window size was set to 3×3. The experiments were performed frame-wise.

Figure 3.9 shows that keeping g_{low} as 1 and reducing g_{high} from 128 to 16 (solid blue line) produced very close results on the 5×2 cross-folding validations. In addition, in such situations, the classification differences are not statistically significant according to Friedman statistical tests. The best results of the experiments performed are in the interval which included gradient values over the interval (1,32), and this confirms the suspicion of relevant printing information in low level gradient areas. This is because of grayscale jitters (*i.e.,* grayscale noise) on flat black and white areas of printed document due to printing variations (positioning, backslash, toner development, *etc.*). It is also important to understand that variations of gradient in the range (1,32) on grayscale neighbor pixels are practically undetectable at normal resolution (600 dpi) for the human eye. Filtering texture values by a convolution window in this gradient interval around flat color areas creates a highly discriminative noise signature.

The result of (g_{low}, g_{high}) filtering by the proposed technique may result in some components that are not significant for the attribution process and a dimensionality reduction approach can be applied. We use a simple dimensionality reduction method that discards dimensions where the distance between its maximum and minimum values

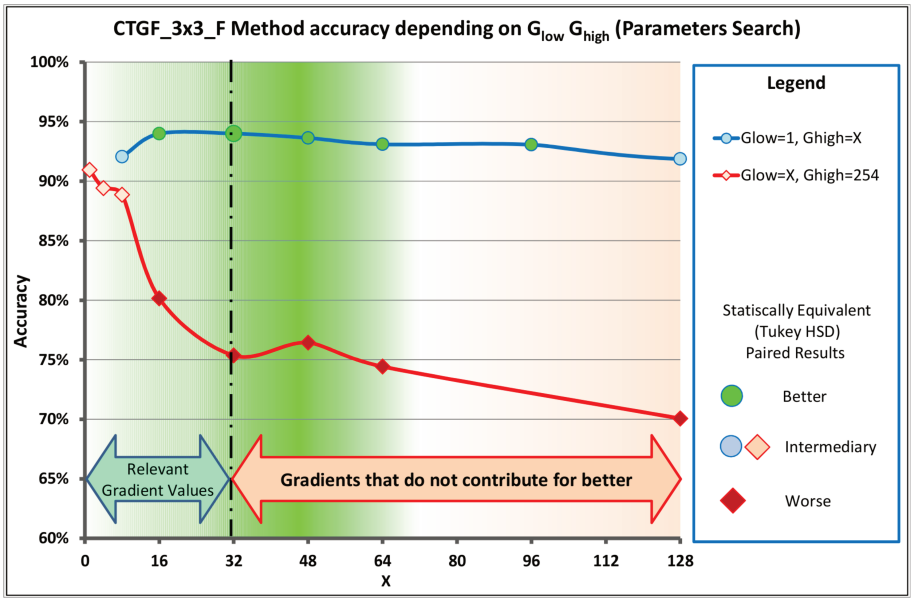


Figure 3.9: Filter parameter search for the proposed Convolution Texture Gradient Filter.

(also known as range) is less than the mean of the overall components distance, calculated during training. This yields a binary vector (***KeepVector***), which is used to eliminate or keep features from feature vectors used for classification.

To find the vector ***KeepVector***, we describe a training set of documents with CTFG and put all the feature vectors $\boldsymbol{V}$ in a matrix F. Then, we calculate the mean of components range (component or dimension here can be seen as each column of matrix F). The range of a component is defined as the subtraction between the maximum and minimum value of that component (or column of matrix F). After this, we calculate the mean and build a binary vector ***KeepVector*** with $255 \times n^2$ dimensions. This vector is used in the feature vector construction, indicating whether a dimension in a new feature vector will be kept (its range is higher or equal the mean of dimensions range of matrix F) or not. ***KeepVector*** is built as Equation 3.13 shows. Figure 3.10 shows an example of the feature vector dimension reduction process and Figure 3.11 shows the mean reduced feature vectors of some printers used in this work.

$$\boldsymbol{KeepVector}(i) = \begin{cases} 1 \text{ if } Max_{F(:,i)}(i) - Min_{F(:,i)}(i) \geq \dfrac{\sum_{i=1}^{255\times n^2} Max_{F(:,i)}(i) - Min_{F(:,i)}(i)}{255 \times n^2} \\ 0 \text{ otherwise} \end{cases} \tag{3.13}$$

To validate the proposed dimensionality reduction technique, we also conducted experiments using 5×2 cross-validation experiments comparing it to PCA. The PCA results are not as good as the ones obtained with the aforementioned method.

The principal components of the CTGF histogram are more related to the structure of the image pixels than the intrinsic noise used to differ printers. Therefore, PCA ends up discarding components otherwise useful for printer attribution, that is why it does not perform so well in comparison with the feature selection method discussed above.

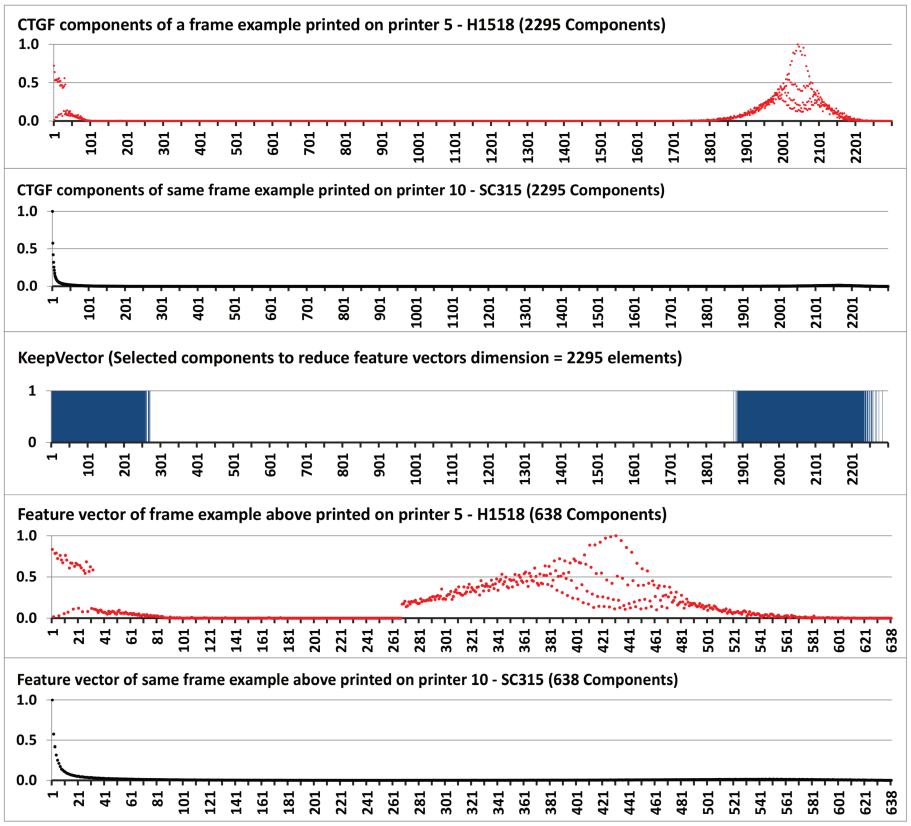


Figure 3.10: Feature vector reduction process. The first and second rows show examples of feature vectors calculated using the CTGF approach with a 3×3 filter size. These feature vectors were calculated on the same document subset (which we call frames) printed by two different printers. The third row shows the binary vector ***KeepVector***, in which the colored regions indicate what dimensions from the 2,295 must be kept. The fourth and fifth rows show the 638 dimensional reduced feature vectors from the two printers showed on first and second rows.

Table 3.2 shows the comparison of the dimensionality reduction methods tested.

Method CTGF_3x3_F Dimensionality Reduction Test Results	*Feature Vector Size (Original 2295)*	*Accuracy (%)*			
		Min	*Mean*	*Max*	*StDev*
Component (Max-min) >= Mean (Max-min) (*)	***638***	93.17	**94.44**	96.59	1.03
PCA Sum of Eigenvalues = 99.9% (**)	***311***	90.29	**92.82**	95.06	1.48
PCA Sum of Eigenvalues = 100%	***2281 (***)***	89.08	**91.53**	93.19	1.38
PCA Sum of Eigenvalues = 99%	***40***	88.06	**90.49**	93.17	1.80
PCA Sum of Eigenvalues = 95%	***2***	33.56	**35.04**	36.69	0.96
PCA Sum of Eigenvalues = 90%	***1***	20.95	**23.43**	26.07	1.54

(*) Means that each selected component has Max-min distance >= Mean of overall component Max-min distances.
(**) Sum of Eigenvalues = x means that eigenvalues sum of selected components does not exceed x.
(***) Components wich Eigenvalues = 0 are discarded.

Table 3.2: Comparison of the proposed dimensionality reduction approach with some PCA variations. We used $n = 3$ for the CTGF filter size, in which the standard feature vector has 2,295 dimensions.

As Table 3.2 shows, the proposed dimensionality reduction technique selected the most important dimensions of feature vectors for classification, achieving a mean accuracy

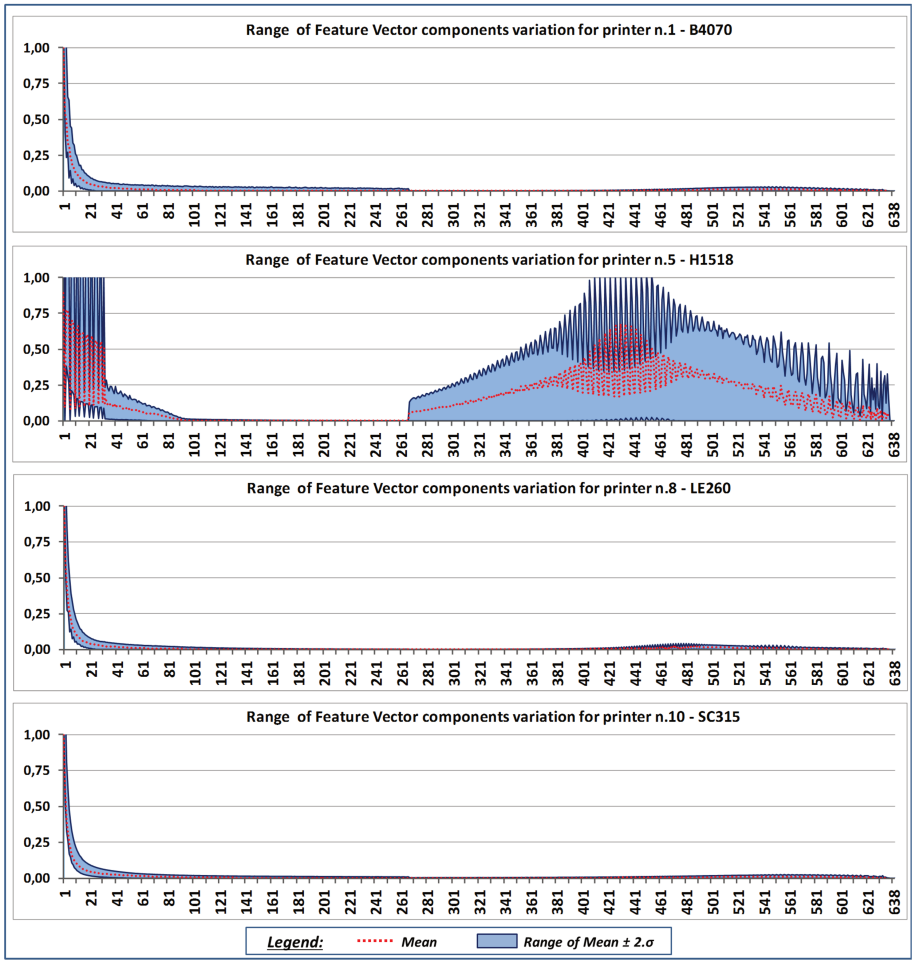


Figure 3.11: Printer signatures of some printers using the proposed CTGF with a 3×3 filter size and the proposed dimensionality reduction approach.

of 94.44%, against the best PCA best configuration, which achieved a 92.82% mean accuracy considering the cross-validation procedure adopted. The results are explainable as the proposed method eliminates dimensions with small variation in the training stage, not performing any additional linear transformation on the data. This dimensionality reduction approach, when applied in CTGF on frames, keeps 638 dimensions of feature vectors for the CTGF with a 3×3 mask, 2,660 dimensions of feature vectors for CTGF with 5×5 mask, 6,672 dimensions of feature vectors for the CTGF with 7×7 mask and 638+2,660+6,672=9,970 dimensions for the multidirectional and multiscale CTGF. When applied on documents, it keeps 471 dimensions of feature vectors for the CTGF with 3×3 mask, 2,347 dimensions of feature vectors for the CTGF with 5×5 mask, 4,842 dimensions for the 7×7 mask and 471+2,347+4,842=7,660 dimensions for the multidirectional and multiscale CTGF.

3.4.2 Laser Printer Attribution Experiments

With the dataset presented in section 3.3.1 and methodology described in section 3.3.2 in mind, we now discuss the experimental results, whereby we validate the proposed approaches against the state-of-the-art methods. Table 3.3 shows experimental results

considering the the 5×2 cross-validation protocol. We applied the techniques on characters, documents and frames as described in section 3.3.3.

Rank	Method	Accuracy Statistics on Crossfolding 5x2 Experiments					
		Mean	σ	Mean-2σ	Mean+2σ	Min	Max
1	***CTGF_GLCM_MDMS_F***	**98.47**	**0.60**	**97.26**	**99.67**	**96.93**	**99.15**
2	***GLCM_MDMS_F***	**98.38**	**0.72**	**96.95**	**99.81**	**97.10**	**99.32**
3	***GLCM_MD_F***	97.15	0.84	95.47	98.84	95.40	98.30
4	***GLCM_MDMS_C***	**97.60**	0.72	**96.15**	**99.05**	**96.63**	**98.99**
5	***GLCM_MD_C***	96.99	0.94	95.12	98.87	95.78	98.82
6	LBP_F [138]	95.21	**0.59**	94.03	96.39	94.22	96.25
7	HOG_C [140]	95.79	0.83	94.14	97.45	94.42	96.79
8	***CTGF_3x3_F***	94.44	1.03	92.38	96.50	93.17	96.59
9	***CTGF_MDMS_F***	94.31	1.40	91.51	97.10	91.48	95.74
10	GLCM_F [69,54]	93.62	1.13	91.36	95.89	91.82	95.23
11	GLCM_C [69,54]	94.19	1.36	91.47	96.91	92.24	96.45
12	***CTGF_GLCM_MDMS_D***	91.81	1.47	88.86	94.76	89.38	94.42
13	LBP_C [138]	90.20	1.22	87.77	92.64	88.16	91.71
14	***CTGF_3x3_D***	90.44	1.35	87.73	93.15	88.03	93.26
15	***GLCM_MD_D***	89.31	***2.28***	84.75	93.87	84.32	92.72
16	***GLCM_MDMS_D***	88.58	1.58	85.43	91.74	86.34	90.69
17	LBP_D [138]	88.08	1.50	85.08	91.07	86.17	90.19
18	***CTGF_5x5_F***	87.78	1.67	84.44	91.11	85.32	90.46
19	RECONST_ERROR_C [86]	84.87	2.09	80.69	89.04	81.79	87.82
20	***CTGF_MDMS_D***	88.45	1.38	85.69	91.20	86.51	90.36
21	***CTGF_7x7_F***	83.80	2.09	79.62	87.98	80.89	86.50
22	***CTGF_5x5_D***	84.80	1.38	82.04	87.56	82.29	87.02
23	***CTGF_7x7_D***	83.85	1.94	79.97	87.73	81.56	88.36
24	GLCM_D [69,54]	82.57	***2.68***	77.21	87.93	78.75	87.31
25	HOG_D [140]	79.66	1.37	76.92	82.41	78.00	81.90
26	HOG_F [140]	74.36	0.81	72.74	75.97	72.74	75.47
27	NOISE_STATS_C [73]	68.87	1.10	66.68	71.06	67.29	70.66
28	DWT_STATS_D [77]	***36.57***	1.94	***32.68***	***40.46***	***33.00***	***39.93***
29	DWT_STATS_F [77]	***34.34***	1.81	***30.73***	***37.95***	***32.08***	***37.65***
30	NOISE_STATS_F [73]	42.27	1.12	40.03	44.51	40.48	43.76
31	NOISE_STATS_D [73]	39.82	1.50	36.82	42.82	37.27	42.13
32	DWT_STATS_C [77]	***28.70***	***3.19***	***22.32***	***35.07***	***24.96***	***33.56***

Legend:
xx.xx = Three best methods in the column metric
xx.xx = Three worst methods in the column metric

Table 3.3: Mean Accuracies of 5×2 cross validation applying the proposed and state-of-the-art techniques on characters (C), frames (F) and Documents (D). The proposed techniques in this chapter are the ones in bold in the column "Methods".

As expected, the worse experiment was DWT_STATS [76]. This happens because this technique was proposed for color documents, operating on RGB and CMYK color bands. NOISE_STATS [72] showed its best accuracy (68.86%) for characters. In this case, the printer noise is extracted from the letters in a region with small background perturbation. For frames and documents, this technique showed worse results (42.26% and 38.81%), as a large background area is considered hardening the noise estimation.

The RECONST_ERROR [85] was proposed to work only in characters, then we applied this proposed technique only in the extracted text letters and it yielded a classification accuracy of 84.86%.

GLCM [68, 53] was the state-of-the-art method which yielded the best results. Although it was originally proposed to operate on characters, on frames and documents it also showed decent classification accuracies (93.62% and 82.56%, respectively). On characters, it yields the best classification accuracy for a method proposed in the literature: 94.19%.

The LBP [137] and HOG [139] approaches are general-purpose texture descriptors but they also showed decent classification results. HOG yielded a 95.79% accuracy for characters, 74.35% for frames and 79.66% for documents. LBP yielded 90.20% classification accuracy for characters, 95.20% for frames and 88.07% accuracy for documents. These good results have a reason: HOG uses a histogram of gradients, hence it identifies the printing process artifacts between the text and background (borders). LBP uses a histogram of relations between a pixel and its neighbors that also enables the identification of printer patterns in a muldirectional way.

In this chapter, we propose to look beyond these simple texture approaches and analyze the multidirectional and multiscale properties of textures from printed documents. As Figure 3.1 depicts (section 3.2.3), by investigating printed letters in a microscope, we can see that the texture is spread over multiple directions. Hence, as expected, the GLCM_MD showed good classification results, 96.99% for characters, 97.15% for frames and 89.30% for documents. In addition, when considering the multidirectional and multiscale properties of texture patterns at the same time, GLCM_MDMS, the method yields the best result for characters: 97.60%. For frames, it also yielded a very good classification accuracy: 98.38%. For documents, it yielded an accuracy of 88.58%.

The proposed CTGF approaches were used here with filter sizes of 3×3, 5×5 and 7×7. These filters, when used individually, analyze the histograms of textures of low-level gradients. These textures are calculated on a neighborhood given by the filter size. These descriptors can be regarded as multidirectional filters. The CTGF with 3×3 filter size yielded accuracies of 94.44% for frames and 83.78% for documents. The 5×5 CTGF filter size yielded accuracies of 87.77% for frames and 80.28% for documents. Finally, the 7×7 CTGF filter size yielded accuracies of 83.80% for frames and 76.90% for documents. The multidirectional and multiscale approach in CTGF results in accuracies of 94.19% for frames and 88.45% for documents.

The fusion of CTGF with the GLCM uses the complementarity of both techniques. We combined the best proposed CTGF technique (CTGF_3x3) and the best multidirectional and multiscale technique (GLCM_MDMS). This last technique better explores the printing patterns more apparent between the printed material and background while CTGF explores micro-textures in regions of low gradient. This fusion yielded the best result of the experiment: a remarkable 98.47% classification accuracy for Frames. This means a 69% reduction of error from the best state of the art considered: LBP on frames (LBP_F). We also tried this fusion considering the entire document other than on frames and it was not as effective: 91.81%.

Our second discussion on the experiments results is about how the techniques behave on the classification for each printer. For that, we show on Table 3.4 the f-measure as percentages.

As Table 3.4 shows, the multidirectional approach used by LBP is useful to identify the texture patterns of printer B4070 in frames, showing an f-measure of 100%. The voting approach and high presence of texture in the printed material explain the high f-measure for this technique for that printer. The f-measure from the proposed multidirectional and multiscale approaches (GLCM_MDMS and GLCM_MD) and the fusion of CTGF and GLCM_MDMS (CTGF_GLCM_MDMS) also present a high f-measure for this

Rank	Method	Mean f-measure by Printer on Crossfolding 5x2 Experiments									
		B4070	C1150	C3240	C4370	H1518	H225A	H225B	LE260	OC330	SC315
		1	2	3	4	5	6	7	8	9	10
1	CTGF_GLCM_MDMS_F	99.76	98.78	98.82	99.57	98.90	94.59	94.76	99.12	100.00	99.92
2	GLCM_MDMS_F	99.24	99.04	99.16	99.24	98.73	94.22	94.45	99.41	100.00	99.92
3	GLCM_MD_F	98.51	98.08	97.44	97.90	98.89	90.22	90.70	99.25	100.00	99.92
4	GLCM_MDMS_C	99.59	95.45	99.01	99.03	94.67	94.36	94.19	99.68	100.00	99.59
5	GLCM_MD_C	99.60	95.02	97.95	97.59	94.14	92.69	92.92	100.00	100.00	99.51
6	LBP_F [138]	100.00	97.25	98.15	99.41	97.22	82.06	77.46	99.21	100.00	99.67
7	HOG_C [140]	95.24	92.63	97.44	98.28	93.74	91.58	91.33	97.41	100.00	99.51
8	CTGF_3x3_F	97.85	96.51	89.59	91.90	95.35	87.89	89.31	96.23	99.65	99.50
9	CTGF_MDMS_F	98.08	93.43	93.20	93.65	96.43	84.72	88.11	96.78	98.64	99.59
10	GLCM_F [69,54]	97.23	87.65	91.95	96.13	95.32	84.12	86.44	97.58	99.17	99.92
11	GLCM_C [69,54]	97.90	89.55	90.54	94.94	93.35	88.27	90.69	96.26	100.00	99.58
12	CTGF_GLCM_MDMS_D	97.30	89.65	88.87	92.47	96.22	82.87	82.25	93.82	95.66	98.02
13	LBP_C [138]	98.86	92.22	94.50	95.94	93.61	73.84	49.39	94.71	99.83	99.43
14	CTGF_3x3_D	91.48	90.26	87.40	85.20	92.30	88.21	85.95	89.98	95.48	97.83
15	GLCM_MD_D	95.85	88.78	91.32	88.69	94.17	76.51	75.85	90.71	94.16	95.54
16	GLCM_MDMS_D	92.79	83.88	88.29	91.02	93.83	76.51	78.11	93.54	90.25	96.32
17	LBP_D [138]	92.82	87.24	87.87	90.23	93.96	72.68	71.18	91.02	94.79	97.40
18	CTGF_5x5_F	87.47	83.42	84.30	81.59	93.20	77.90	78.37	94.26	97.13	98.81
19	RECONST_ERROR_C [86]	87.43	90.75	90.34	92.74	92.47	43.72	48.11	95.18	98.01	97.96
20	CTGF_MDMS_D	91.25	84.30	83.59	88.49	88.15	84.53	83.37	91.53	91.67	96.44
21	CTGF_7x7_F	85.46	78.89	69.58	83.64	93.48	71.14	74.18	88.04	96.48	97.42
22	CTGF_5x5_D	87.13	75.73	80.58	82.92	90.51	75.70	77.71	90.94	91.95	94.36
23	CTGF_7x7_D	86.38	74.30	75.44	82.78	89.27	81.90	80.19	83.86	88.58	95.70
24	GLCM_D [69,54]	93.90	73.33	81.89	76.77	92.81	71.86	69.03	85.95	85.82	93.61
25	HOG_D [140]	85.41	71.43	81.59	81.14	92.31	54.01	53.90	89.78	91.79	93.37
26	HOG_F [140]	77.57	64.18	71.90	68.28	94.05	51.20	46.60	86.87	92.70	86.09
27	NOISE_STATS_C [73]	45.21	54.60	32.56	57.71	92.88	69.81	48.99	72.49	93.07	96.42
28	DWT_STATS_D [77]	15.01	12.27	21.78	21.40	92.60	39.74	10.23	28.08	37.64	53.93
29	DWT_STATS_F [77]	19.32	15.16	0.65	14.68	94.24	34.31	0.00	42.53	15.58	43.94
30	NOISE_STATS_F [73]	55.04	18.94	1.87	38.65	77.15	11.40	40.39	18.47	35.56	59.59
31	NOISE_STATS_D [73]	38.01	27.25	51.94	38.93	67.51	26.01	31.02	20.75	18.44	75.09
32	DWT_STATS_C [77]	29.19	2.00	0.00	0.30	93.82	3.96	0.00	8.16	56.24	25.60

Legend:
xx.xx = Two best *f-measure* results of the method
xx.xx = Two worst f-measure results of the method

Table 3.4: F-measure of each technique per printer. The proposed techniques in this chapter are the ones in bold in the column "Methods".

printer. Hence, texture micro patterns explored by these techniques are more important to identify this printer than its noise signature, explained by the very low f-measure of NOISE_STATS (45.21%). Another multidirectional approach proposed in this chapter, the GLCM_MD, shows the best f-measure to identify printer LE260 (100%).

For printer C1150, the multidirectional approach is not totally discriminative by itself. The highest f-measure for this task was achieved by the proposed multidirectional and multiscale GLCM on characters – GLCM_MDMS_C, with an f-measure of 99.04%. The same behavior is noticed on printers C3240, C4370, H1518, H225A, H225B. The multidirectional and multiscale texture micro patterns analyses and also the fusion proposed in this chapter showed the best f-measures (99.16%, 99.57%, 98.90%, 94.59% and 94.76%, respectively). Note also that the proposed methods performed well on printers H225A and H225B, which are from the same manufacturer and model, but with few firmware modifications. These are the most difficult printers to identify (see Table 3.4).

CTGF_GLCM_MDMS_F			Attributed Printer ID									
			B4070	C1150	C3240	C4370	H1518	H225A	H225B	LE260	OC330	SC315
			1	2	3	4	5	6	7	8	9	10
Actual Printer ID	B4070	1	99.66	0.34								
	C1150	2	0.18	99.82								
	C3240	3			100.00							
	C4370	4			0.84	99.16						
	H1518	5		1.34			97.82	0.84				
	H225A	6		0.34				94.92	4.75			
	H225B	7						5.32	94.68			
	LE260	8			1.53					98.47		
	OC330	9									100.00	
	SC315	10		0.17								99.83

Table 3.5: Confusion matrix for the best proposed technique: the fusion of the Convolution Texture Gradient Filter with 3×3 mask and Multidirectional and Multiscale GLCMs applied on Frames (CTGF_GLCM_MDMS_F). Results shown are in percentages.

GLCM_MDMS_F			Attributed Printer ID									
			B4070	C1150	C3240	C4370	H1518	H225A	H225B	LE260	OC330	SC315
			1	2	3	4	5	6	7	8	9	10
Actual Printer ID	B4070	1	100.00									
	C1150	2	0.35	99.47			0.18					
	C3240	3			98.99	1.01						
	C4370	4			0.50	99.50						
	H1518	5	0.34	1.18			97.65	0.84				
	H225A	6						92.54	7.46			
	H225B	7						3.30	96.70			
	LE260	8	0.85		0.17					98.98		
	OC330	9									100.00	
	SC315	10		0.17								99.83

Table 3.6: Confusion matrix for the second best proposed technique: the Gray-level Co-Occurrence Matrices Multidirectional and Multiscale on Frames (GLCM_MDMS_F). Results shown are in percentages.

Printers OC330 and SC315 are easy to identify. Both the multidirectional and multiscale GLCM and fusion proposed herein showed an 100% f-measure for printer OC330. The multidirectional HOG, LBP and the state-of-the-art GLCM also showed an 100% f-measure for printer OC330. Therefore, any of these techniques is enough to identify this printer. For printer SC315, both the multidirectional and multiscale GLCM and the CTGF_GLCM_MDMS_F fusion yielded 99.62% of f-measure. The multidirectional GLCM and the state-of the-art GLCM yielded the same f-measure. Tables 3.5 and 3.6 show the confusion matrices for the two best proposed techniques using percentages.

Finally, we also performed statistical tests to compare all the techniques. Performing a Friedmann test on f-measure values yielded a p-value of 2.2262×10^{-203}, indicating that there is statistical difference among the techniques. This allows us to perform the Tukey-HSD pairwise tests. Table 3.7 shows such results using the 15 best methods considering the f-measure. Each $(line, column)$ in this table shows whether a technique wins $(line)$, loses or draw in the statistical analysis against another technique $(column)$. The higher the method is in the rows, the better. A Bonferroni statistical test in this analysis also confirms the findings.

There are at least two interest aspects to observe in Table 3.7. The first one is regarding some of the proposed methods (GLCM_MDMS, CTGF_GLCM_MDMS and GLCM_MD). The techniques in this cluster are not statistically significant when compared pairwise with other members of the cluster. Interestingly, they explore multiscale and multidirectionality analyses we hypothesised as important for printer attribution confirming the hypothesis. Also, the two CTGF_GLCM_MDMS and GLCM_MDMS are the two

Rank	Method	CTGF_GLCM_MDMS_F	GLCM_MDMS_F	GLCM_MD_F	GLCM_MDMS_C	GLCM_MD_C	LBP_F [138]	HOG_C [140]	CTGF_3x3_F	CTGF_MDMS_F	GLCM_F [69,54]	GLCM_C [69,54]	CTGF_GLCM_MDMS_D	LBP_C [138]	CTGF_3x3_D	GLCM_MD_D	GLCM_MDMS_D	LBP_D [138]	CTGF_5x5_F	RECONST_ERROR_C [86]	CTGF_MDMS_D	CTGF_7x7_F	CTGF_5x5_D	CTGF_7x7_D	GLCM_D [69,54]	HOG_D [140]	HOG_F [140]	NOISE_STATS_C [73]	DWT_STATS_D [77]	DWT_STATS_F [73]	NOISE_STATS_F [73]	NOISE_STATS_D [73]	DWT_STATS_C [77]	BROS
1	CTGF_GLCM_MDMS_F	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	26
2	GLCM_MDMS_F	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	26
3	GLCM_MD_F	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	22
4	GLCM_MDMS_C	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	21
5	GLCM_MD_C	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	21
6	LBP_F [138]	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	21
7	HOG_C [140]	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
8	CTGF_3x3_F	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
9	CTGF_MDMS_F	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
10	GLCM_F [69,54]	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
11	GLCM_C [69,54]	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	16
12	CTGF_GLCM_MDMS_D	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	7
13	LBP_C [138]	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	7
14	CTGF_3x3_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
15	GLCM_MD_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	-1
16	GLCM_MDMS_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	-3
17	LBP_D [138]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	-3
18	CTGF_5x5_F	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	-3
19	RECONST_ERROR_C [86]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	-3
20	CTGF_MDMS_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	-6
21	CTGF_7x7_F	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	-8
22	CTGF_5x5_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	-9
23	CTGF_7x7_D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	-10
24	GLCM_D [69,54]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	-10
25	HOG_D [140]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	1	1	-14
26	HOG_F [140]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	-17
27	NOISE_STATS_C [73]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	-20
28	DWT_STATS_D [77]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-25
29	DWT_STATS_F [77]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-25
30	NOISE_STATS_F [73]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-26
31	NOISE_STATS_D [73]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-26
32	DWT_STATS_C [77]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-26

1 = Line method is better than column method
0 = Line method is equivalent to column method
-1 = Line method is worse than column method

Table 3.7: Tukey-HSD pairwise statistical test results using f-measures of the 15 best methods present in Table 3.3. The value 0 means that there is no statistical difference between the methods. The value 1 means that the method in the corresponding row is statistically better than the method in the corresponding column while -1 means otherwise.

best performing techniques. The second one regard the cluster comprising the state-of-the-art techniques for printer attribution, the two general-purpose texture descriptors (LBP and HOG) and one variation of the proposed methods (CTGF 3×3 on frames and documents). In this cluster, most of techniques are statistically significant from most of others in the same group.

From Table 3.7, we can see that the Multiscale and Multidirectional Analyses proposed for frames of documents with sufficient printed material is more effective than the state-of-the-art techniques for the dataset we consider herein (with different letter sizes, styles and figures). This is noticeable by the two best techniques ranked in that table, CTGF_GLCM_MDMS_F and GLCM_MDMS_F.

3.5 Final Considerations and Further Developments

In this chapter, we propose new approaches for laser printer attribution, a problem of paramount importance because they can be a powerful tool to help solving crimes involving documents. Our solutions work beyond simple texture approaches as they analyze how the texture on printed text and figures behave when using multidirectional and multiscale analysis.

Our first contribution to achieve this task are descriptors based on statistics of the multidirectional gray-level co-occurence matrices (GLCM_MD) and multidirectional and multiscale gray-level co-occurrence matrices (GLCM_MDMS). The first one analyzes multiple neighboring directions and the second analyzes multiple neighboring directions in

a pyramidal Gaussian image decomposition. This can be helpful when texture spreads over multiple directions and scales. Our second contribution is the Convolution Gradient Texture Filter, which considers low-gradient micro-texture patterns. This descriptor is also multidirectional as it calculates textures over a neighborhood with different kernel sizes. It analyzes the frequency of how a pixel relates to its neighbors on areas of low-level gradients (*i.e.*, inside printing area) in texts and figures printed by different printers. We also proposed a fusion between the analyses made by both of them.

Our last contribution refers to the best place to look at on printed documents to better investigate printing patterns. By analyzing areas of text with enough printing material, we can identify the laser source printer in a better way than just looking at characters and documents as more printing textures and less background are available. This technique has the same advantage of the characters analysis, that is representing a document with multiple feature vectors, classifying them individually and fusing the individual classifications in the end. An additional advantage when compared to a full-document analysis is that this method can be applied if just parts of the document are available.

We compared the proposed approaches against some state-of-the-art and some general purpose texture descriptors in Wikipedia scanned documents and showed their effectiveness when the characterization occurs in characters, frames and documents. The techniques proposed herein yielded the first and second best classification accuracies when applied on the proposed frames. They were the best to identify 90% of the printers and results are statistically different when compared with the state-of-the-art counterparts. The take-home lesson is that the multidirectional analysis is crucial for laser printer attribution, specially when combined with multiscale image decomposition.

From our experience, it is important to highlight that laser printer attribution is a very difficult problem in which many variables play a role. First of all, the reference scanner used in the scanning process when defining the training samples and analyzing an investigated document must be the same as we do not want intrinsic scanning features to play a key role in the printer attribution problem. When using the same scanner for training and investigated documents, we rule out this effect. The scanning process inserts intrinsic features in the documents, which can be used to identify the scanning device. This is known as Scanner Attribution in the literature and there are very good work on this as references [140, 141] show.

This is not a major problem for the forensic expert because our application here is to identify the printer source of a document. So, the scanner variable can be fixed. There are some situations where different scanners have very similar variables (resolution and noise), but we cannot guarantee that in all practical scenarios. Therefore, we recommend that the scanner used for acquiring the investigated documents should be the same as the one used for training the classifier. As just a few documents are necessary for training the classifiers, this is straightforward. This procedure is also used in other devices attribution (cameras and scanners). When a suspect camera is investigated, the classifier must be retrained with data acquired with that camera [142, 143, 144].

Second, it is advisable to use, as much as possible, similar paper to the one collected for investigation. If the investigated document for printer attribution is a white office

Letter with $75g/m^2$, it is recommended to use a similar paper in the training (acquisition of training documents from the suspect printers). If we use training data considering photographic reflective paper, for instance, which are very different from the investigated printed document, it is likely the proposed methods and their counterparts in the literature using vision-based approaches will fail.

In addition, the good results presented in this chapter must come with a salt of grain as well. We are not claiming to have solved the printer attribution problem. The almost 99% classification accuracy is an important and unrivaled result. However, each real case will have its specificities. For example, the result shown in this paper was in a given dataset and in closed set scenarios. The behavior of the proposed approach in another dataset of other printing sources and in open set scenarios is unknown. So, the forensic expert must be aware that vision-based approaches are an initial, non-destructive and cheap analysis. It must be used, whenever possible, with other techniques in order to provide the most accurate results as possible. The vision-based techniques can also be combined to improve the quality of the attribution.

Finally, we envision at least three research paths for extending this research. First, an in-depth study of the analyzed techniques on color documents with proper adaption of the methods for this scenario is worth exploring. Second, a deeper investigation on the complementarity of the proposed methods is paramount for dealing with open-set setups and more classes of interest would be interesting to check if classifier and decision-level fusion could push the classification results even further. Finally, domain adaptation procedures can be studied to adapt features in our training dataset to new testing datasets.

Chapter 4

Data-Driven Solutions for Laser Printer Attribution

In this chapter, we discuss *data-driven* approaches for laser printer attribution that learn features to be used for classification of sources directly from the data, without feature engineering such as used in Chapter 3. For that, we propose the first deep learning solution for laser printer attribution, which uses our *multi-analysis* scenarios. These approaches use multiple representations of multiple data and act in the pre-processing analysis of an input questioned image. Thus, the printing patterns are represented differently and the banding can be better described in letter areas through the networks using the information of our several deep learning networks together.

4.1 Motivation

Printed documents are found everywhere: in offices, public agencies, schools and also residences. From single documents available today as homeworks and warnings to serious ones as contractual clauses and scientific articles there is a printer involved, being it a dot matrix, dye-sublimation, thermal, ink-jet or laser. The last one has been the choice of ordinary people and offices in this last decade [145] because of its speed, quality of printing and low price.

However, with this massive access to printing devices a new treat has emerged: the use of laser printer for criminal intentions. Additional contractual clauses inexistent before, child pornography and animal abuse photos, life treat letters, illegal correspondence, terrorist plots, fake currency and fake documents are printed everyday. Hence, providing ways of pinpointing printing ownership of documents is paramount, mainly to link them to criminals. Also, linking a document to a printer is another way of authenticating official documents.

In this chapter, we aim at designing and developing a multi-analysis data-driven approach to automatically extract meaningful discriminative patterns straight from the analyzed documents, instead of using ordinary feature engineering. For that, we propose the first deep learning solutions for laser printer attribution that use several deep Convolutional Neural Networks (CNN). Our approaches are based on late and early fusion of these deep

networks, both of them are based on different representations (image transformations) of image patches containing different letters of text. The differential of the method, if compared with state-of-the-art counterparts is the CNN recognition of printing patterns of each printer by only running the printer letter samples through the network. In other words, the patterns of interest are learned directly from the data of interest from the forensic investigator. In summary, the main contributions of this chapter are:

1. Design of a unified deep-learning-based architecture comprising different, and lightweight deep networks operating in parallel for laser printer attribution, aimed at detecting features that identify the intrinsic artifacts over different situations (raw image pixels, median filtering residuals, among others) of different data (different extracted patches of printed material).

2. Consideration of multiple representations of the data in the analysis, which are image operations that can better highlight the printing patterns of different printers (e.g., raw image pixels, median filtering residuals, among others).

3. Analysis of the natural complementarity present in different printing patterns of different printed regions (patches), such as letters and shapes, instead of the common 'e' letter used before by virtually all works in the literature.

We organize the remaining of this chapter in five sections. Section 4.2 shows a short tutorial on Convolutional Neural Networks and the necessary concepts for understanding this chapter; Section 4.3 presents our approach, which is, as far as we know, the first approach based on deep learning applied on multiple representations of multiple data for laser printer attribution; Section 4.4 shows the setup we use in this chapter for validating the proposed methods and compare them to the existing counterparts in the literature, while Section 4.5 shows the performed experiments and results. Finally, Section 4.6 concludes this chapter with our final considerations and proposals for future work.

4.2 Convolutional Neural Networks

Before we discuss our proposed method to perform laser printer attribution based on Deep Learning, it is worth discussing some basic Deep Learning (DL) concepts. DL networks are, basically, a Neural-based network with many layers. The benefits of using a Neural Network for classification tasks are basically two: (i) in the earlier layers, complex patterns are broken into simpler patterns, helpful in the image classification and (ii) in the training phase, the classification error is back propagated to the previous layers, which helps the network to adjust its weights and then, within a number of training epochs, the network can finally find the features that will best represent the input data. The most important advantage of this representation form is that it is learned directly from the data and is not custom-engineered.

In this chapter, we are particularly interested in deep Convolutional Neural Networks (CNNs) for solving the printer attribution problem. Pioneered by Lecunn et al. [146], they are a family of Deep Learning methods aimed at image recognition and object detection

tasks. CNNs are dominating the space of Computer Vision in the last years and, in early 2015, a Microsoft-designed CNN outscored humans in the task of image recognition [147]. In its core, a CNN is similar to a Neural Network, with some particularities, using the following layers:

1. Input Layers: where the data enters the network. Raw pixels of images or transformations in the data that can better emphasize some specific aspects searched by the network can be used.

2. Convolutional Layers: contains a series of filters with fixed size used to perform convolution on the image data, generating what is called *feature maps*, treated by the subsequent layers. These filters can highlight some patterns helpful to do the image understanding, such as edges in human faces, for example.

3. Rectified Linear Unit (RELU): RELU layers normally follow convolution operation and are responsible for applying a non-linear function to the output x of the previous Layer, such as $f(x) = max(0, x)$. According to Krizhevsky et al. [148], they can be used for fast convergence in the training of CNNs, speeding-up the training as they deal with vanishing gradient problem by keeping the gradient more or less constant in all network layers.

4. Pooling Layers: These layers ensure that the network focuses only on the most important patterns yielded by convolution and the RELU. A Pooling Layer summarizes the data by sliding a window through the feature map and applying some non-linear operations on the data within the window, such as mean, reducing the dimensionality of the feature maps used by the following layers.

5. Fully-connected Layer: Used for the understanding of patterns yielded by the previous layers. It is located at the end of the network and is commonly a classifier such as soft-max.

Figure 4.1 depicts one possible CNNs architecture. However, the layer arrangement (number) of each type of layer and localization can change depending on the network used for a given application.

Although very powerful at representing patterns present in the data, the main drawback of such deep networks is the fact that common CNNs normally need thousands or even millions of labeled data for the training, which can be challenging and time-consuming for real-world applications. In this chapter, we present an alternative approach that deals with this requirement by considering several lightweight Deep CNNs running in parallel for the problem of laser printer attribution, as we shall discuss in section 4.3.

4.3 Proposed Method

In our work, we aim at solving the requirement of several input training images for deep networks, applying this solution for laser printer attribution. Our proposed methods use several deep networks with very few layers, combining the output at the end. Our

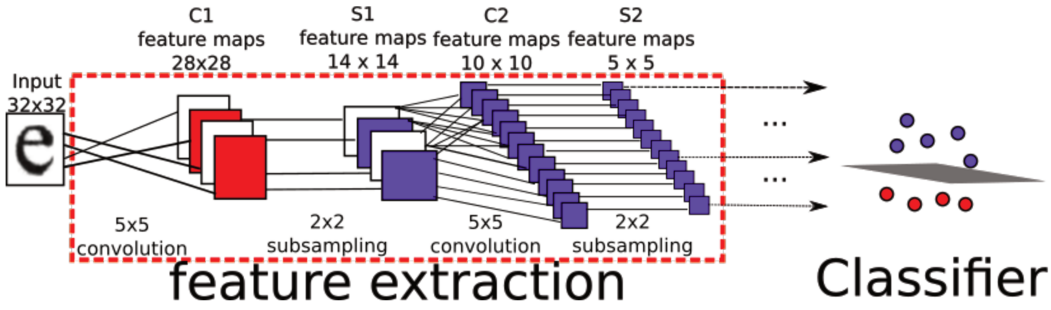


Figure 4.1: Common layer arrangement of a Convolutional Neural Network. The input image is transformed into feature maps after a series of transformations such as convolutions, pooling and RELU, until a feature vector is created and can be used as input to the fully-connected layer, used for classification.

intuition is that (i) several simpler deep networks can be effectively trained using less training examples (ii) the early layers of the simpler networks identify better areas where banding artifacts are present by decomposing them in simpler patterns and (iii) the back propagation in the training phase allow the fine tuning of the filter weights to better identify the features present in printing patterns of different printers.

To create our new representational solution for printer attribution, we resort to a technique considering multiple representations of the input data along with multiple simpler deep networks. Different representations other than the image space directly have already been successfully discussed in the forensics literature considering deep networks [149]. In our work, we consider the following input representations to the different deep networks designed:

1. Raw data: the image pixels are used as input to the network as they are. This is the common representation used as input for CNNs, as it contains high and low frequency components that can be isolated by the CNN filters and can be useful for image classification.

2. Median Filter Residual: we apply the median filter over the image and subtract the image from the filtered version. The yielded noise pattern will be used as input to the network. As the median filter better preserves edges, the Median Filter Residual will contain only some high frequency imperfections, which can be regarded as the banding.

3. Average Filter Residual: we use the average filter over the image and subtract the image from its filtered version, using this residual as input to the network. This residual isolates border effects by subtracting the smoothed image version (the mean filter reduces the amount of intensity variation between one pixel and the next) and the original, using these borders as the input for the network.

Our deep networks are focused on identifying text documents. In other words, we apply the classification on extracted letters from the documents and use the majority voting over the classifications to decide the class of the document. Figures 4.2, 4.3 and 4.4 depict the common letters “e” extracted from 10 different printers using the three representations

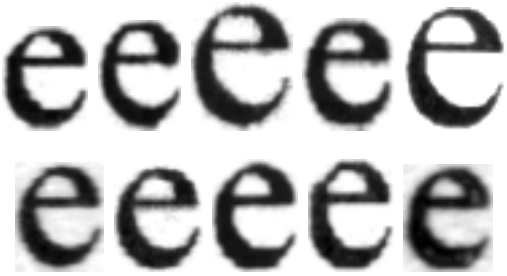


Figure 4.2: Same letter "e" printed by different printers. In some cases, some printers print physically bigger letters than others for the same document.

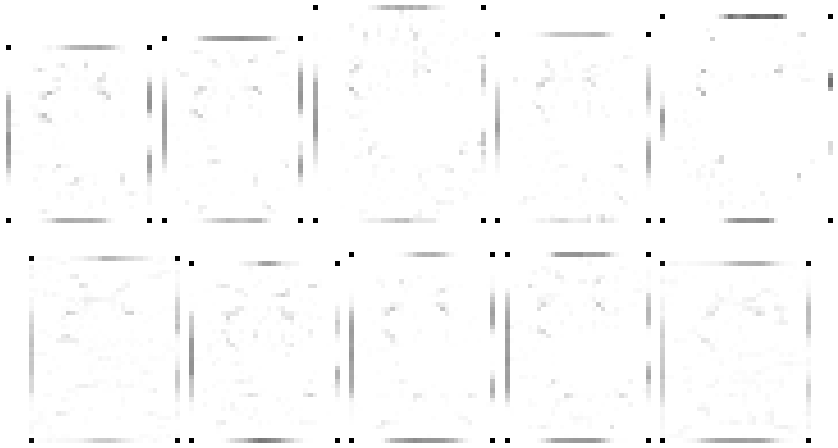


Figure 4.3: Median Filter residual representation of the same letters "e" showed in Figure 4.2. Here, some minimal borders are highlighted. The pixel values are inverted in this Figure for better visualization.

used in our deep networks. One important thing to note from these figures is the fact that some printers used in this chapter (printers C3240 from Canon and H1518 from HP) print the same letter from the same document with different sizes in the physical paper, with consequently different banding patterns.

The used network architecture is the same in spirit to the MNIST network for digit recognition [150]

To identify the source using such tiny images from the documents, we use a simple deep network. The used network is basically the same as the MNIST network for digit recognition [150], with proper custom-tailoring. For a better representation of the data of interest herein, we trained the network from scratch, yielding new filter weights able to recognize letter areas containing banding characteristics of laser printers. As far as we know, this is the first deep network custom-taylored for the printer attribution problem. It differs from traditional deep learning-based solutions in the vision community by focusing the feature learning on the printer source of an input rather than on what the input actually is (e.g., its class). Our core CNN architecture, which is also the basis of the several networks proposed in this chapter has the following layers.

1. One Input Layer, where the raw image or a different representation (median filter residual or average filter residual) is used. It requires 28×28 images as input.

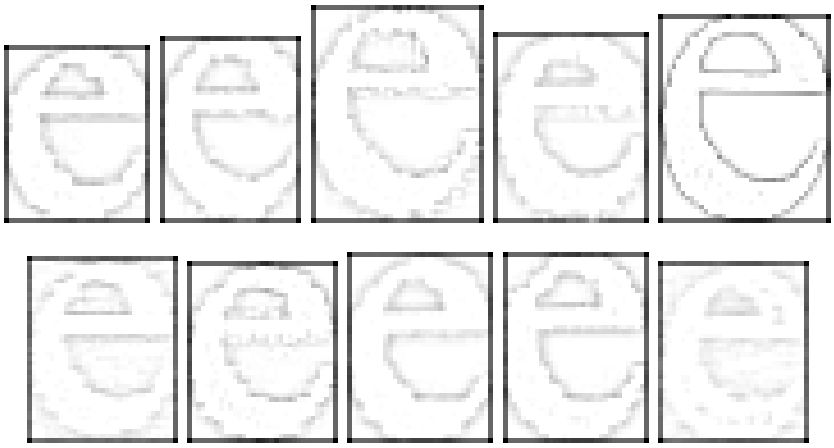


Figure 4.4: Average Filter Residual representation of the same letters "e" showed in Figure 4.2. Here, natural borders are highlighted. The pixel values are inverted in this Figure for better visualization.

2. The first Convolutional Layer comprises 20 5×5 filters and is followed by a pooling layer.

3. After the first pooling layer there is a second Convolutional Layer, with 50 filters of dimensions $5 \times 5 \times 20$, followed by another pooling layer.

4. After the second pooling layer there is a third Convolutional Layer, with 500 filters of dimensions $4 \times 4 \times 50$

5. One RELU layer.

6. After the RELU layer there is another Convolutional layer, with 10 filters of dimensions $1 \times 1 \times 500$, yielding $500d$ feature vectors.

7. A fully-connected layer, which is a soft-max classifier.

In our proposed approach, we train the network using a given number of epochs using this architecture and then feed the training images again to network, extracting the feature vectors in the last but one layer, using 500-dimensional feature vectors to train a linear SVM with an One-vs-One class binarization policy [151]. For testing, we apply the test images in the already trained network and extract the feature vectors in the last but one layer, using them as input to the already trained SVM classifier, which will predict their classes.

The proposed networks extract the features from letter areas containing banding without feature engineering, as the backpropagation propagates the error through the training epochs and updates the filters to minimize the classification error using only the information present in the images to identify the printing patterns. As discussed before, the earlier layers can detect minimal and relevant characteristics from the letter areas containing banding that can be used together in the posterior layers for better classification. Figures 4.5 and 4.6 show the 20 filters of the first layer in a grid and also the characteristics they highlight from a letter printed by a given printer. These figures show that different

filters enhance different areas of letters, such as texture and borders detectable only by hand-crafted features in the literature such as the approach proposed by Ferreira et al. [152] (approach discussed previously on Chapter 3) originated by microscopical analysis on letters.

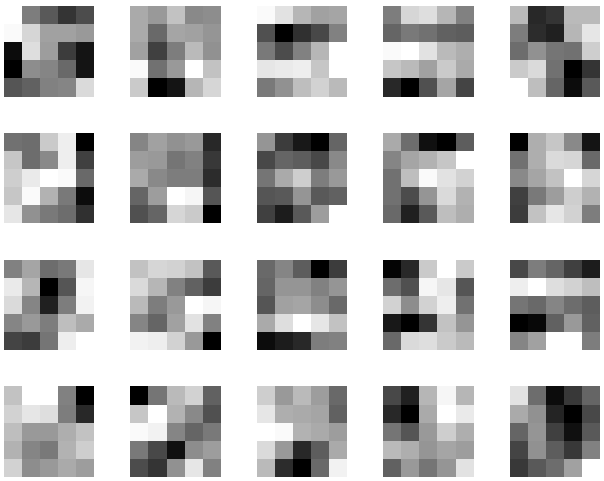


Figure 4.5: Final filters weights of the first convolutional layer operating on the raw input image pixels. The weight values are represented by grayscale values.

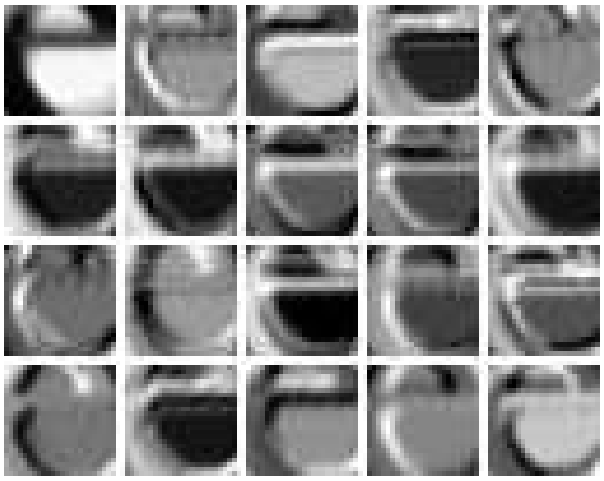


Figure 4.6: Convolutional output of the first layer of the trained network, given an input letter from an investigated printer. For each filter, different areas inside or outside the borders are highlighted.

Although this network is not as deep and complex as some recent ones in the Vision community [153], it still needs several samples in the training for better classification. We aim at minimizing this problem by using several deep networks in parallel and test the complementarity of them by aggregating the output. For that we use several CNNs, each one using a different representation (image operation) of different data (different extracted characters). We hypothesize that it is important to analyze the banding features in different kinds of extracted letters, instead of only the letter 'e'. In this vein, our proposed approach uses CNNs in the following configurations, which are depicted in Figures 4.7 e 4.8:

1. **Multiple representations of the same data (early fusion):** we apply three different networks, each one applied to a different representation (raw, median filter residual and average filter residual) of the same letter and combine the generated feature vectors (weights of the last but one layer as we do not want soft-max outputs in this case) in an early fusion [154] fashion to feed any classifier later on in the decision-making step.

2. **Multiple representations of different data (late fusion):** we apply the approach used before individually to each data (extracted character), each one classifying the same character. The final classification for a document uses voting, in a late fusion [154] fashion, on the different multi-representation networks.

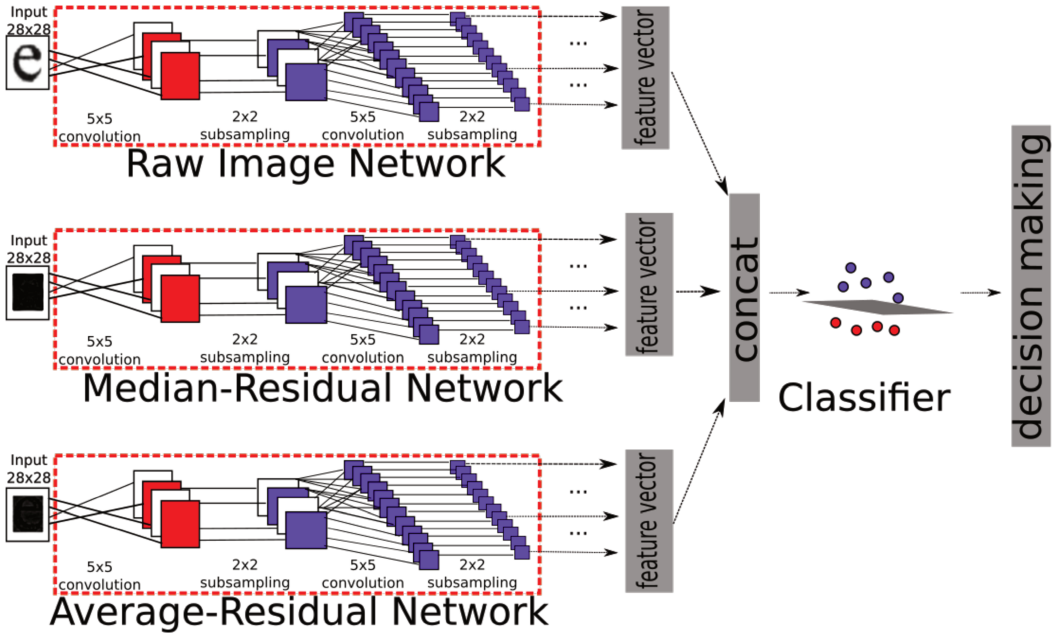


Figure 4.7: Proposed multiple representations of the same data for laser printer attribution through several Convolutional Neural Networks running in parallel. After applying all networks individually on the data, all the final feature vectors of each network are merged by concatenation to yield a final feature vector.

The multi-analysis procedure performed here considers several representations (image transformations) of multiple data (extracted patches containing letters). The reason for that is the fact that different data (letters) contain different printing patterns and by using different transformations, it is possible to better highlight them if using all the analyses together in a CNN. Here the multi-analysis is used in the pre-processing stage, giving more samples with different image transformations to the proposed deep networks, a step that also can be seen as a data augmentation strategy, considering the standard name in the deep learning literature.

We show in the experiments that the CNNs proposed use the complementarity of the information of letter areas containing banding patterns detected by each network on each representation of different data, as the banding can vary depending on the letter printed. This shall be discussed further in section 4.5.

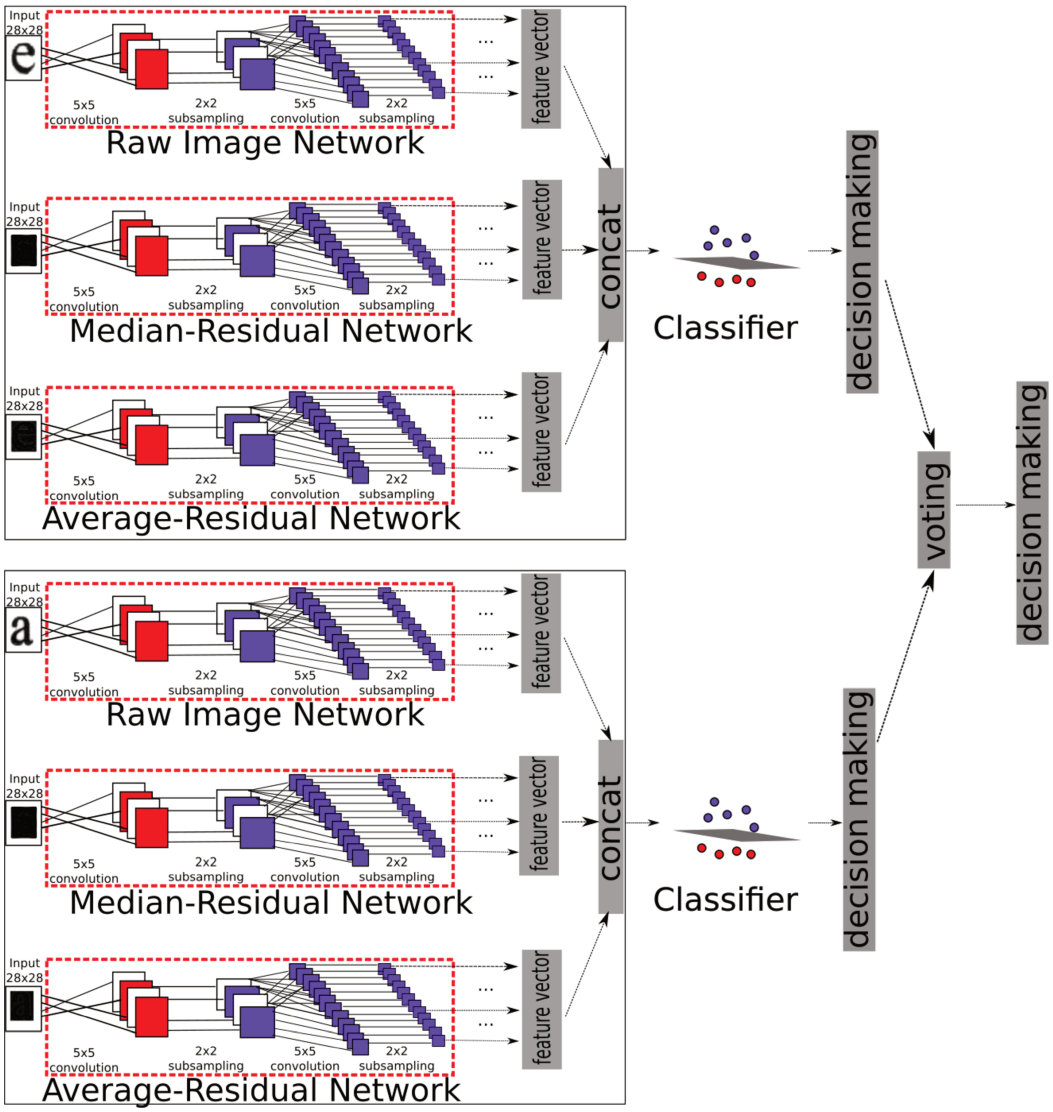


Figure 4.8: Proposed multiple representations of the same data for laser printer attribution through several Convolutional Neural Networks running in parallel.

4.4 Experimental Setup

This section presents the experimental methodology used in this chapter along with the used evaluation metrics, dataset, statistical tests and, finally, details of the existing methods used as baselines in the experiments.

4.4.1 Dataset

For validation, we consider the same dataset discussed previously on chapter 3. We chose to use only patches of letters sub-dataset for two reasons: (i) they have dimensions close enough to be suitable for the networks used and (ii) they can be applied on cases where it is not possible to extract the frame data presented on chapter 3, for example, on documents containing few lines of text. However, to satisfy the CNN requirements of input images with same size, we cropped pixels from the extracted letters, alternating the pixel removal from the right/bottom regions of the image until the final image dimensions is 28×28. In

some cases, in which the picture containing the extracted letter is smaller than 28×28, we pad the borders with zeros. We did not try to reduce the image with interpolation because we wanted to avoid additional artifacts in the printing patterns that can confuse the classification, as the resize will be applied on images bigger or smaller than 28×28 differently. This way, we avoid interpolation artifacts in the dataset and will only use printing artifacts in the analysis. Figure 4.9 shows the variation of the extracted letters dimension of the ten printers considered.

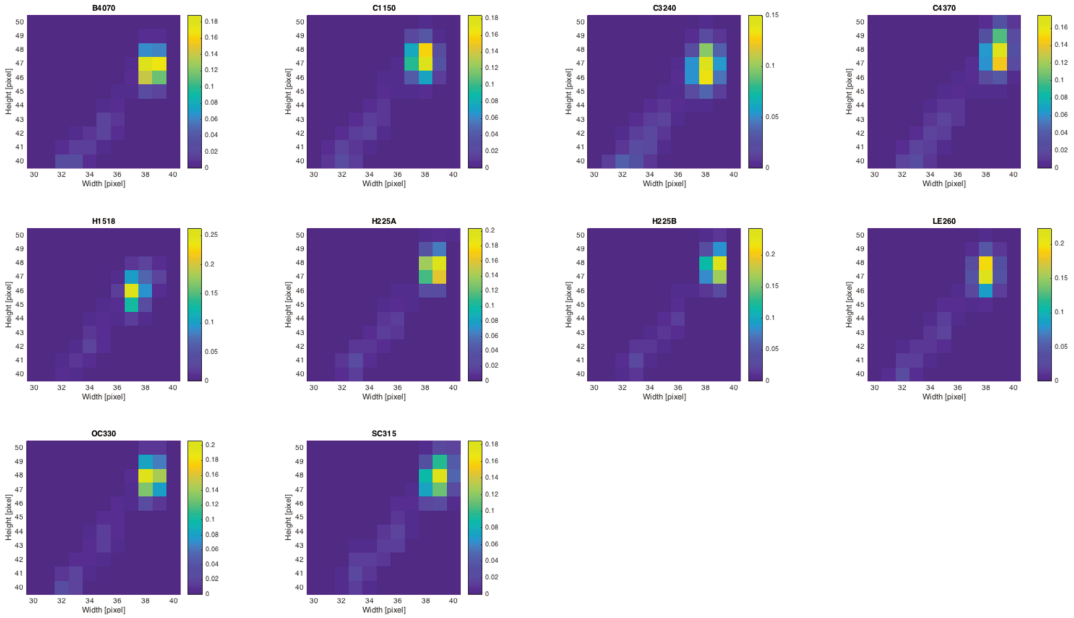


Figure 4.9: Dimension variation of the dataset of extracted patches containing letters "e" per printer. Colors represent percentages on which scale is shown in the right part of the graphic.

We extracted several different letters of these documents according to their frequency in the English language [83], resulting in the following different datasets of different letters:

1. Letters "e" dataset, containing 245,650 extracted letters.
2. Letters "a" dataset, containing 286,098 extracted letters.
3. Letters "d" dataset, containing 185,009 extracted letters.
4. Letters "o" dataset, containing 351,850 extracted letters.

We also apply a 28×28 frame extractor in the documents, extracting 300 valid frames from images whose ratio between black and white pixels r is $0.6 \leq r \leq 0.8$. This results a total of 352,433 printing patches of printing patterns also used in the experiments.

4.4.2 Experimental Methodology, Evaluation Metrics and Statistical Tests

For validation, we consider a 5×2 cross validation protocol. In this protocol, we replicate the traditional 2-fold cross-validation protocol five times (thus 5×2). In each of these 2-fold cross validations, a set of data $\boldsymbol{D}$ is randomly divided into $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$. Then, a classifier is trained with $\boldsymbol{D}_1$ and tested on $\boldsymbol{D}_2$, and then we train the classifier with $\boldsymbol{D}_2$ and test it on $\boldsymbol{D}_1$. After that, we report the statistics and perform the statistical tests after 10 rounds of experiments. According to a study conducted by Dietterich et al. [138], this is considered an optimal experimental protocol for learning algorithms.

In a multi-class problem with n classes, the classification results may be represented in an $n \times n$ confusion matrix. In this case, the main diagonal contains the true positives while the other entries contain either false positives or false negatives. In the 5×2 cross validation protocol, one confusion matrix is yielded per experiment. All the metrics showed in the remaining of this section will be applied over the average of such matrices.

Accuracy

For a multi-class problem, the accuracy is the sum of the main diagonal values divided by the total elements in the matrix. As the correct classifications are in the main diagonal, we can define the classification accuracy as:

$$accuracy = \frac{\sum_{i=1}^{n} M(i,i)}{\sum_{i=1}^{n}\sum_{j=1}^{n} M(i,j)}. \tag{4.1}$$

The accuracy considered in the experiments always takes into account the correct classifications of documents. The classification of documents happens by majority voting on the classification of extracted letters of that document. The classification of each extracted letter casts one vote to be used for the majority voting at the end.

Precision

The precision of a classifier for a printer i measures how many documents attributed to such a printer are actually correct.

$$\boldsymbol{Precision}(i) = \frac{M(i,i)}{\sum_{j=1}^{n} M(j,i)}. \tag{4.2}$$

Recall

The recall of a classifier for a printer i, also called the true positive rate, measures how many documents of the total number of documents from printer i are correctly attributed to such printer

$$\boldsymbol{Recall}(i) = \frac{M(i,i)}{\sum_{j=1}^{C} M(i,j)}. \tag{4.3}$$

F-Measure

The *f-measure* of a classifier to classify a printer i is the harmonic mean of precision and recall for the printer i, as follows:

$$\boldsymbol{f}(i) = 2 \cdot \frac{\boldsymbol{Precision}(i) \cdot \boldsymbol{Recall}(\mathrm{i})}{\boldsymbol{Precision}(i) + \boldsymbol{Recall}(i)}. \tag{4.4}$$

To test the statistical relevance of the obtained experimental results, we consider a two-level statistical test. In the first level, we use the Friedman test as a pre-test to point out whether or not there is statistical difference in the obtained results. Then we refine these results with the Tukey-Kramer post-test, also known as Honestly Significant Difference (HSD) test to point out statistical differences (if any) pairwise. In all tests, we set the confidence level to 95%.

4.4.3 Baselines

We compare the proposed approach with several baselines, some of them we propose in this chapter and others were proposed before in the literature. Firstly we will compare our approaches against several other deep networks, applied over single representations of the same data. The single representations used consists on the application of one network over the median filtering residual of the image (DL_NOISE_IMAGE_MEDIAN_C_CROP), average filter residual (DL_NOISE_IMAGE_AVERAGE_C_CROP) and in the raw image pixels (DL_NATURAL_IMAGE_C_CROP). These approaches are applied over the "e", "a", "d" and "o" letters dataset individually. We also tested the filtered image from the Convolutional Texture Gradient Filter from the work of Ferreira et al. [152] (approach discussed previously on chapter 3) and also the Wiener Filter Residual [155] as the input for the network (DL_CTGF_3X3_C_CROP, DL_CTGF_5X5_C_CROP and DL_NOISE_IMAGE_WIENER_C_CROP, respectively). The latter three approaches were tested only on letters "e".

We also compare our proposed techniques against eight state-of-the-art methods (c.f. section 2.1.2) focused on text documents. The first one is the approach based on Gray-Level-Co-Occurrence Matrices from Mikkilineni et al. [68, 53], which describes the signature present in the banding with 22 statistics calculated per matrix. We call this approach in the experiments as GLCM_C_CROP.

In addition, the next four methods used in the experiments were proposed in the work of Ferreira et al. [152] (approach discussed previously on chapter 3). The first one uses Gray-Level-Co-Occurrence Matrices with more directions (GLCM_MD_C_CROP), while the second uses Gray-Level-Co-Occurrence Matrices with more directions and more scales in the input data (GLCM_MD_MS_C_CROP). The third uses Convolutional Texture Gradient Filters with size dim (we use $dim = 3$ and denote this approach as CTGF_3X3) and finally the fourth method takes the combination of two last methods (CTGF_GLCM_MD_MS_C_CROP).

The sixth state-of-the-art method implemented was proposed by Kee and Farid [85] (RECONST_ERROR_C_CROP). This approach uses reference characters firstly to look for the same characters of different printers. Then, the mean character and Principal

Component Analysis (PCA) [86] are performed on the characters per printer to calculate the printer profile. To detect the source of a printer, the document under investigation has its letters 'e' extracted and are used with each printer profile to calculate a *reconstruction error* of each printer, with the smallest mean error identifying the source. Finally, we also experimented with two well-known texture descriptors widely used in the literature. The first one is the Local Binary Patterns (LBP_C_CROP) [137] and the Histogram of Oriented Gradients [139] (HOG_C_CROP).

4.4.4 Implementation Aspects of the Proposed Methods

In this chapter, we propose a deep network composed of several lightweight deep learning networks running in parallel focused on identifying the banding of letter areas using multiple representations of multiple input data. This approach, which we call the late fusion, is based on voting after the classification of combined feature vectors from multiple representations of different data. We call these approaches DL_LATE_FUSION_C_CROP, specifying on which data they are applied. This yields 4 approaches tested in the experiments.

We also test the performance of this last approach when compared to what is called the early fusion. For this, we concatenate the feature vectors from the last but one layer of deep networks applied on three different representations of the same data, making them the input to a machine learning classifier. We did this for letters "e", "a", "d", "o" and 28×28 frames. We call these approaches DL_EARLY_FUSION_C_CROP, specifying in the experiments results which data they are applied. This yields 5 approaches tested in the experiments. The source code of the proposed approaches in this chapter can be found at GitHub[1].

We found the number of epochs to train the network after doing an experiment using validation and training data. We run the network using 25 epochs and decided to use 20 epochs to train the network, as the validation top1 error is small for this number of epochs. Figure 4.10 depicts this experiment results. We choose 0.001 as the learning rate used in the backpropagation of error in the proposed networks.

4.5 Results and Discussion

We now turn our attention to the experiments with different methods. We firstly show the experiments results considering the comparison between the unique representation and multiple representation (early fusion) of the same data using the proposed CNNs. Secondly, we show the comparison between the representations of unique data and multiple data (late fusion). Finally, we discuss an experiment comparing the performance of our approach against the state of the art discussed in section 4.4.3. All experiments were performed using the methodology presented in section 4.4.2 on the dataset of 1,184 printings shown before in section 4.4.1.

[1]https://github.com/anselmoferreira/deep-learning-printer-attribution

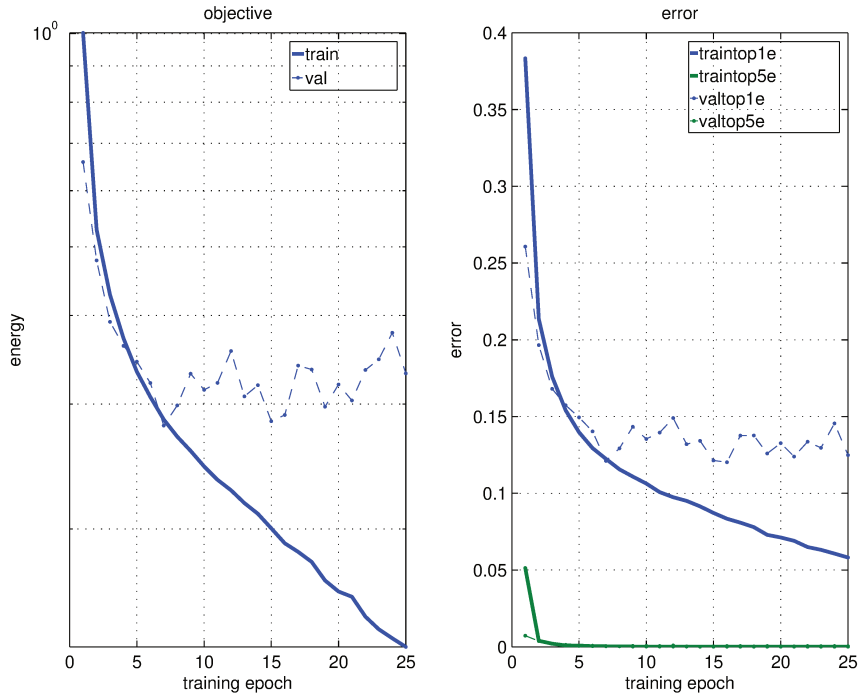


Figure 4.10: Experiment results showing the objective and energy values (left) and error(right) on different number of epochs for training and validating the convolutional neural network proposed in this chapter.

4.5.1 Comparison of Single Representations against Multiple Representations

We start with experiments considering the presented CNNs on multiple and single representations of the single data (extracted letters). In Table 4.1, we show the results of the 5×2 cross validation experiments considering this scenario.

Method	FINAL ACCURACY (ACC) AFTER 5X2 CROSS VALIDATION ON 28x28 IMAGES		
	ACC (%)	σ	Data Considered
DL_EARLY_FUSION_C_CROP_A	**96.89%**	**0.00**	**Letters 'a'**
DL_EARLY_FUSION_C_CROP_E	**96.84%**	**0.00**	**Letters 'a'**
DL_NATURAL_IMAGE_C_CROP_E	96.13%	0.00	Letters 'e'
DL_NOISE_AVERAGE_C_CROP_A	94.89%	0.30	Letters 'a'
DL_NOISE_AVERAGE_C_CROP_E	94.50%	0.03	Letters 'e'
DL_NOISE_IMAGE_MEDIAN_C_CROP_E	94.30%	0.01	Letters 'e'
DL_EARLY_FUSION_C_CROP_D	**93.67%**	**0.03**	**Letters 'd'**
DL_NOISE_IMAGE_MEDIAN_C_CROP_A	93.34%	0.02	Letters 'a'
DL_NATURAL_IMAGE_C_CROP_A	93.07%	0.03	Letters 'a'
DL_EARLY_FUSION_C_CROP_O	**92.21%**	**0.03**	**Letters 'o'**
DL_CTGF_3X3_C_CROP_E	89.12%	0.03	Letters 'e'
DL_NOISE_IMAGE_WIENER_C_CROP_E	84.84%	0.30	Letters 'e'
DL_CTGF_5X5_C_CROP_E	83.15%	0.06	Letters 'e'
DL_EARLY_FUSION_C_CROP_FRAME	**73.69%**	**0.05**	**Frames**

Table 4.1: Results considering unique and multiple representation of the same data. Multiple representation (early fusion) approaches are highlighted in bold.

As Table 4.1 shows, the approaches using early fusion of different representations outperform the ones using only the single representations. This is explained because different representations in the input layers of CNNs can contain important information

that better identify the banding on letter areas over the different networks, as well as other printing artifacts left behind during the physical printing of a document. For example, banding in the borders contained in the average filter residual are better highlighted in its CNN and can complement the information found in the two other CNNs that use information from the raw image data and median filter residual. We also found that the best data to use the multiple representation are over letters "a", firstly because they contain more borders than common letters "e", and also beause they are the most common letter in Portuguese texts and one of the most common letters of English texts [156, 83], so more data are investigated to decide a class of a document.

We also discovered that, for some data (letters), the raw representation in deep networks is not good enough, also justifying the use of multiple representations. For instance, letters "a" deep networks applied on average filter residual (DL_NOISE _AVERAGE_C_CROP_A) yielded an accuracy of 94.89%, against their accuracy of 93.07% on letters "a" raw image pixels (DL_NATURAL_IMAGE_C_CROP_A). We also found that the investigation of representation of the input as 28×28 frames (DL_EARLY_FUSION_C_CROP_ FRAMES), instead of standardized letters are not as effective when deploying a solution using Deep Learning. This happens because the network is fed with different images containing different printing patterns, instead of only a specific kind of patch containing the letter "e" with an almost fixed printing pattern. This way, the network must find traces in different images containing different patterns, and it is difficult for it to do it with so few different images available.

The good result of 96.13% of the deep applied on letters "e" (DL_NATURAL_IMAGE_C_CROP_E) shows that the CNNs proposed are able to cope with letters that do not perfectly fit the 28×28 window required as input for the network, as this dataset has different dimensions in the extracted patches (see Figure 4.9 for details). Hence the network should be easily generalizable to different font sizes.

The results using our CNNs on other single characters such as "a", "o", etc. shows that the data-driven proposed approaches work for different input data. The result of the experiment using the network applied on late fusion of letters "o" (DL_EARLY_FUSION_C_CROP_O) is less accurate because the character extractor makes more mistakes (e.g., "q", "d", "p" or "b" may be interpreted as "o"). One possible solution to avoid noisy data in the dataset is applying Optical Character Recognition (OCRs) in the images and discard non "o" letters, for example, but the dataset contains too tiny images and all OCR implementations tested failed to recognize the letters. Anyway, the network is able to cope with different input shapes, or data, and so probably different input fonts.

The statistical test using the Friedmann pre-test yielded the p-value of 2.32984×10^{-68}, helping us to state that the performance of the proposed methods have statistical significant difference. Table 4.2 shows the statistical Tukey HSD tests, showing that our proposed multi-representation early fusion approaches (namely DL_EARLY _FUSION_C_CROP_A and DL_EARLY_FUSION_C_CROP_E) are statistically significant when compared to all the unique representation approaches.

With these findings, we conclude that the multiple representation approach is useful for laser printer attribution using deep networks, as the different image transformations

Rank	Method	DL_EARLY_FUSION_C_CROP_A	DL_EARLY_FUSION_C_CROP_E	DL_NATURAL_IMAGE_C_CROP_E	DL_NOISE_AVERAGE_C_CROP_A	DL_NOISE_AVERAGE_C_CROP_E	DL_NOISE_IMAGE_MEDIAN_C_CROP_E	DL_EARLY_FUSION_C_CROP_D	DL_NOISE_IMAGE_MEDIAN_C_CROP_A	DL_NATURAL_IMAGE_C_CROP_A	DL_EARLY_FUSION_C_CROP_O	DL_CTGF_3X3_C_CROP_E	DL_NOISE_IMAGE_WIENER_C_CROP_E	DL_CTGF_5X5_C_CROP_E	DL_EARLY_FUSION_C_CROP_FRAME	SCORE
1	**DL_EARLY_FUSION_C_CROP_A**	0	0	1	1	1	1	1	1	1	1	1	1	1	1	12
2	**DL_EARLY_FUSION_C_CROP_E**	0	0	1	1	1	1	1	1	1	1	1	1	1	1	12
3	DL_NATURAL_IMAGE_C_CROP_E	0	0	0	0	0	0	0	0	0	0	1	0	1	1	3
4	DL_NOISE_AVERAGE_C_CROP_A	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
5	DL_NOISE_AVERAGE_C_CROP_E	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
6	DL_NOISE_IMAGE_MEDIAN_C_CROP_E	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
7	**DL_EARLY_FUSION_C_CROP_D**	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
8	DL_NOISE_IMAGE_MEDIAN_C_CROP_A	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
9	DL_NATURAL_IMAGE_C_CROP_A	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
10	**DL_EARLY_FUSION_C_CROP_O**	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
11	DL_CTGF_3X3_C_CROP_E	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	-11
12	DL_NOISE_IMAGE_WIENER_C_CROP_E	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1
13	DL_CTGF_5X5_C_CROP_E	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	-11
14	**DL_EARLY_FUSION_C_CROP_FRAME**	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	-11

1 = Line method is better than column method
0 = Line method is equivalent to column method
-1 = Line method is worse than column method

Table 4.2: Tukey-HSD pairwise statistical tests considering CNN approaches that use unique and multiple representation of the same data. Multiple representation (early fusion) approaches are highlighted in bold.

may highlight in the network some areas in the letters where the banding effect and other telltales of printers may be present in the input data.

4.5.2 Comparison of Unique Data against Multiple Data

In this second set of the experiments, we compare the use of single data (with or without early fusion) and the use of multiple data with late fusion. In addition to the previous experiments, we now include four experiments, considering late fusion of different data. Table 4.3 shows these results.

Method	FINAL ACCURACY (ACC) AFTER 5X2 CROSS VALIDATION ON 28x28 IMAGES		
	ACC (%)	σ	Data Considered
DL_LATE_FUSION_C_CROP_AE	**97.33%**	**0.00**	**Letters 'a' and Letters 'e'**
DL_EARLY_FUSION_C_CROP_A	96.89%	0.00	Letters 'a'
DL_LATE_FUSION_C_CROP_AED	**96.87%**	**0.00**	**Letters 'a', Letters 'e' and letters 'd'**
DL_EARLY_FUSION_C_CROP_E	96.84%	0.00	Letters 'e'
DL_LATE_FUSION_C_CROP_AEO	**96.24%**	**0.03**	**Letters 'a', Letters 'e' and letters 'o'**
DL_NATURAL_IMAGE_C_CROP_E	96.13%	0.00	Letters 'e'
DL_NOISE_AVERAGE_C_CROP_A	94.89%	0.30	Letters 'a'
DL_NOISE_AVERAGE_C_CROP_E	94.50%	0.03	Letters 'e'
DL_NOISE_IMAGE_MEDIAN_C_CROP_E	94.30%	0.01	Letters 'e'
DL_EARLY_FUSION_C_CROP_D	93.67%	0.03	Letters 'd'
DL_NOISE_IMAGE_MEDIAN_C_CROP_A	93.34%	0.02	Letters 'a'
DL_NATURAL_IMAGE_C_CROP_A	93.07%	0.03	Letters 'a'
DL_EARLY_FUSION_C_CROP_O	92.21%	0.03	Letters 'o'
DL_CTGF_3X3_C_CROP_E	89.12%	0.03	Letters 'e'
DL_LATE_FUSION_C_CROP_AEFRAME	**88.72%**	**0.02**	**Letters 'a', Letters 'e' and Frames**
DL_NOISE_IMAGE_WIENER_C_CROP_E	84.84%	0.30	Letters 'e'
DL_CTGF_5X5_C_CROP_E	83.15%	0.06	Letters 'e'
DL_EARLY_FUSION_C_CROP_FRAME	73.69%	0.05	Frames

Table 4.3: Results considering unique and multiple data. Multiple representation and multiple data (late fusion) approaches are highlighted in bold.

As Table 4.3 shows, the approaches using late fusion of different representations also outperform the ones using only the same data. The multiple data works because, with the advantages of the multiple representation discussed before, different data (letters) can contain more explicit banding patterns than using the same letter (data). We also found that the best late fusion technique consists of using letters "a" and "e". This can also be explained by the fact that more voters are considered in the analysis.

The statistical test using the Friedmann pre-test yielded the p-value of 7.52173×10^{-101}, helping us to state that the approaches have statistical significance. Table 4.4 shows the statistical Tukey HSD tests, showing that our proposed multi-representation early fusion approaches (namely DL_LATE_FUSION_C_CROP_AE) is statistically significant when compared to all the single representation using single data approaches, but it is not significant to early fusion of single data approaches.

Rank	Method	DL_LATE_FUSION_C_CROP_AE	DL_EARLY_FUSION_C_CROP_A	DL_LATE_FUSION_C_CROP_AED	DL_EARLY_FUSION_C_CROP_E	DL_LATE_FUSION_C_CROP_AEO	DL_NATURAL_IMAGE_C_CROP_E	DL_NOISE_AVERAGE_C_CROP_A	DL_NOISE_AVERAGE_C_CROP_E	DL_NOISE_IMAGE_MEDIAN_C_CROP_E	DL_EARLY_FUSION_C_CROP_D	DL_NOISE_IMAGE_MEDIAN_C_CROP_A	DL_NATURAL_IMAGE_C_CROP_A	DL_EARLY_FUSION_C_CROP_O	DL_CTGF_3X3_C_CROP_E	DL_LATE_FUSION_C_CROP_AEFRAME	DL_NOISE_IMAGE_WIENER_C_CROP_E	DL_CTGF_5X5_C_CROP_E	DL_EARLY_FUSION_C_CROP_FRAME	SCORE
1	**DL_LATE_FUSION_C_CROP_AE**	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
2	DL_EARLY_FUSION_C_CROP_A	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	12
3	**DL_LATE_FUSION_C_CROP_AED**	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
4	DL_EARLY_FUSION_C_CROP_E	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	11
5	**DL_LATE_FUSION_C_CROP_AEO**	-1	0	-1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	2
6	DL_NATURAL_IMAGE_C_CROP_E	-1	0	-1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	2
7	DL_NOISE_AVERAGE_C_CROP_A	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
8	DL_NOISE_AVERAGE_C_CROP_E	-1	-1	-1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1
9	DL_NOISE_IMAGE_MEDIAN_C_CROP_E	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
10	DL_EARLY_FUSION_C_CROP_D	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
11	DL_NOISE_IMAGE_MEDIAN_C_CROP_A	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
12	DL_NATURAL_IMAGE_C_CROP_A	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
13	DL_EARLY_FUSION_C_CROP_O	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	-1
14	DL_CTGF_3X3_C_CROP_E	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	0	0	-14
15	**DL_LATE_FUSION_C_CROP_AEFRAME**	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	-12
16	DL_NOISE_IMAGE_WIENER_C_CROP_E	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	-1
17	DL_CTGF_5X5_C_CROP_E	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	0	0	-14
18	DL_EARLY_FUSION_C_CROP_FRAME	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	0	0	-14

1 = Line method is better than column method
0 = Line method is equivalent to column method
-1 = Line method is worse than column method

Table 4.4: Tukey-HSD pairwise statistical tests considering CNN approaches that use unique and multiple data. Multiple representation and multiple data (late fusion) approaches are highlighted in bold.

4.5.3 Comparison to Existing Techniques in the Literature

Table 4.5 shows the results of the 5×2 cross validation experiments considering our best approaches and existing counterparts in the literature. It can be seen from this Table that the first proposed method that outperforms the state of the art is the one that uses multiple representations of the letter "e" (DL_EARLY_FUSION_C_CROP _E), classifying a mean of three more documents in each fold of the cross validation, this means a 31% less error than the best state of the art: CTGF_GLCM_MD_MS_C_CROP_E.

When using a different letter rather than "e", such as the letter "a", we also see an improvement in results. The multiple representation of the letter "a"

(DL_EARLY_FUSION_C_CROP _A) classifies a mean of four more documents in each fold when compared to the best state of the art technique. The multiple representation of multiple data "a" and "e" (DL_LATE_FUSION_C _CROP_A_E) shows its efficacy by showing the best overall accuracy of 97.33%, classifying six more documents than the best state of the art on average. The reason for this good performance relies on the fact that this method takes into account multiple data with different banding artifacts on letter areas that can be better highlighted using different representations in deep networks.

Method	FINAL ACCURACY (ACC) AFTER 5X2 CROSS VALIDATION ON 28x28 IMAGES		
	ACC (%)	σ	Data Considered
DL_LATE_FUSION_C_CROP_AE	**97.33%**	**0.00**	**Letters 'a' and Letters 'e'**
DL_EARLY_FUSION_C_CROP_A	**96.89%**	**0.00**	**Letters 'a'**
DL_LATE_FUSION_C_CROP_AED	**96.87%**	**0.00**	**Letters 'a', Letters 'e' and letters 'd'**
DL_EARLY_FUSION_C_CROP_E	**96.84%**	**0.00**	**Letters 'e'**
CTGF_GLCM_MD_MS_C_CROP_E [5]	96.26%	0.00	Letters 'e'
GLCM_MDMS_C_CROP_E [5]	94.30%	0.01	Letters 'e'
GLCM_MD_C_CROP_E [5]	91.08%	0.00	Letters 'e'
HOG_C_CROP_E [140]	90.59%	0.02	Letters 'e'
LBP_C_CROP_E [138]	88.66%	0.01	Letters 'e'
RECONST_ERROR_C_CROP_E [86]	78.90%	0.02	Letters 'e'
GLCM_C_CROP_E [69,54]	77.87%	0.04	Letters 'e'
CTGF_3X3_E [5]	72.46%	0.03	Letters 'e'

Table 4.5: Experiments results comparing the proposed methods against literature solutions after 5×2 validation. Proposed methods (early and late fusion) are the ones in bold.

To validate the efficacy of the proposed methods, we also performed statistical tests in the f-measures of the 5×2 confusion matrices. The Friedmann test showed a p-value of 3.16408×10^{-138}, which helps us to state that there is statistical significance among all the approaches and Table 4.6 shows the Tukey-HSD pairwise tests.

We now turn our attention to the classification of each printer individually in the experiment using 28×28 letters. Tables 4.7 and Table 4.8 show confusion matrices representing the classification accuracies per printer from our best proposed approach and from the best existing method in the literature CTGF_MD_MS_C_CROP_E.

In Table 4.7, the confusion matrix of the proposed method shows that it was able to identify 100% of three out of ten printers used in the experiments. These printers are Canon MF4370DN, OKI Data C330DN and Samsung CLP315. The CTGF_MD_MS_C_CROP_E confusion matrix in Table 4.8, on the other hand, shows 100% classification for only one printer, the OKI Data C330DN.

It is also remarkable to discuss the fact that we are using two printers of same model and brand (H225A and H225B) and it is possible to see in Tables 4.7 and 4.8 that there are just some misclassifications between them. This can be considered normal and happens for both approaches as the banding generated by the two printers can be very similar for some documents. Anyway, our proposed approach misclassified a mean of 6.6% of the documents in these two classes and the best existing method did it for 7.7% of the documents.

It is also important to note that there are some unusual misclassifications between printers H1518 (an HP printer) and C1150 (a canon printer) when using our best proposed approach and also the best state of the art. The misclassifications may happen for two reasons: (i) for letters printed bigger than others, important information regarding some

Rank	Method	DL_LATE_FUSION_C_CROP_AE	DL_EARLY_FUSION_C_CROP_A	DL_LATE_FUSION_C_CROP_AED	DL_EARLY_FUSION_C_CROP_E	CTGF_GLCM_MD_MS_C_CROP_E [5]	GLCM_MDMS_C_CROP_E [5]	GLCM_MD_C_CROP_E [5]	HOG_C_CROP_E [140]	LBP_C_CROP_E [138]	RECONST_ERROR_C_CROP_E [86]	GLCM_C_CROP_E [69,54]	CTGF_3X3_E [5]	SCORE
1	**DL_LATE_FUSION_C_CROP_AE**	0	0	0	0	1	1	1	1	1	1	1	1	8
2	**DL_EARLY_FUSION_C_CROP_A**	0	0	0	0	0	1	1	1	1	1	1	1	7
3	**DL_LATE_FUSION_C_CROP_AED**	0	0	0	0	1	1	1	1	1	1	1	1	8
4	**DL_EARLY_FUSION_C_CROP_E**	0	0	0	0	0	1	1	1	1	1	1	1	7
5	CTGF_GLCM_MD_MS_C_CROP_E [5]	-1	0	-1	0	0	0	1	1	1	1	1	1	4
6	GLCM_MDMS_C_CROP_E [5]	-1	-1	-1	-1	0	0	0	0	1	1	1	1	0
7	GLCM_MD_C_CROP_E [5]	-1	-1	-1	-1	-1	0	0	0	0	1	1	1	-2
8	HOG_C_CROP_E [140]	-1	-1	-1	-1	-1	0	0	0	0	1	1	1	-2
9	LBP_C_CROP_E [138]	-1	-1	-1	-1	-1	-1	0	0	0	1	0	0	-5
10	RECONST_ERROR_C_CROP_E [86]	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	-9
11	GLCM_C_CROP_E [69,54]	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	-8
12	CTGF_3X3_E [5]	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	-8

1 = Line method is better than column method
0 = Line method is equivalent to column method
-1 = Line method is worse than column method

Table 4.6: Tukey-HSD pairwise statistical test results comparing the proposed methods against the state of the art considered. Proposed methods (early and late fusion) are the ones in bold.

borders may be removed during the cropping operation used to fit the images to the input layer of the network and (ii) some printers may share similar firmwares with other manufacturers, as we suspect Canon does, and this can yield similar printing patterns in some documents.

BEST PROPOSED		Attributed Printer									
		B4070	C1150	C3240	C4370	H1518	H225A	H225B	LE260	OC330	SC315
Actual Printer	B4070	99.50		0.33	0.17						
	C1150	0.52	99.48								
	C3240	0.67		98.83	0.50						
	C4370				100.00						
	H1518	0.33	10.50			89.17					
	H225A						93.10	6.90			
	H225B	0.18					6.37	93.45			
	LE260			0.17			0.33		99.50		
	OC330									100.00	
	SC315										100.00

Table 4.7: Confusion Matrix of the best proposed approach showing, in percentages, the right and wrong mean hits per printer after the 5 × 2 cross validation.

4.6 Final Considerations and Further Developments

Laser printer attribution is a difficult task because it involves investigating several printing patterns, created by different manufacturing processes of hundreds of printer models and brands. Several solutions in the scientific literature use computer vision and machine learning algorithms in the scanned version of the documents, aiming at finding intrinsic

CTGF_GLCM_MD_MS_C_CROP		Attributed Printer									
		B4070	C1150	C3240	C4370	H1518	H225A	H225B	LE260	OC330	SC315
Actual Printer	B4070	98.67	0.33	1.00							
	C1150	1.72	98.28								
	C3240			97.83	2.17						
	C4370		1.00	0.50	98.50						
	H1518	1.33	10.33			86.83	0.50			0.84	0.17
	H225A		0.50				96.98	2.52			
	H225B						12.90	87.10			
	LE260		0.67	0.50			0.17		98.66		
	OC330									100.00	
	SC315		0.50		0.33						99.17

Table 4.8: Confusion Matrix of the best literature solution showing, in percentages, the right and wrong mean hits per printer after the 5×2 cross validation.

signatures in the printing material that better discriminate different printers. However, the problem with these approaches is that they use what is called *custom-engineered features*, applicable only in specific situations and are based on an initial suspicion of the artifact behavior (*e.g.* the texture is specific, there is a geometric distortion in the printed material, etc.).

In this chapter we propose the first solution able to *learn* features directly from the data from laser printer intrinsic signatures present in the printed paper. Our solution takes into account the benefits of deep learning networks and backpropagation procedures, with the latter evolving the descriptor in the end of the training by fine-tuning the filter weights used to recognize printing patterns of different filters. Our approach investigates the artifacts in different letters of documents in different languages by using others letters rather than "e", and we also use different data representation by applying some image transformations, which are able to better highlight areas in the letters where the banding is present.

As we discussed throughly in this chapter, multiple representations of multiple data outperforms the state of the art when dealing with the laser printer attribution problem. Multiple representations used as input to such deep networks are important because a given image transformation applied in a deep network can have the banding in letter areas better highlighted in letter areas through convolutional layers of the network, and this can help the analysis of other representations used in other deep networks when they are considered toghether. We also showed that multiple representations of multiple data is the best choice for laser printer attribution on deep networks, as, with the benefits of the multiple representations presented before, multiple data considers more data/voters to be analyzed with different printing patterns. However, using deep learning in laser printer attribution as shown in this chapter has one drawback: as the input for the first convolutional layer requires a fixed size image, the cropping performed in the letters used to validate the proposed approach can erase some artifacts in letters borders that may confuse a little bit the classification of bigger letters.

As future work we aim at developing deep networks to be applied in another kind of data, such as bigger letters and frames. Also, we believe that other different representations can be taken into account in the investigation and we also plan to study the behavior of the proposed approaches in interpolated data. Finally, our future investigations are also aimed at fusing non-handcrafted features with another handcrafted features in a machine

learning classifier and proposing a two-tier deep network, which is a deep network that can deal with features from different deep networks applied on our different representations of different data.

Part III

Multi-Analysis Solutions for Tampering Detection

Chapter 5

Multi-Analysis Solutions for Median Filtering Detection

In this chapter, we aim at detecting traces of image tampering through median filtering by a set of *multi-analysis* approaches. These approaches apply multiple perturbations to questioned images using multiscale filters, acting in the pre-processing step of the analysis pipeline. These approaches seek to highlight a structural inconsistency, called *streaking*, in questioned images.

5.1 Motivation

Recently, image doctoring has been made easier by a range of cheap and easy-to-use digital image editing software packages with effective algorithms aimed at reducing the artifacts (visible or non-visible) left behind in the manipulated images. Although image adjustments allow us to properly correct images of familiar members in an innocent party, it also can be used for negative effects such as defaming politicians (e.g., Sarah Palin's case[1]), showing photographs of non-existent military power to the citizens (e.g., Iranian missiles case[2]), deceiving insurance companies by multiplying or creating damages in digital photos of properties[3], among others [157]. Therefore, the development of reliable tools to fight misinformation is paramount.

A digital image tampering method commonly used is the image re-sampling, which can be helpful to make copy-and-move forgery operations more convincing. Copy-and-move operations allied with sophisticated re-sampling operations allow a forger to change the size of multiple copies of an object, making them closer or farther away. A way of detecting the presence of resampling is through the analysis of its artifacts left behind. Popescu and Farid [96] noted that re-sampling operations use interpolation techniques (which results in one image with pixels correlated in some way) and proposed an Expectation-Maximization technique for finding periodic samples of the image and detecting re-sampling operations. A particular problem of this technique is the assumption of a linear correlation of the

[1]http://www.nytimes.com/2004/03/11/technology/the-camera-never-lies-but-the-software-can.html

[2]http://thelede.blogs.nytimes.com/2008/07/10/in-an-iranian-image-a-missile-too-many/

[3]http://www.economist.com/news/technology-quarterly/21572915-digital-imaging-insurers-publishers-law-enforcement-agencies-and-dating-sites-are

pixels. As stated by Kirchner and Bohme [97], a non-linear filter such as the median filter can destroy these re-sampling artifacts by replacing each pixel with the median-valued pixel within a neighborhood therefore rendering resampling undetectable by Popescu and Farid [96]'s technique.

Median filtering has several applications in the context of forensics. From hiding traces of resampling to enhancing image retouching, being useful to remove imperfections on images in digital editors such as Photoshop [93]. From concealing forgery artifacts to fooling detection techniques such as the ones proposed by Johnson and Farid [94]'s work and its extension by Saboia et al. [95] (which detect image manipulation by means of the eye specular highlights in images containing people) not counting hiding traces of JPEG blocking as in Stamm et al [39].

In this chapter, we propose multi-scale and multi-directional median filtering detection algorithms based on multi-perturbations in the input image. These algorithms are originated by the hypothesis that the median filtering *streaking* artifacts affect the Image Quality Metrics (IQMs) of median filtered images in a different way under multi-scale filterings (filtering with different regions of interest) and over progressive perturbations (henceforth perturbations are defined as cascade-wise successive image filterings). The proposed techniques are multi-scale because different filter sizes are used to perform the perturbations. The techniques proposed in this chapter are also multidirectional because, for each filter size, a different neighbor pixel can be used to replace the pixel in the center of the window, so the streaking artifacts are propagated in different ways depending on the mask size, image content and the region where the sliding window is located. In our novel approaches, we evaluate several image quality metrics upon each perturbed image and build a highly discriminative feature space for classification. Experiments with complex datasets show that the proposed method outperforms state-of-the-art solutions without assuming anything about the underlying filtering process of the input images.

We organized the rest of this chapter in four sections. In Section 5.2, we introduce our novel approach, based on multi-scale and multi-directional perturbations to detect median filtering operations in digital images. In Section 5.3, we discuss about the experimental setup used to validate the proposed methods. In Section 5.4, we present the experimental results that validate the proposed methods and also to compare them with state-of-the-art counterparts. Finally, in Section 5.5, we conclude the chapter and discuss some possible future work.

5.2 Proposed Method

Our technique is inspired on the observation of what happens in text file compression using, for example, the Run-Length encoding algorithm [158]. When a text file is compressed for the first time, its size will decrease because there are redundancies in the text used in the compression (for example, a text file with AAAABBB will become 4A3B, in other words, a file with 7 characters is represented by 4 characters in the compressed file). However, when a second compression is applied to an already compressed file, chances are the file size will increase if compared to the previous compression (in the same example, 4A3B will

become 141A131B). This happens because there are much less or no redundant elements to compress, leading to a file size increase compared to the previous compression.

Our multi-perturbation approach is also inspired by the Rocha and Goldenstein [159] steganalysis detection technique. In their work, they propose to perform progressive insertion of hidden messages (this can be also regarded as perturbations) in digital images. Then, in regions of interest, statistical descriptors are used to feed a classifier able to detect if hidden messages exist in digital images. The authors realized that pristine and stego images suffer different behavior when are disturbed (in this case, with hidden messages).

We found that the same happens in median filtered images: they also show different behavior when they are disturbed. These blurred images suffer different degradation when compared to pristine images after a series of successive perturbations (median filtering operations). These perturbations will highlight the streaking artifacts and can be detected or not by measuring the degradations of the disturbed images when compared to the input image.

Hence, we propose in this chapter a novel technique based on multi-directional and multi-scale multiple perturbations in the image, measuring the image degradations by means of image quality metrics. We propose the use of multi-scale median filtering masks because, when applying the multiple perturbations on multiple median mask sizes, we are able to find groups of streaking (redundant) pixels no matter which median filter mask size was used to originally blur the image. The proposed technique can be regarded as multi-directional because, for each mask size, a different neighbor pixel can be used to replace the pixel in the center of the window, so the streaking artifacts are propagated in different ways, depending on the mask, image content and the region where the sliding window is located. The multiple perturbations are done to propagate the streaking artifacts and to make them more detectable to our developed metrics.

To detect image median filtering in our proposed approach, we build a feature vector to be used by machine learning classifiers that encodes the median and non-median information present in the image. For that, we perform f progressive filterings with m median filtering masks with different $n \times n$ dimensions in the input image. We consider median filtering windows (regions of interest) of size 3×3, 5×5, 7×7 and 9×9. In other words, we define $n = \{3, 5, 7, 9\}$, $1 \leq m \leq 4$ and $1 \leq f \leq 5$. After each filtering, we measure how the perturbed image was degraded using the input image for comparison. Fig. 5.1 depicts the workflow of the proposed technique.

As Figure 5.1 shows, the proposed method works by progressively blurring f times a set of training images using m mask sizes, comparing each filtered image with the input image by using $q = 8$ IQMs. All the IQMs calculated are combined to yield a final description (feature) vector $\vec{F} \in \mathbb{R}^{f \times m \times q}$. Then, a machine learning classifier learns the behavior of the median and pristine images and, given a test image, the approach repeats the operation and the classifier labels the new images based on what it has learned during training. Algorithm 1 shows how the approach works for a given window size. Basically, for the first perturbation, the IQMs extraction happens at the first blurring of the image, using the input image as the reference image. For the second perturbation onwards, the blurred image from the previous perturbation is blurred again and the IQMs extraction takes place in a similar way. To apply Algorithm 1 to multiple scales, it is necessary to apply it to

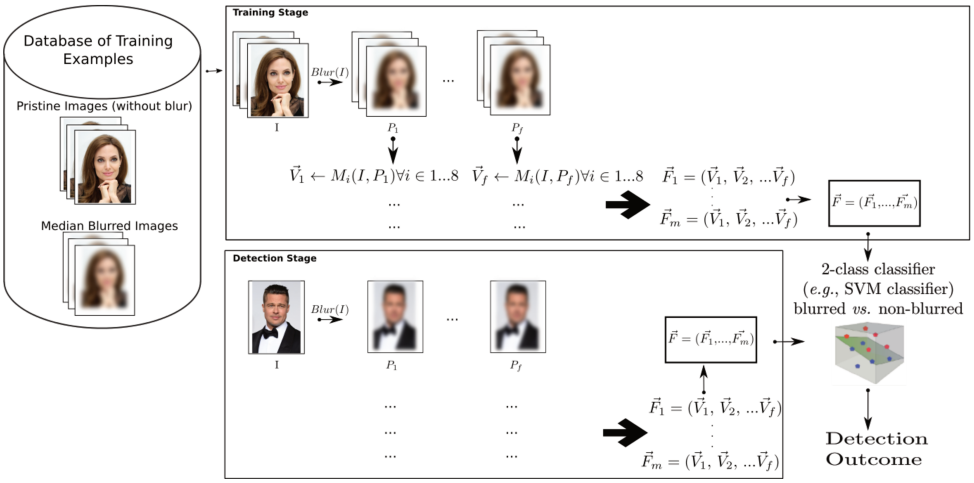


Figure 5.1: Proposed technique to detect median filtering. In the training phase (top), the image quality metrics are calculated per image after f progressive perturbations using m scales of the median filter mask. All these metrics are then combined to yield a final feature vector. Feature vectors of all training images are finally used to build a machine learning classifier. In the testing step (bottom), the same procedure is done but now the feature vector is classified as pristine or median filtered image, according to the trained classification model.

each scale individually and concatenate all vectors at the end to yield the final feature vector used to feed an SVM classifier.

Algorithm 1: Algorithm to extract image quality metrics after a set of progressive and multiscale image perturbations in the input image.

function Calculate_Features (I, F, f, q);
Input : **I**: Input image to extract feature vector.
F: Multiscale median filters $F = \{F_1, ..., F_m\}$.
f: Number of Progressive Median Filtering Operations.
q: Number of IQMs to Calculate at Each Perturbation.
Output : **V**: Feature Vector with Progressive and Multiscale Perturbations Image Quality Measures.
//1. Perturbations: Perform f progressive median filterings using f filter scales
$\{P_{ij}\}_{j=1,...,m}^{i=1,...,f} =$
$\{T_1(I, F_1), ..., T_f(T_{f-1}(I, F_1), F_1), ..., T_1(I, F_m), ..., T_f(T_{f-1}(I, F_m), F_m)\}$
//2. IQM calculation: for the $i-th$ perturbation using $j-th$ filter scale, calculate q IQMs
$\{V_{ijk}\} = \{d_k(P_{ij})\}, i = 1, ..., f, j = 1, ..., m, k = 1, ..., q$

As an example, given an image I, for each of m mask sizes used, the method performs f successive median filterings on such image with m different window sizes and q image quality metrics. For each window size m, the method successively filters the image f times, comparing the outcome with the input image, measuring the distortion through the selected $q = 8$ IQMs. The result for each image is a feature vector $\vec{F} \in \mathbb{R}^{f \times m \times q}$ that is used by a machine learning classifier.

The multi-analysis procedure performed herein considers several directions (multidirectionality) in the filtering (the filter replaces the central pixel by the median value in the neighborhood, and the neighbor used to replace the value of the central pixel can be at any direction), applying multiple and progressive filterings to the image (multiple-perturbations). If the multiple perturbations use several scale filters the multi-scale multi-analysis scenario is considered. The reason for the use of these multi-analyses operations lies in the fact that multiple perturbations simulate the presence of the streaking artifact in the suspected images and, if this artifact already exists in the image, it is highlighted differently by the perturbation, affecting the image quality differently in pristine and tampered images. This procedure happens in the pre-processing step of the analysis pipeline, as the image is changed by the multiple perturbations for the description (image quality metrics calculation).

IQMs [160, 161, 162] have been successfully employed in the literature and are described in terms of the visibility of the distortions, such as color shifts, blurriness, Gaussian noise and blockness [160]. The most common way of creating an image quality metric is quantifying the visibility of these distortions by comparing a distorted image to a reference one. They were already used before in digital image forensics. Avcibas et al. [163] used four different image quality metrics and a classifier to detect traces of image manipulation. The authors also used such IQMs to detect hidden messages in digital images [164]. Differently from these approaches, we explore the effects of the proposed perturbations to highlight streaking artifacts by using IQMs and then build a highly discriminative feature space suitable for detecting median filtering in digital images.

To measure the streaking artifacts in our proposed technique, we use $q = 8$ bivariate image quality metrics per perturbation and window mask: Mean Squared Error, Peak Signal to Noise Ratio, Structural Content, Average Difference, Maximum Difference, Normalized Cross Correlation, Normalized Absolute Error and Structural Similarity. We chose these eight quality measures because they were proven to be efficient in measuring image degradation in previous works [160, 161, 162].

Mean Squared Error (MSE): this IQM measures the mean of pixel differences (hereinafter referred to as error) between an ideal image I and its distorted image K.

$$MSE(I,K) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} D(I,K), \tag{5.1}$$

where $D(I,K) = [I(i,j) - K(i,j)]^2$ and M and N are the dimensions of the two images I and K.

Peak Signal to Noise Ratio (PSNR): this IQM is the ratio between the maximum possible power of the ideal image I and the power of corrupting noise that affects it in image K. PSNR is usually expressed in terms of the logarithmic decibel scale and typical values for JPEG images are between 30 and 50 dB, where higher is better. This metric is

calculated as:

$$PSNR(I,K) = 20 \times \log_{10}\left(\frac{max(I)}{\sqrt{MSE(I,K)}}\right). \tag{5.2}$$

Structural Content (SC): this IQM calculates the ratio of squared sum of pixels in the original image I and in its distorted version, K,

$$SC(I,K) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j)^2}{\sum_{i=1}^{M}\sum_{j=1}^{N} K(i,j)^2}. \tag{5.3}$$

Average Difference (AD): this IQM measures how the pixels change in a distorted image K when compared to its ideal image I by calculating the mean of errors (the error is not squared as in MSE) between pixels of the two images:

$$AD(I,K) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j) - K(i,j)}{M \times N}. \tag{5.4}$$

Maximum Difference (MD): this IQM measures the highest error found in a distorted image K related to its ideal image I:

$$MD(I,K) = max(|I(i,j) - K(i,j)|. \tag{5.5}$$

Normalized Cross Correlation (NCC): this IQM measures the similarity between two dimensional signals I and K:

$$NCC(I,K) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j) \times K(i,j)}{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j)^2}. \tag{5.6}$$

Normalized Absolute Error (NAE): this IQM calculates the normalized error between I and K, defined as the ratio of sum of pixel differences (error) between the ideal and distorted image and the sum of pixels in the ideal image I.

$$NAE(I,K) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j) - K(i,j)}{\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j)}. \tag{5.7}$$

Structural Similarity (SSIM): this IQM measures the change of the structural information of the image, which are the inter-dependencies of close pixels. These dependencies carry important information about the structure of the objects in the visual scene. The SSIM metric is applied on various windows of the investigated images as:

$$SSIM(x \in I, y \in K) = \frac{(2 \times u_x \times u_y + c_1) \times (2 \times \sigma_{xy} + c_2)}{(u_x^2 \times u_y^2 + c_1) \times (\sigma_x^2 \times \sigma_y^2 + c_2)}, \tag{5.8}$$

where x and y are $n \times n$ (typically 8×8) image patches from I and K respectively, u_x and u_y are the mean pixel values of patches x and y, σ_x^2 and σ_y^2 are the same for variance, σ_{xy} is the covariance of x and y and c_1 and c_2 are constants that are calculated depending on the bits per pixel of the images. The calculation can be done in just one window in both images or in a subgroup of them. When using the latter approach, the mean of SSIMs is reported as the final SSIM. The resultant SSIM is a value between -1 and 1, and the value 1 occurs when I and K are the same image.

5.3 Experimental Setup

In this section, we discuss the used benchmarks, the experimental methodology, the state-of-the art approaches used for comparison and statistics used to compare all studied methods.

5.3.1 Benchmarks

The experiments considered four benchmarks. The first one comprises 3,996 JPEG images from the *Chinese Academy Image Tampering Database* [5]. These images are compressed, have similar lighting conditions, low resolution and most of them were taken with the same camera. Here, 1,998 images are pristine and 1,998 are median filtered with a 3×3 Matlab median filter implementation. We use this dataset, referred to as CASIA_COMP, for finding the best parameters of the proposed technique, to compare the proposed methods to the state of the art in a cross-validation scenario and to train the classifier in a cross-dataset scenario used in the experiments.

The second benchmark contains 800 JPEG images from a personal image dataset. It comprises 800 JPEG images collected with very different resolutions, camera noise, lighting conditions and compressing factors. These images were taken from different dedicated cameras and smartphones. It is used to test the classifier trained with compressed images from the previous database. Here, 400 images are pristine and 400 are median-blurred images with different window masks (3×3, 5×5, 7×7, 9×9). These median blurrings were performed using four different image processing tools: Matlab, OpenCv, Gimp, and Photoshop. Table 5.1 shows the devices used to acquire the images of this database, which is referred to as COMPLEX.

The third benchmark comprises 2,773 uncompressed images from CASIA [5] (we refer to this benchmark as CASIA_UNCOMP) and the fourth benchmark contains 1,338 uncompressed images from Uncompressed Image Database (we refer to this benchmark as UCID) [165]. We use the images from CASIA_UNCOMP to find the best parameters of the proposed technique, to compare the proposed methods against the state of the art in a cross-validation scenario and to train the classifier in the cross-dataset scenario used in the experiments. The images from UCID are used to test the classifier in the cross-dataset scenario. A total of 8,907 images are used for validation. Three out of four benchmarks are freely available at their original websites but are also freely available at[4] and the source

[4]http://www.recod.ic.unicamp.br/∼anselmo/median-detection-dataset

Type	Brand	Model
Camera	Olympus	C120
Camera	Olympus	C150
Camera	Olympus	Style Tough
Camera	Canon	EOS 50D
Camera	Canon	Powershot A540
Camera	Panasonic	DMC-FS3
Smartphone	Nokia	N8
Smartphone	Sony	X10A
Smartphone	Motorola	EXZ

Table 5.1: Cameras and Smartphones used to acquire images of the COMPLEX benchmark.

code of the proposed approaches can be found at GitHub[5]

5.3.2 Experimental Methodology

To find the best parameters of the proposed technique and also to compare it against the state of the art, we chose two experimental protocols, one is the 5×2 cross-validation and the other is the cross dataset.

In the 5×2 cross validation protocol, for each one of the five rounds, the data is randomly divided, with 50% of the data used as the training set of the classifier and the other 50% of data used for testing. Then, the process is inverted: the data used for testing is used for training and the data used for training is used for testing. This process is repeated five times (five rounds). In the end, 10 experiments of training and testing of the classifier are performed. This is regarded as an optimal benchmarking protocol for learning algorithms [138]. We used data from the CASIA_COMP and CASIA_UNCOMP dataset in this experiment.

The second form of validation considers the cross-dataset protocol, a more real-world situation, whereby the training data is known and come from one database and the testing dataset come from a different and unknown dataset collected by different people, with different acquisition conditions, parameters and settings. In this scenario, we use just one training (with the known data) and one test (with the unknown data). The training data here come from CASIA_COMP and CASIA_UNCOMP and the testing data came from COMPEX and UCID. This setup is close to a real-world situation, whereby the data used during operation of the method may come from completely different acquisition conditions.

5.3.3 State-of-the Art Methods Considered

We compare the proposed methods to some state-of-the-art approaches presented in Sec. 2.2.2:

1. Kirchner and Fridrich [99] with T = 3 and second order Markov Chains as described in their work, yielding a 686-d feature vector (hereinafter referred to as SPAM);

[5]https://github.com/anselmoferreira/median-filtering-detection

2. Yuan [101] in 3 × 3 blocks and a 44-d feature vector (hereinafter referred to as MFF); and

3. Chen et al. [103, 104] with parameters T=10, B=3 and K=2 and 56-dimensional feature vectors as described in their work (which we hereinafter referred to as GLF).

We use these three approaches because the SPAM and MFF source were available (in [166] and [167], respectively) and the papers that presents GLF [103, 104] were clear enough to allow a complete reproduction of the work.

5.3.4 Metrics and Statistical Tests

To compare the proposed method against the state of the art, we choose a set of standard metrics and conduct tests to identify if there is statistical significance in the reported results.

The first metric used is the classification accuracy. It measures the ratio of the number of correct positive (in our case, median filtered images) and negative (pristine images) classifications and the total set of testing data. It is calculated as

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP}, \tag{5.9}$$

where TP, TN, FP and FN are true positives, true negatives, false positives and false negatives respectively. We don't use the normalized accuracy here because the positive and negative examples in our data are always balanced.

The Sensitivity is the ratio of number of samples correctly classified as positive and the total number of positive samples in the testing data. It is also known as true positive rate and is calculated as

$$Sensitivity = \frac{TP}{TP + FN}. \tag{5.10}$$

The Specificity is the ratio of number of samples correctly classified as negative and the total number of negative samples in the testing data. It is also known as true negative rate and is calculated as

$$Specificity = \frac{TN}{TN + FP}. \tag{5.11}$$

The precision tells the percentage of correct positive classifications given all the positive classifications given by the classifier. It is calculated as

$$Precision = \frac{TP}{TP + FP}. \tag{5.12}$$

In the statistical significance tests, we first perform a pre-test to confirm if all techniques are statistically different. If so, a pairwise test compares one technique against another (also known as post-test). Each of these steps usually involves a statistical test and a confidence level. We consider a 95% confidence level for each test, which means that a p-value returned must be less than 0.05. The pre-test is used to determine if subjects

change significantly across occasions and conditions. We consider the ANOVA test to investigate the parameters of the proposed technique and the Friedmann test to cross validation and cross-dataset experiments. To compare the techniques against each other, we use the Tukey-Kramer approach (also known as Honestly Significant Difference (HSD)) in the investigation of the parameters of the proposed technique and also the McNemar's test in the cross-validation and cross-dataset experiments.

5.4 Experiments and Discussion

In this section, we show the experiments performed to validate the proposed technique. We present the experiments for finding the best parameters of the proposed technique, study the importance of the features proposed and compare it to the state of the art in a cross-validation and cross-dataset scenarios, using compressed and uncompressed images.

5.4.1 Tuning of Parameters

The minimal number of perturbations n and number of windows q of the proposed technique were found after statistical tests in a series of 5×2 cross-validation experiments in the CASIA_COMP [5] dataset. Here, we characterize the images as described in Sec. 5.2 and train an SVM classifier with an RBF kernel, whose parameters are automatically learned during training, according to the 5×2 cross-validation protocol used.

In this experiment, we fixed $q = 8$ image quality measures to be calculated at each filtering, comparing the output with the input image. The blurring is done progressively f times, where we vary f in $1 \leq f \leq 5$. We use m different scales of the median filtering mask, where $1 \leq m \leq 4$. With these, the total number of experiments performed to find the best parameters was 25. Table 5.2 shows the results of the best parameters (f, m). Table 5.3 shows the ANOVA statistical test results, whereby we investigate the *windows* and *perturbations* factors.

Name	Label	#Perturbations (f)	Windows (m)	Accuracy
Three Perturbations, Single Window	TPOW	3	3×3	98.8% ± 0.22
Four Perturbations, Multiple Windows	FPMW	4	3×3, $5{\times}5$, $7{\times}7$, $9{\times}9$	98.7% ± 0.29
Three Perturbations, Multiple Windows	TPMW	3	3×3, $5{\times}5$, $7{\times}7$, $9{\times}9$	98.7% ± 0.28

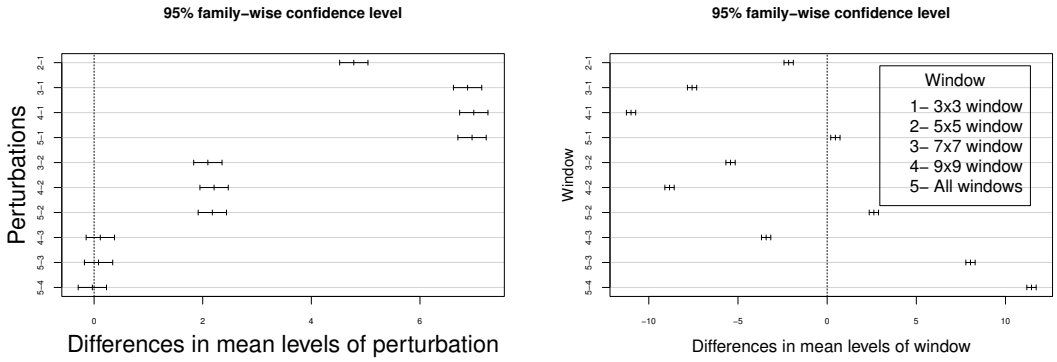
Table 5.2: Mean classification accuracy after a 5×2 cross-validation on CASIA dataset [5].

Factor	p-value
Window	2.2e-16
Perturbation	2.2e-16
Window x Perturbation	2.2e-16

Table 5.3: ANOVA p-value results in accuracy values for 25 experiments in the CASIA dataset [5].

The ANOVA statistical test results in Table 5.3 shows that varying the number of windows and perturbations are statistically significant (p-value < 0.05) and these factors

are correlated. Figure 5.2 depicts the results of Tukey-HSD tests for pairwise comparisons.



(a) (b)

Figure 5.2: (a) Tukey-HSD pairwise test comparison in factor perturbation (b) Tukey-HSD pairwise test comparison in factor window.

Fig. 5.2 shows that there is no statistical difference when using three and four, three and five and four and five perturbations. However, there is significant difference when using more than one perturbation. In addition, varying the window sizes is statistically significant according to Fig. 5.2. The ANOVA test in the three best algorithms yielded a p-value of 0.79, which helps us to state that the accuracy difference between these techniques is not statistically significant. Hence, we chose to use the three last configurations (namely TPOW, TPMW and FPMW) in the second part of the experiments.

5.4.2 Studying the Importance of Features

To justify the use of multiple scales and perturbations, we used the random forest classification technique. We used this classifier to investigate the importance of the used features after the training. We then show, in Figure 5.3, the features importance after training the classifier with the proposed TPMW configuration. Figure 5.3 shows that, if training the classifier with pristine images and 3×3 median filtered images (*i.e.*, images from database CASIA_COMP), the most important dimensions (or peaks) are located in the area of the first 24 dimensions (the same dimensions from the proposed TPOW) and are in the first scale of median mask (3×3 filter). These 24 features comprise eight image quality metrics calculated after three perturbations. We find that the three most important dimensions are the first, sixth and the fourteenth (*i.e.*, the Mean Squared Error of the first perturbation, the Maximum Difference of the first perturbation and the Maximum Difference of the second perturbation, respectively). The quality metrics calculated in other scales are not so important for the classification in this scenario. Hence, the use of multiple perturbations is justified here.

In another scenario, we investigate the dimensions importance when the training has pristine and blurred images with different median filtering windows scales (rather than only 3×3). We trained the Random Forest classifier with the proposed TPMW applied

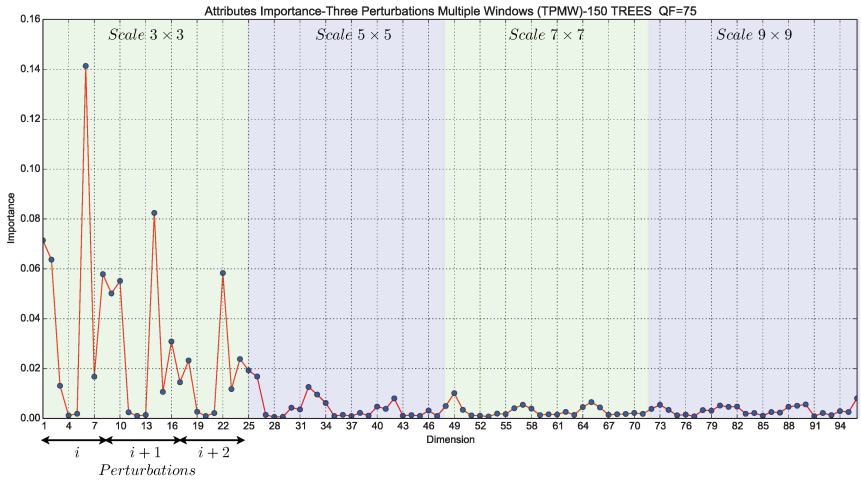


Figure 5.3: Importance of features after the training of a Random Forest classifier using the description from the cross validation compressed data. Here, the blurred images were smoothed by a 3×3 median filter window and the number of trees chosen was 150 as it yielded the best classification accuracy in the parameter search.

on our COMPLEX image dataset and the result of dimensionality importance analysis is shown in Figure 5.4. It is clear that the multiple scales are also important. We can see other peaks in the figure rather not seen in Figure 5.3. They can be found in the 39th, 55th and 90th dimensions (*i.e.*, the Structural Similarity of the second perturbation from the 5×5 filter mask, Structural Similarity of the first perturbation of the 7×7 filter mask and the Normalized Cross Correlation of the third perturbation of the of the 9×9 filter mask, respectively). This training scenario is more diverse because it better captures the variations of the real world (different cameras to acquire images, different compression settings, different image resolutions and different median filter masks using different implementations). Hence, the multiple perturbations and scales are justified.

5.4.3 Comparison with the State of the Art on a Cross-Validation Scenario

As discussed in section 5.3.2, the experiments are performed considering two evaluation protocols: cross validation and cross dataset. We perform the 5×2 cross validation in the proposed techniques and in the state-of-the-art methods. Figures 5.5, 5.6, and 5.7 show the classification accuracy results of the proposed techniques against the state of the art in the CASIA compressed dataset under a series of different compression settings. Here, half of the images were blurred with 3×3 mask. We also use two variations of the proposed technique based on classifier fusion: the first one uses voting of classifications of the three best proposed techniques (we denote this technique as VOTE) and the other technique proposed is the meta-fusion of the three best techniques. In other words, this last proposed technique uses the distance to three hyperplanes as a four-dimensional feature vector for classification (labeled here as META).

Note that although we present results here using the SVM classifier, any other classifi-

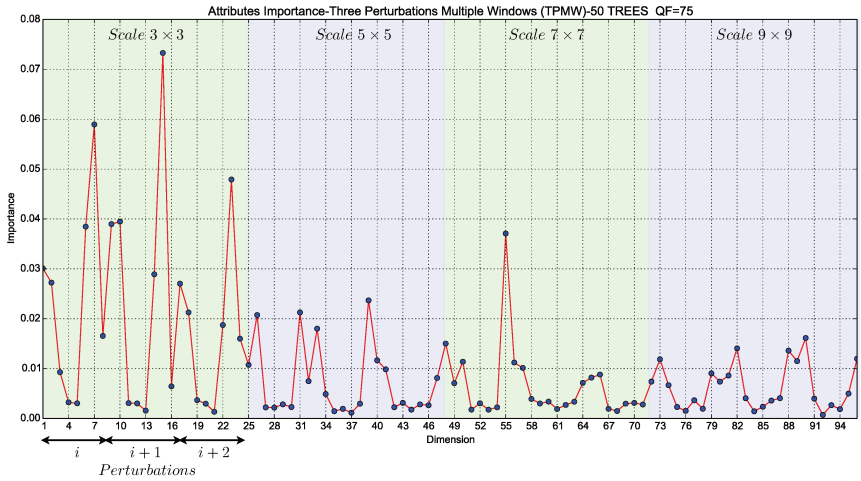


Figure 5.4: Importance of features after the training of a Random Forest classifier using the description from the cross-validation compressed data in dataset CASIA_COMP. Here, the blurred images were smoothed by a 3×3, 5×5, 7×7 and 9×9 median filter windows and the number of trees chosen was 50 as it yielded the best classification accuracy in the parameter search.

cation method could be used. The reason for choosing SVM is that most of the existing methods in the literature consider this method. Therefore we decided to keep it for a fair comparison regarding this possible factor. Using random forests, for instance, yielded similar results for our methods although slightly worse than SVM.

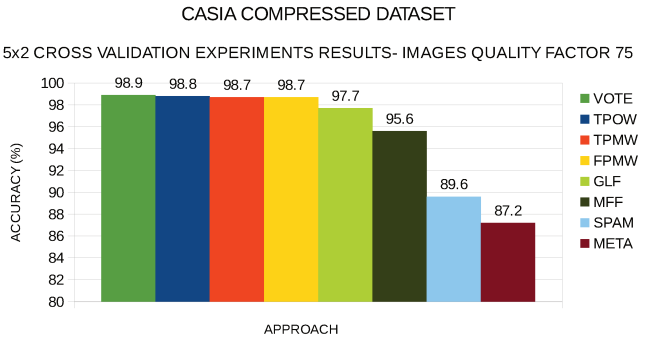


Figure 5.5: Mean classification accuracy after a 5 × 2 cross-validation comparing the proposed methods and the ones from literature. The database used is the CASIA compressed dataset with image quality factor 75.

As Figures 5.5, 5.6, and 5.7 show, the proposed techniques are the ones that achieve classification accuracies higher than 98% in most of the cases considered in this database. The reductions of errors if compared to the best state of the art in all compressed databases considered were 52%, 23%, and 51% respectively. It is expected that accuracies get worse by changing the compression settings, but even in this scenario, the proposed techniques showed a very small accuracy change and were still the best ones to detect the median filtering. We performed Friedmann statistical tests and there is statistical difference in the experiments related to the compression setting 75 (p-value$= 7.81 \times 10^{-12}$). Also, Tukey-HSD pairwise tests state that the best proposed method (VOTE) is statistically

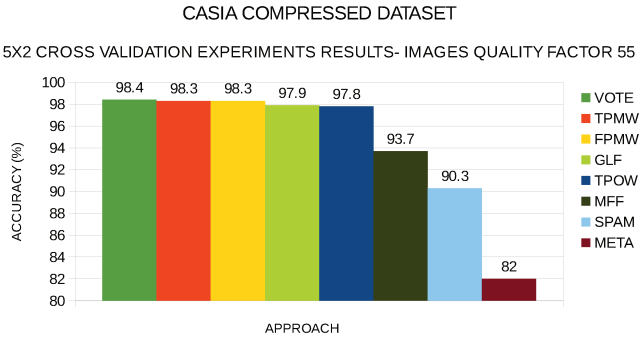


Figure 5.6: Mean classification accuracy after a 5×2 cross-validation comparing the proposed methods and the ones from literature. The database used is the CASIA compressed dataset with image quality factor 55.

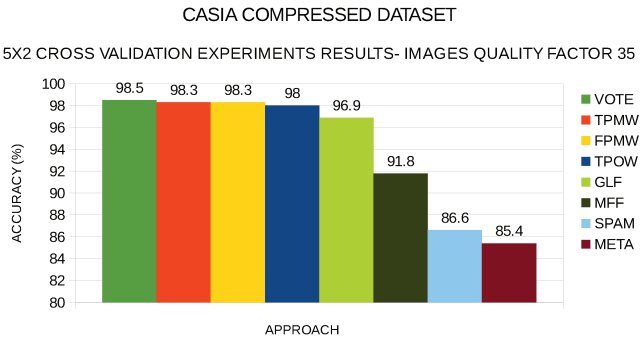


Figure 5.7: Mean classification accuracy after a 5×2 cross-validation comparing the proposed methods and the ones from literature. The database used is the CASIA compressed dataset with image quality factor 35.

significant when compared to all state of the art (GLF, MFF, and SPAM) and also to the fusion-based META. For compression setting 55, there is also statistical difference in the experiments (p-value 1.77×10^{-11}) but the best proposed method is not statistically better than the state of the art GLF (the best state-of-the-art method in this scenario), although it is true to the proposed META and also the methods SPAM and MFF. Finally, for compression setting 35, there is statistical significance (p-value 6.09×10^{-12}) in the experiments, and the best proposed method (VOTE) is statistically significant when compared to GLF, MFF and SPAM and the proposed META.

Figure 5.8 shows a case whereby the proposed techniques are outperformed by the state of the art. In this case, the same 5×2 cross validation scenario is used, but using the CASIA_UNCOMP database. Our approach works better in compressed images because it doesn't matter if the image is blurred or not, the compression already disturbs the image. So, the streaking is more evident using the multiple perturbation (filtering) in compressed images, as the image quality metrics behavior is more clear in blurred and compressed images. For uncompressed images that didn't suffer any kind of disturbance in their creation, it is better to only analyze the neighborhood, although the proposed approach also yielded good results. This is not a serious problem because, in a real-world scenario, uncompressed images are less common than compressed ones. Note, however, that the Tukey-HSD pairwise tests indicates that the best method in this scenario (the state of the art MFF) is not statistically significant when compared to our best proposed

method (VOTE) although it is true when compared to the proposed TPOW and the proposed META (the p-value of Friedmann statistical tests is 2.7 $\times 10^{-10}$).

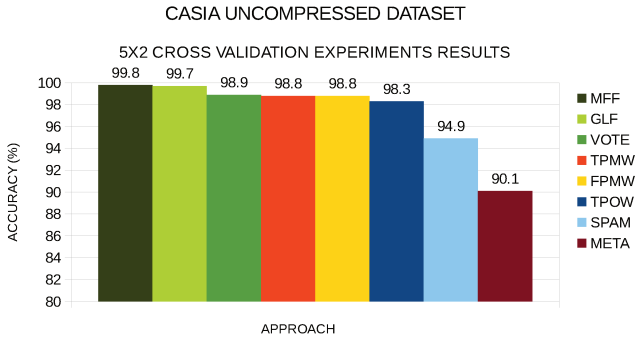


Figure 5.8: Mean classification accuracy after a 5 × 2 cross-validation comparing the proposed methods and the ones from literature. The database used contains 2,773 CASIA uncompressed images.

One interesting point to note in these experiments is the fact that, in all compressed and uncompressed scenarios, there is no trend favoring any of the proposed methods VOTE, TPMW, TPOW, and FPMW preventing us to choose and recommend one of them in all cases. However, this will change in the next section when considering a cross-dataset evaluation.

5.4.4 Comparison with the State-of-the-Art on a Cross Dataset Scenario

In the second round of experiments, we want to validate the proposed techniques in a real-world scenario. For that, we use only one round of training and testing of the classifier. The training data are the same used in the cross-dataset scenario (compressed and uncompressed) from CASIA_COMP and CASIA_UNCOMP and the testing set comes from different (compressed and uncompressed) benchmarks. The compressed unknown data come from the COMPLEX benchmark and the uncompressed come from UCID, as we previously discussed. Table 5.4 shows the average results for the cross-dataset scenario results when using compressed images and uncompressed images.

Metrics	*Approach*							
	TPOW	***TPMW***	***FPMW***	***META***	***VOTE***	*SPAM [100]*	*MFF [102]*	*GLF [104,105]*
Mean Accuracy	79.10%	86.60%	87.10%	73.40%	85.70%	81.70%	74.30%	80.90%
Sensitivity	73.90%	81.70%	81.10%	52.00%	91.00%	67.50%	95.20%	93.90%
Specificity	84.30%	91.60%	93.10%	95.00%	80.50%	96.00%	53.30%	67.80%
Precision	80.00%	89.80%	91.20%	84.00%	88.90%	90.60%	74.60%	78.90%
Significant?	✗	✗	-	✓	✗	✓	✓	✓

Legend:
xx.xx = Best result for the line statistic in a column method
xx.xx = Worst result for the line statistic in a column method

Table 5.4: Cross-dataset experiment average results. Techniques in bold are variations of the proposed method discussed in Sec. 5.2.

According to Table 5.4, the best technique considering the cross-dataset scenario is the proposed FPMW, with an 87.1% mean accuracy. This highlights the use of multiple and multi-scale perturbations. We find that this better captures the streaking artifacts already present in median filtering images. Although this technique does not show the best sensitivity and specificity, it showed the best accuracy because it showed specificity and sensitivity higher than 80%. Also, it showed the best precision in this experiment (91.2%).

The proposed META uses the three distances to the classifier hyperplanes (TPOW, TPMW, and FPMW) as a three dimensional vector for classification. It achieves the second best mean specificity (95%) but it has the worst accuracy in these experiments. This shows that META fusion as a combination of the three proposed methods is not necessary or worth, in general.

The proposed method TPOW yields the second worst mean sensitivity of the experiments (73.9%). Hence, the use of just one single filter scale (median filter mask size) cannot highlight the streaking artifacts and more scales are recommended. The proposed TPMW on the other hand, shows the second best mean accuracy results of the experiment (86.6%), showing that the number of perturbations influences the classification result.

The proposed VOTE method combining the three best proposed methods using majority voting correctly classified 85.7% of the test images. Although not statistically significant when compared to the best single approach here (FPMW, no fusion) in all experiments in the cross-validation scenario, VOTE yielded the best accuracy considering the cross-validation protocol, but it is not the best in this cross-dataset setup. This is explained because the combined methods used by VOTE (FPMW, TPMW, and TPOW) performed well enough individually to boost VOTE in the cross-validation scenario in compressed or in uncompressed scenario. However, the same does not happen in the more strict cross-dataset evaluation protocol, as the failure happens because one of the classifiers used in the fusion (TPOW) had worse classification in compressed and uncompressed images datasets. As we are using both compressed and uncompressed images in this experiment, the error multiplied and the fusion accuracy decreased. This highlights the high complexity of the compressed testing dataset, whereby images were taken by different cameras and smartphones, with different resolutions, noise and light conditions, illustrating what is the closest to a real-world situation.

Regarding the state of the art, the best mean sensitivity was from MFF (95.2%). This means that this approach is the best to detect forgeries, although its low mean specificity (53.3%) render this technique several problems in the forensic scenario as it would blame several innocents (false detections), which is unacceptable in a criminal scenario. The best mean specificity is from SPAM: 96%. This means that this last technique is the best to avoid blaming innocents, although its low mean sensitivity (67.5%) misses about one third of the forgeries.

In summary, multiple perturbations and multiple windows (FPMW) outperform the state-of-the-art methods in this complex cross-dataset setup. Also, as FPMW performs well in nearly all tested scenarios, we further recommend this technique for better median filtering detection.

5.5 Final Considerations and Further Developments

In this chapter, we present novel median filtering detection approaches based on multi-perturbations using one or multiple scales of median filtering. Our technique is different from others because we describe pristine and blurred images by means of image quality metrics, calculated after multiple filterings with different window sizes (filtering intensities), building a discriminative feature space for later decision making. We showed our methods' reliability considering cross-validation and cross-dataset scenarios, with compressed and uncompressed images.

The obtained results here further emphasise the importance of taking into account multiple perturbations in our proposed techniques as they better highlight the median filtering artifacts, namely the *streaking artifacts*. The analogy with text compression is clear: when a series of text compressions are done, if the file is already compressed, chances are that the file size will increase due to the redundancy already coded in previous compressions. Therefore, to find if a suspect text is compressed, one needs just to compress it again and compare the file sizes. In median filtering images, when a series of filterings are performed, if the image is already filtered, there are chances that the streaking artifacts are more propagated and highlighted than in pristine (non-filtered) images. To find if a suspect image is blurred, we can successively blur it and compare the results to the original image using image quality metrics and analyze the behavior of such measures.

Complementing, the use of multiscale filters are also important because, once the original mask used for filtering a suspect image is unknown, the multi-scale filters can be used as some of them can propagate the streaking in a more efficient way than others. A study of the most important features corroborate our findings, showing the features from different scales and perturbations are activated during training.

Finally, observe that the fusion of the three proposed approaches is not necessary, in general, because it works only when all the proposed techniques already perform reasonably well and when they are complementary in some way. Therefore, the take-home method here is the FPMW, first, because the best approaches in the cross-validation experiments in compressed and uncompressed images (VOTE and MFF respectively) are not statistically significantly better when compared to FPMW. Also, the proposed FPMW is the best in the cross-dataset scenario, where there is only one chance to detect forgeries (there is only one test which has different acquisition conditions than the training images). We find that this situation is better than the cross-validation, because in a real-world setup, the data acquired often are in this condition. Finally, FPMW is faster than applying three approaches and combining them (as VOTE does).

For future work, some interesting research branches span out. For instance, one could focus on studying more image quality metrics to be incorporated in the description phase; including more median filtering variations in the training sets; and studying the application of the proposed technique on tiny image patches. Finally, one research branch worth pursuing would be validating the proposed method and possible enhancements under median filtering anti-forensic operations such as the one proposed by Fontani and Barni [168].

Chapter 6

Multi-Analysis Classifier Fusion for Copy-Move Detection

In this chapter, we propose classifier fusion approaches using our multi-analysis solutions to take advantage of the natural complementarity present in several copy-move detectors in the literature. For that we combine such detectors through *multi-analysis* Behavior Knowledge Space Representations. These approaches analyze the input data in multiple scales to generate more samples, more resilient to noise and rescaling, common operations in image forgery. We also employ a multi-directional post-processing approach to the neighborhood of a pixel in order to gather more evidence regarding its true class, instead of the common individual analysis performed by most approaches in the literature. The multi-analysis methods herein are performed in the classification step because the proposed approach is applied on existing individual classifiers published in the scientific literature, acting to define the final classification based on the fusion of outputs from such classifiers.

6.1 Motivation

In the context of fauxtography and digital misleading made possible through image manipulation, the scientific community has been seriously focused on fighting misinformation and detecting these activities in the past few years. One of the most common forgeries consists of selecting, copying and pasting regions from and to an image, multiplying or hiding objects or parts of interest, a process referred to as copy-move tampering or cloning.

Basically, commonly known copy-move detection approaches are divided into two branches according to Christlein *et al.* [1]. The first one uses image patches containing raw or transformed pixels and, by lexicographical sorting and thresholding, similar patches are found in the image. The second set of approaches uses similarity of points of interest, such as those yielded by the Scale-Invariant Feature Transform (SIFT) [169] and also the Speeded-Up Robust Features (SURF) [170] to find copied and pasted regions. By using just image patches, however, rotated and resized regions are difficult to detect. In turn, while points of interest-based approaches can solve this problem as they are invariant to uniform scaling and orientation, they are only partially invariant to affine distortion and illumination changes [169]. As a viable and more interesting alternative to solve the

problem, the combination of these approaches seems promising, as the fusion of different approaches can explore the best of both worlds.

Several methodologies were proposed in the literature in this regard, such as the Majority Voting, Threshold Voting and the Bayesian Fusion [133]. Notwithstanding, these classical approaches for classifier fusion in the copy-move forgery detection scenario do not show groundbreaking effectiveness, as they have strong simplification assumptions on the data that, oftentimes, cannot capture two important properties of the problem: (i) a pixel classification is not solely dependent on the actual pixel, it depends also on the pixel's neighborhood due to the very nature of the forgery process, which involves combining different pixels in a given neighborhood and (ii) it is necessary to know, for each method that is good for classification, the cases in which the others methods are not, which can decrease the detection accuracy after a voting is performed, for example. Therefore, it is necessary to know the probability of a fake pixel that has not been detected by most of the fused approaches. The problem of fusion here must be designed as a conditional probability estimation problem instead.

A very promising and under explored way of modelling conditional probabilities for classifier fusion is through the Behaviour Knowledge Space representation (BKS) of the data [134]. This fusion approach encodes the combination of outputs of the combined classifiers as a posteriori probabilities in a probability table on a training step. In the testing step, the combination of outputs from the approaches applied in a suspected image is queried in the table, which returns the probability of that combination being the combination of a forged pixel. The final probability map is thresholded to decide to which class each combination of outputs belongs. The drawbacks of this approach for image processing applications in general are: (i) it is almost impossible to have all the 2^K classifier output combinations in the training data (considering a binary classification problem) to fill the table properly (in a digital image forensic scenario, it is very difficult to gather enough training examples, as each image is forged in a different way and detecting all details of image forgeries by these classifiers is a daunting task); and (ii) the probabilities in this table do not take into account the intrinsic dependencies present in the data, which can yield imprecise classifications. Thus far, these aspects have limited the use of conditional probabilities for modeling output combinations of classifiers in the copy-move forensic scenario.

In this work, we deal with the limitations of fusing approaches by designing a robust and efficient Behaviour knowledge Space (BKS) [134] representation more appropriate for copy-move detection, modeling the problem as a conditional probability estimation problem instead. The extensions and contributions are threefold.

First, we deal with the problem of missing probability estimations caused by lack of training data, using generative models to better determine missing entries and remove noise from the existing probabilities in the representation space adopted. This first contribution is very important when combining forgery detection algorithms as oftentimes they are complementary but it is very hard to find enough training examples to cover all cross-effects of their combinations. Second, we incorporate expert knowledge to the adopted BKS representation in order to be more robust to some common operations in image tampering that can lead to confusion in the classification of individual classifiers, such as resizing

and noise addition. For that, we propose a Multiscale Behavior Knowledge representation, which takes into account different scales of training data. Finally, we deal with the problem of individual pixel classification, present in most copy-move detection approaches, which can decrease the classification accuracy, as the neighborhood plays an important role in the fate of a pixel's classification. For that we incorporate a post-processing step to the detection BKS-based technique, which classifies a pixel based on the outcomes of its neighborhood.

We show by experiments that these three problems when properly dealt with yield a better classifier (i.e., forgery detector), which takes into account the benefits of each individual classifier aggregated and is statistically better than its counterparts in the literature. These extensions were properly thought of and custom-taylored to the problem of forgery detection and, we believe, represent a major leap toward the design of more effective forensic methods that can take advantage of complementary features to solve a hard problem.

We organized the remainder of this chapter into four sections. In section 6.2, we introduce the proposed schemes to perform the fusion of classifiers based on BKS modelling. In section 6.3, we set forth the experimental setup used to validate the proposed methods while, in section 6.4, we present the experimental results. Finally, in section 6.5, we conclude the chapter and discuss some possible future work.

6.2 Multi-Scale and Multi-Directional Behavior Knowledge Space Classification for Forgery Detection

In order to better understand the BKS fusion applied in copy-move detection and our contributions in this chapter, we discuss how the single BKS-based classification workflow works for copy-move detection. Firstly, given a training set of images, we apply K copy-move detectors and use their binary detection maps to generate the Behavior Knowledge Space representation. This is done by analyzing, pixel by pixel, the combination of K outputs for that pixel and the class of that pixel in the ground-truth used in a training set. In the testing set, the combination of the K outputs is queried in the table and a decision threshold in the conditional probability of that combination is used to classify the pixel in the test image.

In this chapter we propose a series of BKS-based approaches aimed at fighting the drawbacks presented for BKS fusion, extending it to consider the multi-scale and multi-directionality nature of the data in the copy-move forgery detection problem. The multi-scale approaches applies Gaussian pyramidal decomposition of the training images, filling the remaining conditional probabilities in the BKS representation that could not be found using only the original scale of training images. They are also used to give better examples to the BKS, as the pyramidal decomposition eliminates noise that can be mislabeled as copy-move pixels.

The multidirectionality approaches, on the other hand, aim at improving the classification results, by taking into account the dependency nature of the data. We also propose the use of generative models that can act alone or allied with the multiscale approaches to

better estimate the probabilities of forgeries when combining different methods. Figure 6.1 depicts the pipeline of our proposed BKS-based approaches aimed at copy-move detection. Algorithm 2 shows the main steps of the proposed approach.

Algorithm 2: Proposed Method.

```
function CMD_BKS (I, BKS, K, neigh_size, neigh_type)
Input  : I: Input image to detect copy move forgery.
         BKS: Multiscale BKS Table.
         K: Number of underlying forgery detectors to combine.
         neigh_size: Size of neighborhood to analyze.
         neigh_type: Type of neighborhood analysis.
Output : B: Binary image with the manipulation detection
//Apply each detection approach individually per pixel
for each pixel i in I do
    for k=1 to K do
        output[k]=detect_forgery(k,i)
    end
    //Look for outputs probability in the BKS table built as discussed in sections
     6.2.1 and 6.2.2
    prob=search_table(output, BKS);
    prob_map(i)=prob;
end
//Perform the neighborhood analysis using approaches discussed in section 6.2.3
for each pixel j in prob_map do
    class=neigh_analysis(prob_map(j), neigh_size, neigh_type);
    B(j)=class;
end
return B;
```

The multi-analysis procedure performed herein is done in the classification step of input image analysis, as the proposed approach is applied when combining existing individual classifiers proposed before, acting to define the final classification based on such classifiers. The scenarios used to solve this problem are the multi-direction and multi-scale.

We now turn our attention to discussing the main contributions for BKS-based fusion detection of copy-move forgeries.

6.2.1 Multiscale Behavior Knowledge Space

In this chapter, we propose a novel data fusion approach by using multiscale analysis of the data to build a more robust BKS representation, invariant to operations such as noise and resizing. For that, we use the Pyramidal Decomposition [121] from input images. We use the pyramidal decomposition in two ways in our proposed BKS classification:

1. Multiscale BKS: we use s image scales of training images to generate only one BKS representation table used for testing. This is performed to complete the BKS table with more samples robust to common operations used with copy-move tampering, such as noise addition and resizing.

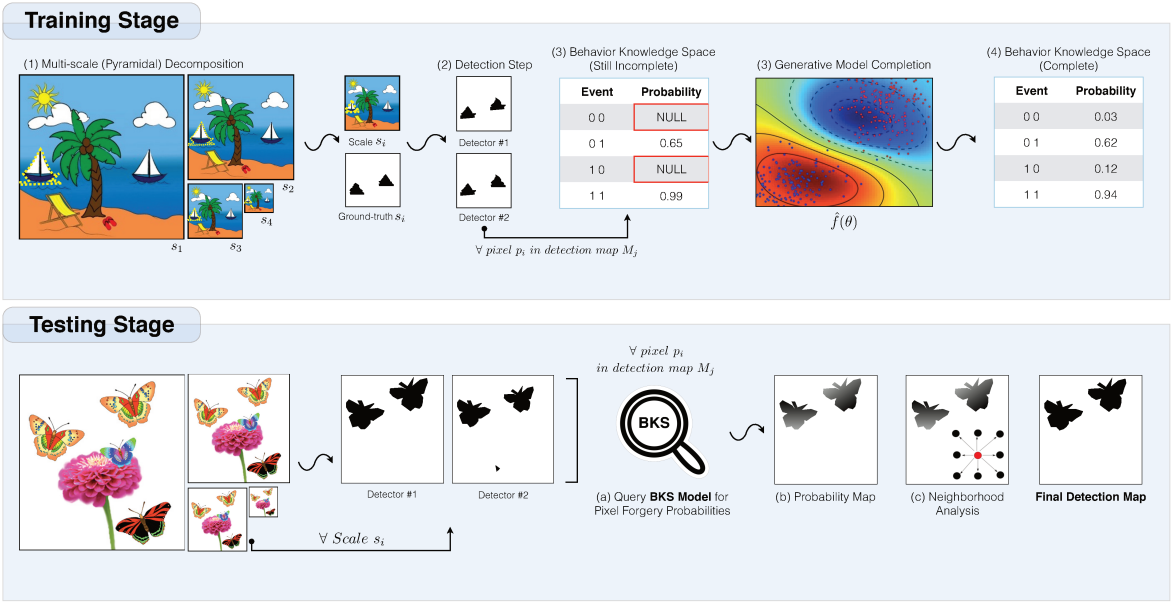


Figure 6.1: Workflow of the proposed Behavior Knowledge Space applied to copy-move forgery detection. We start by building a Multiscale representation of the data by applying the Gaussian pyramidal decomposition on input images (contribution labeled as (1) in the training stage). This makes the combined classifier more robust to some operations applied in copy-move forgery, such as resizing and noise addition. This process results in an incomplete representation, as all the possible combinations of binary outputs from K combined detectors often cannot be found in the training scenario. We solve this problem by applying a generative model completion such as regression to better fit the conditional probability data, filling missing probabilities and also removing possible noise and outliers from the BKS representation (contribution labeled as (3) in the training stage). Finally, in the test stage, for each pixel in the image, we calculate, querying the BKS representation, its probability of being a copy-move forgery given the K detection maps. This generates a probability map which is further processed by multidirectional neighborhood analysis (contribution labeled as (c) in the testing stage) to classify a pixel based on its neighborhood information, which is crucial for the problem we deal with in this chapter. The pyramidal decomposition happens in the training/testing depending on the proposed technique as we detail in the text.

2. Multiscale BKS Voting: s scales are used to generate t $(s = t)$ BKS representation tables in the training stage. In the testing phase, the s scales of the test image are classified by the corresponding t representation of that scale. The final result is the voting of s multiscale final binary maps.

We believe that the proposed Multiscale BKS approaches are effective because: (i) they improve the robustness of the proposed detector against different scale forgeries; (ii) the augmented representation (with more training data) leads to a more precise prediction of the probability of a given pixel being fake; and (iii) outliers in the image that could be interpreted as copied regions are reduced due to the low-pass filters used in the pyramidal decomposition.

6.2.2 Generative Models for Behavior Knowledge Space Completion

Even using the multiscale approach presented before, some conditional probabilities cannot be calculated from the training set, as some output combinations of classifiers may never be present in such data. This can be a problem because, during testing, an unknown entry could be considered as a false negative. In order to overcome this issue, we propose a completion procedure based on regression, as it is widely used to predict unknown values from existing ones.

We propose to train Random Forests and Support Vector Regressions with the entries in the eventually incomplete Behavior-Knowledge Space representation table. Our hypotheses is that the regression should eliminate some noises present in the training data and, thus, generalize better for the testing environment. We detail each of these approaches in the following.

Random Forests (RFs)

Random Forests is a method composed by a collection of classification or regression trees, each constructed upon a random resampling of the original training set. In the notation provided by [171], a training set is denoted by $\mathcal{L} = \{(\boldsymbol{x_i}, y_i)\,,\ i = 1, 2, ..., N\}$ where N is the number of samples, $\boldsymbol{x_i}$ is the vector of attributes and $y_i \in \{1, 2, ..., C\}$ is the n-th example in the training set.

Before describing the Random Forest procedure, let's first consider the concept of bootstrap aggregation or tree bagging applied to tree learners. Given a training set $\mathcal{L}$, bagging repeatedly selects a random sample with replacement of the training set and fits trees to such samples. This process is repeated B times. In each iteration b, we sample with replacement, N examples from $\mathcal{L}$, creating $\mathcal{L}_b$, and train a regression tree f_b on $\mathcal{L}_b$. After training, we can predict the outcome of unseen examples $\boldsymbol{x}_t$ by averaging the predictions from all the individual regression trees on $\boldsymbol{x}_t$

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(\boldsymbol{x}_t) \tag{6.1}$$

The bootstrapping decreases the variance without impacting the bias of the model thus leading to a better model performance. As the parameter B is free, we can set its value through cross-validation, or by observing the mean prediction error on each training sample $\boldsymbol{x}_i$, using only the trees that do not contain $\boldsymbol{x}_i$ in their bootstrap sample, a process referred to as out-of-bag error.

The difference of the process described above and actual random forests is that RFs use a modified tree learning algorithm that selects a random subset of the features for each candidate split (tree) in the learning process, an approach oftentimes referred to as "feature bagging". This sampling is applied mostly to reduce correlation among different trees and, therefore, better explore the feature space. More information about Random Forests and their properties can be found in [171]

In our scenario, Random Forests will be used to fill all the BKS entries. We use as

$\boldsymbol{x}$ an $n-dimensional$ vector containing the binary output of each copy-move detection approach that are present in the BKS tables and, as y, we use the probabilities also in the BKS entries. For training, we use only $\boldsymbol{x}$ and y (binary outputs combinations and probabilities, respectively) that are already present in the BKS entries (binary outputs without calculated probabilities are discarded for training). Then, after trained, the random forests will predict the missing probabilities in the BKS table for each table entry (outcomes of the detectors). For instance, suppose a BKS in Fig. 6.1. In that case, the table entry $\boldsymbol{x} = \{0, 0\}$ is missing. After the RF regression, it is estimated in $P(\boldsymbol{x}) = 0.03$.

Support Vector Regression (SVR)

Consider again a training data set $\mathcal{L} = \{(\boldsymbol{x_i}, y_i) \mid i = 1, 2, \ldots, N\}$ where $\boldsymbol{x_i}$ denotes the input vectors and associated targets y_i and N the number of samples. Training an original SVR means solving the regression problem as a convex optimization [172]:

$$\text{minimize} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 \text{ , subject to} \begin{cases} y_i - (\boldsymbol{w}.\boldsymbol{x} + b) \leq \varepsilon \\ (\boldsymbol{w}.\boldsymbol{x} + b) - y_i \leq \varepsilon \end{cases} \tag{6.2}$$

The convex optimization problem is *feasible* if there exists a function that approximates all pairs $(\boldsymbol{x_i}, y_i)$ with ε precision. When solving the complex optimization for finding $\boldsymbol{w}$, there are points that often violate the restrictions of the problem and cannot guarantee the *feasibility*. Then, we adopt a loss function that introduces non-negative slack variables $\xi_i, {\xi_i}^*$ to the problem formulation to cope with infeasible constraints of the optimization problem in Equation 6.2.

In the nonlinear case, we use a function to map $\Psi : \chi \rightarrow \Im$ onto a feature space $\Im$ [172] and apply the SVM Regression algorithm on the transformed data. As [172], the Supprt Vector algorithm only depends on dot products between patterns $\boldsymbol{x}_i$. So, $k(\boldsymbol{x}, \boldsymbol{x}') = \Psi(\boldsymbol{x}, \boldsymbol{x}')$ rather than Ψ explicitly. In this case, we operate in this transformed space. More details about SVR can be found in [172].

In our scenario, SVR will be used the same way as Random Forests were used to fill all the BKS entries. We use as $\boldsymbol{x}$ an $n-dimensional$ vector containing the binary output of each copy-move detection approach that are present in the BKS tables and, as y, we use the probabilities also in the BKS entries. For training, we use only $\boldsymbol{x}$ and y (binary outputs combinations and probabilities, respectively) that are already present in the BKS entries (binary outputs without calculated probabilities are discarded for training). Then, after trained, the random forests will predict the missing probabilities in the BKS table for each table entry (outcomes of the detectors). For instance, suppose a BKS in Fig. 6.1. In that case, the table entry $\boldsymbol{x} = \{0, 0\}$ is missing. After the RF regression, it is estimated in $P(\boldsymbol{x}) = 0.05$.

6.2.3 Multidirectional Neighborhood Analysis for BKS Classification

In the first formulation of the BKS fusion classification scheme, the testing phase works as follows: first, the table is queried for the probability given the output combination of

individual classifiers for a testing pixel. This method will produce a probability of a pixel being forged, given a combination of the individual classifiers, creating a final probability map for the image, which is then compared pixel-wise to a threshold to classify its pixels as forged or not. The probability and threshold are always the same for a particular combination of classifiers' output and this can be a problem, as the neighborhood also has influence on a pixel's classification. To solve this issue, we propose novel neighborhood-based classification schemes considering the Behavior Knowledge Space-based classification fusion. These new approaches are based on multidirectional analysis of the data, classifying a pixel based on its neighborhood. We discuss each of them in the following subsections.

Neighborhood Agreement (NA)

The Neighborhood Agreement method uses the probability computed with the BKS method, but taking into account the information present in the pixel neighborhood. The rationale is that a forged region should have a minimum size and that an observation of the detectors' outputs in isolated pixels should be conditioned with the observations of nearby pixels as well.

In this proposed approach, the probability map used for further classification is generated after a convolution operation on the original probability map, built after each image pixel evaluation in our extended BKS model. The kernel we select for this approach is the mean filter. The new probability of a pixel is the mean probability of its neighbors. A base threshold of 0.5 can be used to perform the final detection map (conditional probabilities with higher values can pinpoint a forgery).

Local Variable Threshold (LVT)

The main idea behind this method is using the neighborhood of a pixel to dynamically adapt the decision threshold to classify it. The rationale is that if the neighbors of a pixel p are likely forged, then p is also probably a forged pixel. In other words, the more pixels are forgeries in the neighboord of p, the more likely p is to be forged, dynamically adapting the decision threshold.

To create the dynamical decision process, we create a local variable threshold that moves into a fixed interval around a base threshold, hereinafter referred to as the Max Displacement (MD). If we define MD to be 0.2, for instance, and the base threshold T to be 0.5, it means that we expect that the threshold can take values in the interval [0.3, 0.7]. The final Local Variable Threshold for a given neighborhood is calculated as

$$LVT = T - 2 \times (MC - T) \times MD, \tag{6.3}$$

where MC is the mean classification output of pixels in a dubious pixel's neighborhood.

As an example, suppose a 3×3 neighborhood with five pixels classified as copy-move pixels and our task is to classify the center pixel of this region. For this case, the threshold for that pixel is $VT = 0.5 - 2 \times (\frac{5}{8} - 0.5) \times 0.2 = 0.45$. If the BKS table gives the probability of being forged as, for instance, 0.48 but the pixel is a copy-moved pixel, then

a false negative would be avoided due to the less strict threshold 0.45 in this proposed approach.

6.2.4 Complexity Analysis

The complexity of the proposed method depends mostly on the complexity of three elements: complexity of underlying classifiers used in the fusion, complexity to access the probability of the combined responses in the BKS representational space and the complexity of the neighborhood (multi-directional) analysis. The complexity of the underlying classifiers used in the fusion is clearly dominated by the complexity of the most complex method. Considering k methods to be combined and assuming that the most complex one is $O(N^2)$, the complexity of the combined classifier is $O(kN^2) = O(N^2)$.

The complexity of accessing the BKS table can be done in $O(1)$ if we implement the representation space with a hash. The complexity of the neighborhood analysis is done in a fixed-neighborhood size times the number of pixels in the image. Hence the neighborhood analysis complexity is $O(N)$. Summing up, the final complexity of proposed method is $O(N^2 + 1 + N) = O(N^2)$. In other words, the complexity of the proposed fusion method depends on the complexity of the underlying methods used in the fusion scheme.

6.2.5 Known Limitations

As the proposed method works with fusion of classifiers, its weakness happens when there is no complementarity of the underlying classifiers for a given image. For example, when combining block-based methods and interest points-based methods, if the first ones fail in detecting the forgery and the image is too homogeneous to have enough interest points detected, the fusion can fail.

Finally, as the method consists of running and combining k detection methods, search the output combination in the multiscale BKS representational space and the neighborhood should be analyzed for the final classification, the proposed method is slightly slower when compared to other existing methods, mainly the ones not using any fusion scheme. However, the obtained effectiveness boost is significant and worth the slight increase in the computational time, as we show in the experiments.

6.3 Experimental Setup

With all the proposed solutions in place, we start turning our attention to the methodology used to validate them against counterparts in the literature. In this section, we show the datasets, the validation setup, the statistics used for comparison, the methods considered and the variations of the proposed approaches used in the experiments.

6.3.1 Datasets

We have used two datasets for evaluating and comparing the proposed techniques with the ones from the literature. The first dataset, proposed and used in [4], comprises 108

examples of copy-move forgeries. Each image is stored in uncompressed PNG format and in compressed JPEG format, totalling 216 images. The images have different dimensions, varying from 845×634 pixels (the smallest) to $1,296 \times 972$ pixels (the largest). We refer to this dataset as *Copy-Move Hard (CPH)* as it comprises forgeries created through mixed operations such as resizing, rotation, scaling, compression, light matching, among others. We separate this dataset in two subsets: the one comprising the compressed version of the images (*CPHCOMPRESSED*) with 108 images and the uncompressed version of the images (*CPHALL*) also with 108 images. Each subset may be further break down as:

- 23 images, in which the cloned area was just copied and moved (simple case);
- 25 images with a rotation of the duplicated area (orientations in the range of -90 and 180 degrees);
- 25 images with cloned area resizing (scaling factors between 80% and 154%);
- 35 images involving rotation and resizing altogether.

The second dataset comprises images from Christlein *et al.* [1] who compared several copy-move detection methods. We refer to this dataset as *Copy-Move Erlangen-Nuremberg (CMEN)*. In total, we considered 212 images stored in PNG format with dimensions varying from 800×533 pixels (the smallest) to $3,872 \times 2,592$ pixels (the largest). $CMEN$ datasets comprise:

- 48 images where the cloned area was only copied and then moved (simple case);
- 78 images with a rotation of the duplicated area (orientations of 2, 4, 6, 8, 10, 20, 60 and 180 degrees);
- 86 images with a resizing of the cloned area (scaling factors of 50%, 80%, 91%, 93%, 95%, 97%, 99%, 101%, 103%, 105%, 107%, 109%, 120%, 200%).

We have chosen exactly these two dataset configurations because they are the same used in the validation of a recent work [4] and are freely available by the authors at the project's website[1]. Moreover, we use a slightly different validation from [4] when performing experiments on these two datasets, as there is a training step in our proposed approaches and some state-of-the-art fusion techniques. In the experiments reported in this chapter, we randomly choose images from these datasets to be used in training and test steps in a validation protocol explained in section 6.3.2.

6.3.2 Setup

We adopt a 5×2 cross-validation protocol in the experiments, as the proposed approaches need a training stage. Five replications of the 2-fold cross-validation protocol are performed. In each one, a set S is randomly divided into S_1 and S_2 and a classifier is trained on S_1 and tested on S_2. Thereafter, training/testing sets are switched and the process repeated. There are $5 \times 2 = 10$ different executions in the end of the process. This is considered an optimal benchmarking protocol for learning algorithms [138].

[1] http://dx.doi.org/10.6084/m9.figshare.978736

6.3.3 Metrics and Statistics

In the experiments, all metrics are calculated in a pixel-level fashion to evaluate the effectiveness of the detection maps yielded by the methods applied on the benchmarks. This approach has been chosen mainly because it is the preferred approach used in the IEEE Image Forensics Challenge (IFC) presented in [173]. It is worth mentioning that recent trends in the information forensics community have pushed for pixel-wise classification and localization instead of only image-wise binary metrics. For evaluating all the proposed methods against the state of the art, we have chosen the following metrics, also used in the IFC [173]:

- **True Positive Rate (TPR):** also known as recall, it indicates the percentage of correctly classified copy-move/cloned (or positive) regions $TPR = \frac{|TP|}{|R_{clone}|}$, where $|TP|$ (True Positive) represents the number of pixels correctly classified as cloned in the detection map, and $|R_{clone}|$ represents the number of real cloned pixels in the reference map.

- **False Positive Rate (FPR):** indicates the percentage of incorrectly located cloned regions $FPR = \frac{|FP|}{|R_{normal}|}$, where $|FP|$ (False Positive) represents the number of pixels wrongly classified as cloned in the detection map, and $|R_{normal}|$ represents the number of pixels, in the reference map, that do not belong to the cloned regions.

- **Accuracy (ACC):** gives the quality of detection based on TPR and TNR (True Negative Rate), which indicates the percentage of correctly located non-cloned regions $ACC = \frac{TPR+(1-FPR)}{2}$, where $(1 - FPR)$ represents the TNR.

- **Precision:** is the fraction of events in which the classifier *correctly* classified forged pixels out of all instances classified as being copy-move pixels $Precision = \frac{TPR}{TPR+FPR}$.

- **F-Measure:** is a measure that can be interpreted as the harmonic mean of precision and recall (also known as True Positive Rate, or TPR as discussed before). It reaches its best value at 1 and worst score at 0:

$$f = 2 \cdot \frac{Precision \times TPR}{Precision + TPR}. \tag{6.4}$$

We also report Standard Deviations (STD) for TPR, FPR and ACC in all experiments to give an idea of how the results vary across the different cross-validation rounds.

A series of statistical tests are also performed to check if the reported results are significantly different. First, we confirm if all techniques are statistically different (also known as pre-test). If so, we check the techniques pairwise to define which ones are statistically different when compared to each other (also known as post-test). Each of these steps usually involves a statistical test and a confidence level for the test. We consider a confidence level of 95% for each test. As pre-test, we considered the Friedmann test [174], a non-parametric test used to determine if subjects change significantly across occasions and conditions. For pairwise comparison, also known as multi-comparison approach, we use the Wilcoxon rank-sum paired test [175] for two reasons: (i) we do not assume that the

difference between the two variables being compared is interval and normally distributed; and (ii) the sample sizes are small (10 f-measures per method representing each result of the 5×2 cross validation procedure). As there are multiple pairwise comparisons, a p-adjustment must be done during the tests. We chose the p-value adjustment using the method by Benjamini and Yekutieli [176] as it controls the false discovery rate in the test, being more powerful than other p-value adjustments methods, such as Bonferroni [177] or Holm [178].

6.3.4 Implementation Aspects of the Proposed Methods

In this section, we discuss a series of variations of the proposed approaches based on multi-scale and multi-directional evaluation of the considered BKS representations:

1. Initial BKS proposed methods considering the Support Vector Regression (BKS-SVR) and Random Forest (BKS-RF) regression taking place for finding missing probabilities in the training data.

2. Second set of proposed methods using the multidirectional neighborhood analysis techniques on top of the previous two approaches by Neighbor Agreement (BKS-SVR-NA and BKS-RF-NA) and by Local Variable Threshold (BKS-SVR-LVT and BKS-RF-LVT).

3. Third set of proposed methods. In this set, we used Otsu's threshold [79] on the probability map representation (BKS-SVR-OTSU and BKS-RF-OTSU) to generate the final classifications.

Therefore, for a single scale of the image, a total of eight variations of the proposed approach are applied. For the Local Variable Threshold approach, we define the Max Displacement parameter (MD) to be 0.2 and base threshold T to be 0.5 in all variations of the proposed approaches where LVT is used.

For multiple scales, we also use a similar configuration as presented in section 6.2.1 (we only report the Local Variable Threshold approach because it yielded the best results). We label these approaches as MULTISCALE VOTING BKS-RF-LVT, MULTISCALE VOTING BKS-SVR-LVT, MULTISCALE BKS-RF-LVT and MULTISCALE BKS-SVR-LVT respectively), totalling 12 proposed approaches.

For finding the best parameters in the used methods, we considered a simple grid-search procedure using 80% of the training from one fold of the 5×2 cross validation for training and the remaining 20% for validation. The experiments results allowed us to specify a 9×9 window in the proposed Local Variable Threshold approach for all kinds of images as well as a 9×9 window for the Neighborhood Agreement in the case of uncompressed images. For the Neighborhood Agreement in compressed images when used with random forest regression, it is used a 5×5 window while the version with SVR uses a 3×3 window. All the parameters, once again, are automatically calculated based on the training data. For Random Forests, we varied the parameter *number of trees in the forest* and the parameter *number of features randomly sampled* and found the best ones as being $(1,000; 2)$ and $(250; 2)$ for compressed and uncompressed images, respectively. For SVR, we varied the

Cost and Gamma parameters and found the best ones as being $(1; 0.125)$ and $(1; 0.5)$, respectively, for compressed and uncompressed images.

All of the proposed methods are based on BKS representation built upon the outcomes of eight individual detectors: four block-based (Popescu and Farid [109], Ryu *et al.* [115], Ryu *et al.* [116] and Bashar *et al.* [118]) and four interest-point based (Amerini *et al.* [130], Shivakumar and Baboo [3] SIFT, Shivakumar and Baboo [3] SURF and Silva et al. [4]). We chose this configuration because of the good classification results of these individual approaches reported in the literature and because we want to take into account the advantages of block-based and interest point based detections in the fusion of classifiers. The source code of the proposed approaches in this chapter can be found at GitHub[2].

6.3.5 Baselines

We compare the proposed techniques to 16 individual state-of-the-art methods (presented in section 2.3). These methods have been chosen based on a previous study conducted by Christlein *et al.* [1] and based on other works as well. All of these copy-move detectors and their labels used in this chapter are presented in Table 6.1.

Method	**Label**
Mahdian and Saic [111]	Blur
Wang *et al.* [120]	Circle
Fridrich *et al.* [108]	DCT
Bashar *et al.* [118]	DWT
Bayram *et al.* [119]	FMT
Wang *et al.* [122]	Hu
Lin *et al.* [123]	Lin
Bashar *et al.* [118]	KPCA
Popescu and Farid [109]	PCA
Shivakumar and Baboo [3]	SIFT
Shivakumar and Baboo [3]	SURF
Amerini *et al.* [130]	Hierarch-SIFT
Kang and Wei [114]	SVD
Ryu *et al.* [115]	Zernike
Ryu *et al.* [116]	Zernike2
Silva *et al.* [4]	Multiscale Voting

Table 6.1: Label associated with each individual state-of-the-art copy-move detector used in the experiments.

We also compared the proposed methods against all fusion approaches presented in section 2.4. Basically, we combined the same eight individual state-of-the-art approaches used in the proposed methods (labeled as DCT, Zernike, Zernike2 and KPCA, Hierarch-SIFT, SIFT, SURF and Multiscale Voting in Table 6.1). For the threshold voting (which we label THRESHOLD VOTING), we used two configurations: one with hard voting (we

[2]https://github.com/anselmoferreira/bks-copy-move-detection

defined six votes as the minimum base threshold to classify a pixel as forged) and other with soft voting (four positive answers from the fused approaches for a pixel classifies it as forged). We considered six votes for hard voting because more votes would require a very high consensus between the combined approaches, missing several detections. We also use the original Behavior Knowledge Table [134] (labeled simply as BKS) and the Bayesian Fusion approach [133] (labeled as BAYESIAN FUSION) in the comparison, with 0.5 used as the base threshold. With these four additional fusion approaches, we compare the proposed methods with a total of 20 state-of-the art copy-move detection techniques, containing individual and fused classifiers.

6.4 Results and Discussion

We now turn our attention to the experiments and results. We present the experiments for all methods considering the selected datasets in quantitative forms, along with the proper statistical analysis of the results. Qualitative results are also shown in this section to compare the detections maps generated by the proposed approach and the counterparts in literature.

6.4.1 CPH and CPHCOMPRESSED Datasets

With the datasets presented in section 6.3.1 and methodology described in section 6.3.2 in mind, we now discuss the experimental results, whereby we validate the proposed approaches comparing them to the state-of-the-art methods presented in section 6.3.5. Table 6.2 shows results considering the measures presented in section 6.3.3 in a 5×2 cross-validation protocol on $CPHCOMPRESSED$ dataset.

Table 6.2 shows results for fusing patch-based and interest-based copy-move approaches in a probabilistic way as BKS does. The original BKS and the methods proposed in this chapter outperform all the baselines compared in this experiment. The Local Variable Threshold was used in the best four approaches, highlighting the importance of studying the neighborhood before deciding to which class a given pixel belongs, as the proposed multi-directional thresholding approach does. This approach yielded a 21% reduction of error if compared to the best state of the art: the common BKS.

The best result is the one which uses the proposed multiscale BKS-based solution (MULTISCALE BKS-RF-LVT) with an f-measure of 84.14%, higher than the ones achieved by original BKS (77.69%) and the best individual approach (SURF), with 76.49%. This shows the benefits of applying the multiscale approach, eliminating noise and updating the BKS representation with samples robust to resizing and noise additions, using generative models for missing probabilities estimation and studying the neighborhood before deciding the class of a given pixel. The fusions by Voting (THRESHOLDING VOTING) and Bayesian (BAYESIAN FUSION) approaches are far from being acceptable in this scenario. Varying the base threshold from hard ($T = 6$) to soft ($T = 4$) voting does not change the classification results of THRESHOLDING VOTING, and the assumption of independence of classifiers (as the Bayesian approach uses) is wrong in this scenario, as the worst results of this approach shows. The proposed BKS-based methods relying on voting (MULTISCALE

Rank	Method	Statistics Calculated on CPHCOMPRESSED Dataset after 5X2 Cross-Validation Experiments							
		F-MEASURE (%)	ACC (%)	ACC (σ)	TPR (%)	TPR (σ)	FPR (%)	FPR (σ)	PRECISION (%)
1	**MULTISCALE BKS-RF-LVT**	**84.14**	**85.48**	**18.21**	**77.93**	**34.52**	**0.06**	**7.88**	**99.92**
2	**MULTISCALE BKS-SVR-LVT**	**83.67**	**85.08**	**17.94**	**77.09**	**34.07**	**0.06**	**7.87**	**99.92**
3	**BKS-SVR-OTSU**	**83.07**	**84.70**	**17.71**	**76.56**	**33.69**	**7.14**	**7.37**	**91.47**
4	**BKS-RF-LVT**	**82.50**	**84.19**	**18.68**	**79.22**	**35.47**	**10.00**	**7.85**	**88.79**
5	**BKS-SVR-LVT**	**81.32**	**83.74**	**18.62**	**78.25**	**35.44**	**0.10**	**7.87**	**99.87**
6	**BKS-RF-NA**	**80.94**	**82.99**	**19.02**	**72.12**	**36.36**	**6.13**	**7.52**	**92.17**
7	**BKS-RF**	**80.59**	**82.77**	**19.07**	**71.58**	**36.53**	**0.06**	**7.33**	**99.92**
8	**BKS-RF-OTSU**	**79.51**	**82.08**	**18.78**	**71.09**	**35.94**	**6.91**	**7.36**	**91.14**
9	**BKS-SVR**	**79.02**	**81.71**	**19.01**	**69.45**	**36.51**	**6.00**	**7.35**	**92.05**
10	**BKS-SVR-NA**	**78.91**	**81.68**	**19.23**	**69.05**	**37.09**	**0.06**	**7.01**	**99.92**
11	*BKS [135]*	77.69	81.57	19.90	64.64	39.46	1.49	2.09	97.75
12	*SURF [3]*	76.49	80.99	20.68	62.22	41.34	0.24	0.55	99.61
13	**MULTISCALE VOTING BKS-RF-LVT**	**71.07**	**77.54**	**20.45**	**55.85**	**41.16**	**0.00**	**1.24**	**100.00**
14	*SIFT [3]*	69.80	76.89	23.11	53.93	46.30	0.15	0.22	99.73
15	**MULTISCALE VOTING BKS-SVR-LVT**	**67.44**	**75.82**	**19.42**	**52.92**	**38.74**	**1.27**	**1.81**	**97.66**
16	*Hierarch-SIFT [131]*	64.83	74.04	21.15	48.29	42.41	0.21	0.30	99.57
17	*THRESHOLD VOTING (T=4)*	54.83	68.97	18.30	37.96	36.64	0.02	0.05	99.94
18	*THRESHOLD VOTING (T=6)*	54.83	68.97	18.30	37.96	36.64	0.02	0.05	99.94
19	*Mutiscale Voting [4]*	45.95	64.73	17.31	30.15	34.79	0.69	1.24	97.29
20	*Zernike [116]*	44.33	64.18	14.85	28.60	29.57	0.23	0.95	99.21
21	*Zernike2 [117]*	39.25	60.28	19.91	25.95	38.34	5.39	7.32	82.80
22	*KPCA [119]*	38.63	61.92	15.44	24.06	30.88	0.22	1.11	99.10
23	*DWT [119]*	37.69	61.55	14.71	23.33	29.34	0.23	1.03	99.03
24	*DCT [109]*	37.57	61.55	14.76	23.21	29.47	0.10	0.35	99.56
25	*FMT [120]*	15.44	54.18	8.44	8.42	16.86	0.06	0.28	99.29
26	*Circle [121]*	14.67	53.90	6.81	7.96	13.77	0.15	0.71	98.12
27	*Lin [124]*	10.55	52.79	6.13	5.61	12.27	0.03	0.12	99.50
28	*Hu [123]*	4.07	50.86	2.09	2.09	4.21	0.36	1.17	85.38
29	*Blur [112]*	1.95	50.31	1.47	0.99	2.95	0.36	1.23	73.52
30	*BAYESIAN FUSION [134]*	1.88	50.52	1.11	1.04	2.23	0.00	0.00	99.88
31	*SVD [115]*	1.54	50.31	1.36	0.78	2.63	0.15	0.47	83.90
32	*PCA [110]*	0.00	49.96	0.19	0.02	0.16	0.09	0.31	18.61

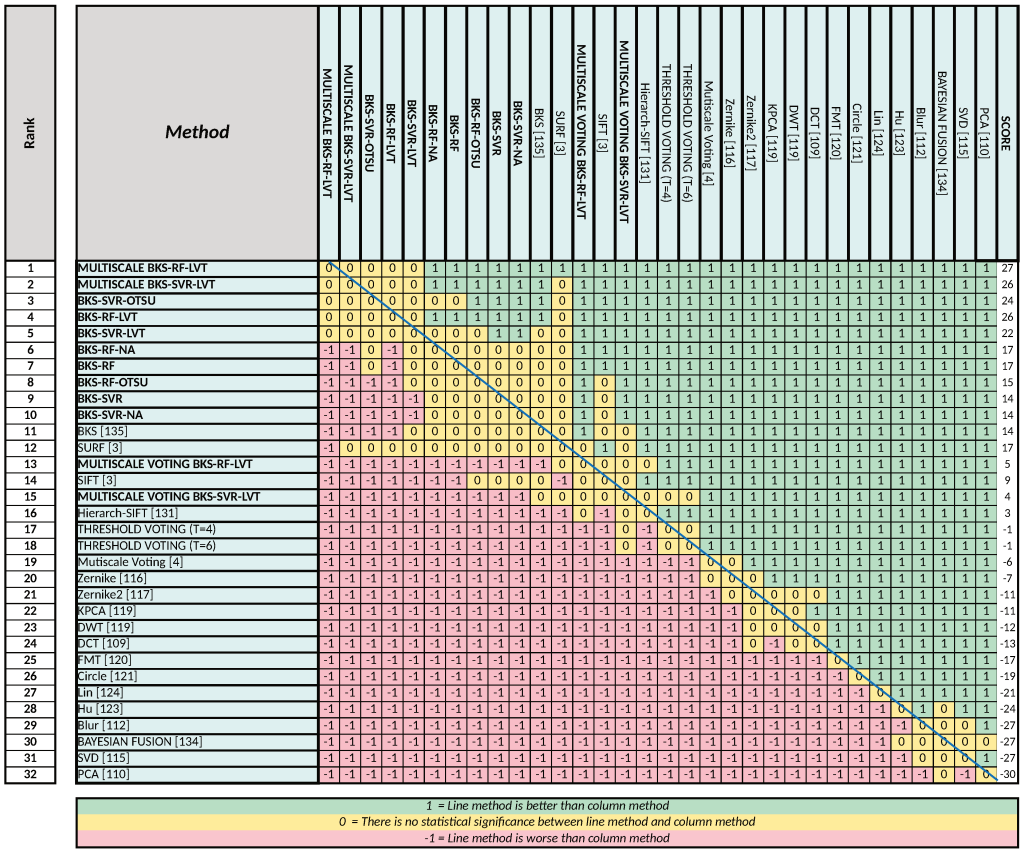
Legend:
xx.xx = Five best methods in the column metric
xx.xx = Five worst methods in the column metric

Table 6.2: Experiments considering the compressed version of CPH dataset ($CPHCOMPRESSED$). The proposed methods are highlighted in bold and results are ordered by f-measure.

VOTING BKS-RF-LVT, MULTISCALE VOTING BKS-SVR-LVT) improved the basic voting method, although they were not better than other state-of-the-art approaches.

By performing Friedmann statistical test on the results calculated for this dataset, we found a p-value of 3.49×10^{-45}, which help us to state that the approaches have significantly different performances. By applying the Wilcoxon approach for pairwise comparisons, we found that the best proposed approach, which uses Multiscale BKS, Random Forests generative model and Local Variable Threshold (Multiscale BKS-RF-LVT), is better than 27 out of 31 compared approaches. It is not statistically different than the second, third, fourth and fifth best approaches, which are all variations of the proposed fusion procedure. Table 6.3 shows the results of Wilcoxon pairwise tests applied in this dataset.

Table 6.4 shows the results for the uncompressed version of CPH dataset ($CPHALL$). The results in this table corroborate the potential of the proposed multiscale approach, as there are more and better samples to fill the probabilities contained in this representation. Also, as discussed previously in section 6.2.1, the multiscale approach eliminates noise in the training samples, which could be regarded as copy-move pixels by common classifiers. The Random Forests generative model procedure was the best generative model for this problem, probably because it uses a summarization over the trees with complementary properties. As the BKS fusion uses classifiers that are complementary in copy-move detection in some way, this kind of regression is appropriate. Random Forests classifiers also performed very well in previous classifications tasks (non-forensics related) in the literature [179] outperforming SVMs, for instance. The best approach proposed approach yielded a 42% reduction of error if compared to the best state of the art: Multiscale Voting.



Rank	Method	MULTISCALE BKS-RF-LVT	MULTISCALE BKS-SVR-LVT	BKS-SVR-OTSU	BKS-RF-LVT	BKS-SVR-LVT	BKS-RF-NA	BKS-RF	BKS-RF-OTSU	BKS-SVR	BKS-SVR-NA	BKS [135]	SURF [3]	MULTISCALE VOTING BKS-RF-LVT	SIFT [3]	MULTISCALE VOTING BKS-SVR-LVT	Hierarch-SIFT [131]	THRESHOLD VOTING (T=4)	THRESHOLD VOTING (T=6)	Mutiscale Voting [4]	Zernike [116]	Zernike2 [117]	KPCA [119]	DWT [119]	DCT [109]	FMT [120]	Circle [121]	Lin [124]	Hu [123]	Blur [112]	BAYESIAN FUSION [134]	SVD [115]	PCA [110]	SCORE
1	**MULTISCALE BKS-RF-LVT**	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	27
2	**MULTISCALE BKS-SVR-LVT**	0	0	0	0	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	26
3	**BKS-SVR-OTSU**	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	24
4	**BKS-RF-LVT**	0	0	0	0	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	26
5	**BKS-SVR-LVT**	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	22
6	**BKS-RF-NA**	-1	-1	0	-1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
7	**BKS-RF**	-1	-1	0	-1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
8	**BKS-RF-OTSU**	-1	-1	-1	-1	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	15
9	**BKS-SVR**	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
10	**BKS-SVR-NA**	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
11	BKS [135]	-1	-1	-1	-1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
12	SURF [3]	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
13	**MULTISCALE VOTING BKS-RF-LVT**	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5
14	SIFT [3]	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	-1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	9
15	**MULTISCALE VOTING BKS-SVR-LVT**	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	4
16	Hierarch-SIFT [131]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3
17	THRESHOLD VOTING (T=4)	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
18	THRESHOLD VOTING (T=6)	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
19	Mutiscale Voting [4]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	-6
20	Zernike [116]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	-7
21	Zernike2 [117]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	1	1	1	1	1	1	1	1	-11
22	KPCA [119]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	1	1	1	1	1	1	1	1	-11
23	DWT [119]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	1	1	1	1	1	1	1	1	-12
24	DCT [109]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	1	1	1	1	1	1	1	1	-13
25	FMT [120]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	1	1	-17
26	Circle [121]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	1	-19
27	Lin [124]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	-21
28	Hu [123]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	0	1	1	-24
29	Blur [112]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	-27
30	BAYESIAN FUSION [134]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	-27
31	SVD [115]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	-27
32	PCA [110]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-30

1 = Line method is better than column method
0 = There is no statistical significance between line method and column method
-1 = Line method is worse than column method

Table 6.3: Wilcoxon f-measures paired tests results considering the approaches applied in the compressed version of CPH dataset ($CPHCOMPRESSED$). The proposed methods are highlighted in bold.

Finally, the Local Variable Threshold was used in the two best approaches, showing that it is better to dynamically adapt the decision threshold based on the behavior of a given neighborhood instead of a hard decision or even not considering the opinion of the neighbors.

Rank	Method	Statistics Calculated on CPHALL Dataset after 5X2 Cross-Validation Experiments							
		F-MEASURE (%)	ACC (%)	ACC (σ)	TPR (%)	TPR (σ)	FPR (%)	FPR (σ)	PRECISION (%)
1	**MULTISCALE BKS-RF-LVT**	**91.34**	**91.56**	**9.79**	**90.88**	**17.52**	**0.07**	**7.88**	**99.92**
2	**MULTISCALE BKS-SVR-LVT**	**86.37**	**83.16**	**7.82**	**92.23**	**14.13**	**25.91**	**6.27**	**78.07**
3	**BKS-SVR-OTSU**	**84.59**	**86.01**	**13.75**	**77.63**	**25.98**	**5.60**	**7.59**	**93.27**
4	**BKS-RF-LVT**	**83.26**	**86.28**	**14.79**	**83.70**	**28.35**	**0.11**	**7.79**	**99.87**
5	*Multiscale Voting [4]*	83.03	85.34	13.08	71.91	26.55	1.22	1.91	98.33
6	**BKS-SVR-LVT**	**81.48**	**86.64**	**16.38**	**79.49**	**31.99**	**0.06**	**4.19**	**99.92**
7	*BKS [5]*	81.32	83.90	18.29	70.33	36.35	2.52	3.33	96.54
8	**BKS-RF-NA**	**80.97**	**83.52**	**15.86**	**73.14**	**30.60**	**6.40**	**7.49**	**91.95**
9	**BKS-RF**	**80.48**	**83.15**	**15.42**	**72.72**	**29.75**	**6.40**	**7.28**	**91.91**
10	**BKS-RF-OTSU**	**80.30**	**82.86**	**15.96**	**71.73**	**30.65**	**6.00**	**7.29**	**92.28**
11	**BKS-SVR-NA**	**79.94**	**83.27**	**17.17**	**69.56**	**33.81**	**0.03**	**3.51**	**99.96**
12	*SURF [3]*	79.14	82.82	20.53	65.82	41.13	0.19	0.22	99.72
13	**BKS-SVR**	**77.83**	**81.80**	**17.01**	**66.88**	**33.32**	**3.27**	**3.89**	**95.34**
14	**MULTISCALE VOTING BKS-RF-LVT**	**71.11**	**77.62**	**19.12**	**56.53**	**38.47**	**0.01**	**2.05**	**99.98**
15	*SIFT [3]*	71.02	77.56	23.53	55.35	47.12	0.00	0.62	100.00
16	*BAYESIAN FUSION [134]*	70.08	77.06	18.86	54.17	37.76	0.05	0.07	99.91
17	**MULTISCALE VOTING BKS-SVR-LVT**	**69.43**	**76.85**	**19.09**	**54.50**	**38.31**	**0.79**	**1.21**	**98.57**
18	*THRESHOLD VOTING (T=4)*	67.02	75.28	19.01	50.61	38.07	0.04	0.06	99.93
19	*Hierarch-SIFT [131]*	66.78	75.13	21.51	50.45	43.20	0.20	0.24	99.61
20	*Zernike [116]*	51.11	67.13	15.87	34.43	31.63	0.15	0.78	99.56
21	*DWT [119]*	47.56	65.59	16.96	31.30	33.84	0.12	0.71	99.63
22	*KPCA [119]*	44.81	64.40	16.59	28.97	33.20	0.17	0.93	99.42
23	*DCT [109]*	42.79	63.63	15.74	27.29	31.47	0.03	0.11	99.88
24	*Zernike2 [117]*	38.47	59.94	19.82	25.40	38.35	5.51	7.19	82.17
25	*Circle [121]*	37.65	61.61	15.42	23.32	30.82	0.09	47.51	99.64
26	*THRESHOLD VOTING (T=6)*	35.03	60.67	15.61	21.34	31.20	0.00	0.00	99.94
27	*FMT [120]*	30.15	58.88	14.75	17.86	29.64	0.09	0.62	99.47
28	*SVD [115]*	28.50	58.34	15.32	16.71	30.64	0.03	0.10	99.83
29	*Hu [123]*	28.26	58.23	14.27	16.55	28.58	0.08	0.23	99.53
30	*Lin [124]*	27.08	57.86	14.38	15.75	28.79	0.02	0.13	99.86
31	*Blur [112]*	25.58	57.36	14.04	14.75	28.08	0.03	0.13	99.81
32	*PCA [110]*	24.87	57.12	13.83	16.92	27.66	0.03	0.14	99.80

Legend:
xx.xx = Five best methods in the column metric
xx.xx = Five worst methods in the column metric

Table 6.4: Experiments considering the uncompressed version of CPH dataset ($CPHALL$). The proposed methods are highlighted in bold and results are ordered by f-measure.

The Friedmann statistical test shows a p-value of 4.99×10^{-43}, which helps us to state that the approaches have significantly different performance. By applying the Wilcoxon pairwise tests, we found that the best proposed approach is statistically better than 30 out of 31 approaches compared. It is not statistical significant only with the second best approach, which is also proposed in this chapter. Table 6.5 shows Wilcoxon pairwise statistical test results applied in this dataset.

6.4.2 CMEN Dataset

Table 6.6 shows the results of the experiments on the $CMEN$ dataset. We show in this table only the results from the best proposed approaches, individual state-of-the-art classifiers used for the fusion and the state-of-the-art fusion methodologies. In this setup, we noticed similar f-measure classification performances to the previous results on $CPHALL$ dataset, with a reduction of error of 44% if compared to the best state of the art: SURF. The Friedmann statistical test shows a p-value of 4.37×10^{-25}, which helps us to state that the approaches have significantly different performances. By applying the Wilcoxon tests, we found that the best proposed approach is better than 15 out of 16 techniques compared. It is not only significantly different with the second best approach,

Rank	Method	MULTISCALE BKS-RF-LVT	MULTISCALE BKS-SVR-LVT	BKS-SVR-OTSU	BKS-RF-LVT	Multiscale Voting [4]	BKS-SVR-LVT	BKS [135]	BKS-RF-NA	BKS-RF	BKS-RF-OTSU	BKS-SVR-NA	SURF [3]	BKS-SVR	MULTISCALE VOTING BKS-RF-LVT	SIFT [33]	BAYESIAN FUSION [4]	MULTISCALE VOTING BKS-SVR-LVT	THRESHOLD VOTING (T=4)	Hierarch-SIFT [131]	Zernike [116]	DWT [119]	KPCA [119]	DCT [109]	Zernike2 [117]	Circle [121]	THRESHOLD VOTING (T=6)	FMT [120]	SVD [115]	Hu [123]	Lin [124]	Blur [112]	PCA [110]	SCORE
1	**MULTISCALE BKS-RF-LVT**	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	30
2	**MULTISCALE BKS-SVR-LVT**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	14
3	**BKS-SVR-OTSU**	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	18
4	**BKS-RF-LVT**	-1	0	0	0	0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	22
5	Multiscale Voting [4]	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
6	**BKS-SVR-LVT**	-1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	19
7	BKS [135]	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	18
8	**BKS-RF-NA**	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
9	**BKS-RF**	-1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
10	**BKS-RF-OTSU**	-1	0	0	-1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	16
11	**BKS-SVR-NA**	-1	0	0	-1	0	0	0	0	0	0	0	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	16
12	SURF [3]	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	15
13	**BKS-SVR**	-1	0	0	-1	0	-1	0	0	0	-1	0	0	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	11
14	**MULTISCALE VOTING BKS-RF-LVT**	-1	0	-1	-1	0	-1	-1	-1	0	0	-1	0	-1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	5
15	SIFT [3]	-1	0	-1	-1	-1	-1	-1	0	0	-1	0	-1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	7
16	BAYESIAN FUSION [134]	-1	0	-1	-1	-1	-1	-1	-1	0	-1	-1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	4
17	**MULTISCALE VOTING BKS-SVR-LVT**	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	2
18	THRESHOLD VOTING (T=4)	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
19	Hierarch-SIFT [131]	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	-1
20	Zernike [116]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	1	1	1	1	1	1	-7
21	DWT [119]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	1	1	1	1	1	-9
22	KPCA [119]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	0	1	1	1	1	1	1	1	-12
23	DCT [109]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	1	-14
24	Zernike2 [117]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	1	1	1	1	-15
25	Circle [121]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	1	1	1	1	-16
26	THRESHOLD VOTING (T=6)	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	1	1	1	1	-18
27	FMT [120]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	1	1	1	-21
28	SVD [115]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	-24
29	Hu [123]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	-25
30	Lin [124]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	1	-26
31	Blur [112]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	-29
32	PCA [110]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-31

1 = Line method is better than column method
0 = There is no statistical significance between line method and column method
-1 = Line method is worse than column method

Table 6.5: Wilcoxon f-measures paired tests results considering the approaches applied in the uncompressed version of CPH dataset ($CPHALL$). The proposed methods are highlighted in bold.

which is also proposed in this chapter, which in turn, is also statistically better than 15 approaches. Table 6.7 shows the results of Wilcoxon paired tests for this dataset.

Rank	Method	*Statistics Calculated on CMEN Dataset after 5X2 Cross-Validation Experiments*							
		F-MEASURE (%)	*ACC (%)*	*ACC (σ)*	*TPR (%)*	*TPR (σ)*	*FPR (%)*	*FPR (σ)*	*PRECISION (%)*
1	***MULTISCALE BKS-RF- LVT***	**89.96**	**90.76**	**14.36**	**82.95**	**28.41**	**1.43**	**3.15**	**98.31**
2	***BKS-RF-LVT***	**88.72**	**89.76**	**14.51**	**81.10**	**28.73**	**1.57**	**2.85**	**98.10**
3	***MULTISCALE BKS-SVR-LVT***	**87.12**	**88.52**	**15.67**	**78.03**	**31.27**	**0.98**	**2.27**	**100.00**
4	***BKS-SVR- OTSU***	**86.49**	**87.95**	**13.06**	**77.35**	**25.38**	**1.44**	**3.34**	**98.17**
5	***BKS-SVR- LVT***	**83.64**	**85.91**	**16.28**	**72.55**	**32.55**	**0.72**	**1.60**	**100.00**
6	*SURF [3]*	80.40	83.50	19.97	67.89	39.41	0.87	2.42	100.00
7	*BKS [135]*	79.52	82.76	17.57	66.00	35.18	0.48	1.32	100.00
8	*SIFT [3]*	75.49	80.03	21.30	60.63	42.96	0.56	1.19	100.00
9	*Multiscale Voting [4]*	74.56	78.33	17.77	59.45	37.24	2.79	5.23	99.97
10	*THRESHOLD VOTING (T=4)*	68.15	75.73	19.42	51.69	38.82	0.22	1.01	100.00
11	*Zernike2 [117]*	66.35	73.98	20.74	49.65	40.36	1.69	3.30	99.98
12	*Zernike [116]*	62.38	72.56	16.79	45.33	33.39	0.20	1.35	100.00
13	*DCT [109]*	54.22	68.53	16.79	37.19	33.55	0.13	0.53	100.00
14	*KPCA [119]*	48.51	65.93	15.66	32.02	31.32	0.14	0.77	100.00
15	*THRESHOLD VOTING (T=6)*	42.48	63.46	17.02	26.97	34.00	0.04	0.43	100.00
16	*Hierarch-SIFT [131]*	39.51	61.99	17.72	24.62	35.70	0.64	2.56	100.00
17	*BAYESIAN FUSION [134]*	8.48	52.20	2.01	4.43	4.02	0.03	0.16	100.00

Legend:
xx.xx = Five best methods in the column metric
xx.xx = Five worst methods in the column metric

Table 6.6: Experiments considering the $CMEN$ dataset. The proposed methods are highlighted in bold and results are ordered by f-measure.

Rank	Method	MULTISCALE BKS-RF- LVT	BKS-RF-LVT	MULTISCALE BKS-SVR-LVT	BKS-SVR- OTSU	BKS-SVR- LVT	SURF [3]	BKS [135]	SIFT [3]	Multiscale Voting [4]	THRESHOLD VOTING (T=4)	Zernike2 [117]	Zernike [116]	DCT [109]	KPCA [119]	THRESHOLD VOTING (T=6)	Hierarch-SIFT [131]	BAYESIAN FUSION [134]	SCORE
1	**MULTISCALE BKS-RF- LVT**	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	15
2	**BKS-RF-LVT**	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	15
3	**MULTISCALE BKS-SVR-LVT**	-1	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	11
4	**BKS-SVR- OTSU**	-1	-1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	11
5	**BKS-SVR- LVT**	-1	-1	-1	-1	0	0	0	0	1	1	1	1	1	1	1	1	1	5
6	SURF [3]	-1	-1	-1	-1	0	0	0	1	1	1	1	1	1	1	1	1	1	6
7	BKS [135]	-1	-1	-1	-1	0	0	0	0	1	1	1	1	1	1	1	1	1	5
8	SIFT [3]	-1	-1	-1	-1	0	-1	0	0	0	1	1	1	1	1	1	1	1	3
9	Multiscale Voting [4]	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	1	1	1
10	THRESHOLD VOTING (T=4)	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	-3
11	Zernike2 [117]	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	1	1	1	1	1	-3
12	Zernike [116]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	1	-6
13	DCT [109]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	1	-8
14	KPCA [119]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	1	1	1	-10
15	THRESHOLD VOTING (T=6)	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	-13
16	Hierarch-SIFT [131]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	1	-13
17	BAYESIAN FUSION [134]	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-16

1 = Line method is better than column method
0 = There is no statistical significance between line method and column method
-1 = Line method is worse than column method

Table 6.7: Wilcoxon f-measures paired tests results considering the approaches applied in the $CMEN$ dataset. The proposed methods are highlighted in bold.

6.4.3 Different Forgery Sizes

To check the effectiveness of the proposed method, we analyze its detection accuracy considering different forgery sizes. For that, we divide the images in CPH dataset into three sets of images, considering the ratio of fake pixels to the whole images (number of modified pixels divided by the number of total pixels in an image). After that, we create three sets of images from CPH dataset, obeying the criteria shown in Table 6.8. Table 6.9 shows the mean 5×2 cross-validation accuracy of each CPH sub-datasets, comparing the proposed method to the best performing in the literature.

The proposed fusion approach improves upon the common BKS approach and the classification of each individual method used in the fusion. As expected, the larger the forgery the easier the detection regardless of the detection method used.

Subdataset ID	Tamper Region Size	Number of Images
#1	0% <= r_fake_original <= 3%	42
#2	3% < r_fake_original <= 5%	40
#3	r_fake_original > 5%	26

Table 6.8: Set of images extracted from CPH dataset to check the effectiveness of the proposed method over tampering size variation.

Approach/Subdataset ID	#1	#2	#3
SIFT [3]	74.07%	77.33%	83.56%
ZERNIKE [116]	61.90%	71.61%	68.73%
BKS [135]	79.73%	85.54%	88.14%
MULTISCALE BKS-RF-LVT	**88.88%**	**92.50%**	**93.52%**

Table 6.9: Classification results of the proposed approach (in bold) against some state of the art considering variation size of tampered regions.

6.4.4 Different Compression Qualities

To assess the performance of the proposed method under different compression setups, we take the CPH uncompressed images dataset and create three new versions of it, called $CPHCOMPRESSED\_90$, $CPHCOMPRESSED\_80$ and $CPHCOMPRESSED\_70$, which are composed by compressed images considering image quality factors 90, 80 and 70, respectively. Table 6.10 shows the mean 5×2 cross-validation accuracy for each dataset, comparing the proposed method to the best performing in the literature. It can be seen from this Table that our approach significantly outperforms the existing methods regardless of the compression conditions, specially when we consider that these results are measured at pixel-level.

APPROACH/DATASET	CPHCOMPRESSED_90	CPHCOMPRESSED_80	CPHCOMPRESSED_70
SIFT [3]	78.00%	74.65%	74.53%
ZERNIKE [116]	64.92%	63.48%	62.35%
BKS [135]	82.72%	80.81%	79.29%
MULTISCALE BKS-RF-LVT	**85.73%**	**84.62%**	**82.94%**

Table 6.10: Classification results of the proposed approach (in bold) against some state of the art considering compression variation of images.

6.4.5 Noise Variation

We also tested the proposed method under varying noise conditions. For that, we also considered images in the CPH uncompressed images dataset, adding white Gaussian noises with varying variances, called $CPHNOISE\_1$, $CPHNOISE\_2$ and $CPHNOISE\_4$, which are composed by uncompressed images with white Gaussian noises with variances (σ^2) equal to 0.0001, 0.0002 and 0.0004 respectively. These values were chosen because they do not affect the visual image quality, which would, otherwise allow the detection of a forgery by a simple visual inspection. Table 6.11 shows the mean 5×2 cross-validation accuracy of each dataset, comparing the proposed method against some of the proposed literature. The results shown highlights the benefits of the multiscale BKS Table to create samples robust to noise in the training step.

6.4.6 Running Times

To compare the running time of the proposed method against its counterparts in the literature we separate 20 images from one test combination of our 5×2 cross validation

Approach/Dataset	σ^2=0.0001	σ^2=0.0002	σ^2=0.0004
SIFT [3]	77.89%	76.64%	73.95%
ZERNIKE [116]	64.79%	63.70%	62.48%
BKS [135]	81.63%	79.77%	77.86%
MULTISCALE-BKS-RF-LVT	**87.49%**	**86.65%**	**84.34%**

Table 6.11: Classification results of the proposed approach (in bold) against some state of the art considering noise variation of images.

and then evaluated the mean running time per image. We used these 20 images in the experiments because most of them have the same dimensions and thus we want to decrease as much as possible image dimension effects in running times. We ran the experiments on an Intel(R) Core(TM) i7-5820K CPU @3.30GHz with 62GB of RAM. Table 6.12 shows the running times of some existing methods in the literature along with the proposed method showing that, as expected, the proposed method is less efficient than its counterparts. This happens because, as it consists of running different methods (currently eight of them) individually, search the outputs combination in the BKS representational space and then, for each pixel in the probability map resulted, calculate the decision value to classify a pixel according to its neighborhood. However, all of these tasks can be easily parallelized and does not represent a problem in face of the important advances obtained in terms of method's effectiveness when detecting forgeries.

APPROACH	Mean Running Time Per Image (s)
SIFT [3]	2.43
ZERNIKE [116]	42.08
ZERNIKE2 [117]	2025.85
BKS [135]	2893.75
MULTISCALE BKS-RF-LVT	2955.91

Table 6.12: Mean running times per image (in seconds) of the proposed method (in bold) against some state of the art methods used in the experiments.

6.4.7 Qualitative Analysis

We now turn our attention to a qualitative analysis of the results, showing how the reported f-measure results influence the final classification in the binary map generated, which is important for helping a forensic researcher deciding whether or not a given image is forged. Figure 6.2 depicts the detection maps generated by our best proposed approach and the ones generated by the best state-of-the-art applied in this scenario.

6.5 Final Considerations and Further Developments

Image tampering detection is a hard problem to solve because it involves different methodologies and abilities. This way, it is impossible that just one image tampering detection approach reveals perfectly an image manipulation. Also, any image manipulation detector can be deceived by anti-forensic operations created by a forger.

Figure 6.2: Qualitative results, showing the binary detection maps useful to compare some of the proposed approaches and the state of the art on compressed and uncompressed images. First column shows the images of the database, second column shows the labeled ground truth and the four following columns show the binary maps generated by the SURF classifier [3], by Multiscale Voting classifier [4], by the THRESHOLD VOTING fusion approach and by our best proposed approach MULTISCALE BKS_RF_LVT

In this sense, the combination of different detectors is promising and paramount, as it can explore complementary properties from the combined detectors. However, traditional

classifier fusion approaches in the literature failed to completely solve the problem because they often did not consider important intrinsic properties of the digital image forensic scenario: conditional and spatial dependence of tampered pixels with respect to their neighboring pixels.

To address this problem, we explored approaches to combine methods that investigate the best of two worlds in the copy-move detection problem: block-based and points of interest detection methods. We proposed three extensions for Behavior Knowledge Space representation fusion: the multi-scale BKS representations, generative models to complete missing information in the BKS representation and multi-directional neighborhood analysis to integrate the neighborhood behavior into the decision-making process of a given pixel.

The proposed approaches proved to perform better than existing ones for fusion and for individual detectors considering either compressed or uncompressed images. The main reasons are: (1) the multi-scale approaches act by giving samples to make the classifier robust to common post-processing operations in tampering, such as noise and resizing, to feed the Behavior Knowledge Space. This occurs because the Gaussian image decomposition does Gaussian filtering, making available images without noise to the feeding of probabilities and it also creates images with multiple dimensions and so, with different forgery sizes; (2) the generative models aim at completing the remaining conditional probabilities not present in BKS tables and potentially eliminating noise and outliers in the existing entries; and (3) the multi-directional approaches perform the classification of a pixel by investigating also its neighbors, a major difference with respect to previous fusion methods used in this problem.

However, the method has two main drawbacks: the first one happens when there is no complementarity of the underlying methods to be combined. This happens when we combine block- and interest point-based methods and the questioned image has several homogeneous regions, on which the block-based approaches fail and there are no interest points enough to be extracted from the image. Finally, the proposed method is slightly less efficient than its counterparts as it involves combining k detection methods and evaluating the probability of their outcomes for defining the final detection map.

With our proposed methods, we conclude that the tampering conditional analysis is essential, and this is done by Behavior Knowledge Space Representation. Besides that, it is important to consider the pixel spatial dependency. The top best classification results in all experiments proved that using the Local Variable Threshold multi-directional neighborhood analysis is suitable to this task. In addition, we solved the inherent BKS representation problem when dealing with complex issues as the location of image manipulation: lack of data. To deal with it, we proposed to learn, from just a few examples available in the training data, the conditional dependence of tampering operations from a set of used individual detectors. We also have compensated this lack of training data by using multi-scale decomposition of the input data allied with generative models to calculate missing probabilities. These decisions make it possible to conclude that generative models are a key ally to build more robust BKS representation spaces and better tackle the problem of detecting forgeries in images.

As future work, one promising investigation would be improving the detection methods to also consider possible counter-forensic techniques. In an adversarial attack scenario, it

is possible that simple methods and also basic fusion approaches will easily break down. Robust fusion methods as the ones discussed herein are naturally more resilient to such attacks especially if we model some possible attacks in the construction of the detection method itself. This could be done by considering possible attacks in the training images and some methods to respond to such attacks, which could be incorporated in the low-level detection step before building the BKS representations. Such new developments and studies would be paramount for the next stage in digital forensics. Finally, we also aim at considering other generative models, such as Expectation Maximization, to fill the BKS tables.

Part IV

Conclusion

Chapter 7

Conclusions and Future Work

7.1 Final Considerations

Digital Image Forensics procedures normally involve the search for characteristic artifacts that can represent well defined aspects of a questioned document. To perform source attribution of a crime-proof document, for example, patterns specific to the device, intrinsically or extrinsically inserted, may be searched for in the generated data. To identify an image forgery composed by the splicing of two or more images, several artifacts can be searched for, such as light inconsistencies in regions of the questioned image, unusual stitches around a suspected alien region of the image, among others. These artifacts are regarded as structural inconsistencies and are the cornerstone of several digital image forensic algorithms. As the clues for image forensic are different, it is only natural that existing solutions seek different paradigms to solve each problem.

In this thesis, following a slightly different approach toward solving varying forensic problems, we propose a main idea that can be the basis of several digital image forensic algorithms: the multi-analysis. This approach considers several scenarios in the investigation of the digital image that can be combined or not, such as multiple directionality, multiple resolutions, multiple perturbations, multiple representations and multiple data. All of these take into account the complexity of digital forensic-interest data and can be used for different applications. One distinguishing aspect of the proposed approach in this thesis is the fact that it can be used in any step (pre-processing, description and classification) of the forensic analysis of a questioned image.

Multi-analysis considering multi-perturbation proves to be important in the forensic analysis of a questioned image as it highlights structural inconsistencies that can be present in the image, artificially inserting new artifacts where they can already exist. This can differentiate, for example, images with or without structural inconsistencies, such as pristine and fake images. We showed in Chapter 5 how the multi-perturbation scenario is effective for Median Filtering Detection, even in cross-dataset experiments.

The multi-directionality analysis, in turn, is important for the digital image forensic approach as this scenario considers that the image data is naturally interpolated, so any pixel is correlated with its neighbors and, in some sense, depends on them. Thus, multi-directionality is important for finding structural inconsistencies in the data. We showed that for source attribution, in Chapter 3, the banding should be described using

all directions of the questioned image and, in Chapter 5, we showed that for classifying a pixel as pristine or fake all neighbors' behavior should be considered.

The multi-resolution analysis shows effectiveness as it considers that the structural inconsistency can happen in multiple scales, as discussed in Chapter 3; this scenario also inserts artifacts considering filters with non-fixed sizes as discussed in Chapter 5 and creates additional samples for training a classifier invariant to noisy and resized fake images, as discussed in Chapter 6.

Finally, multiple representation is important in our multi-analysis because this scenario can enhance artifacts present in parts of the data by doing different transformations in such a way that the structural inconsistency can be better represented. If used along with multiple data, different patches of interest can be extracted from the image data and all the individual analyses can be aggregated at the end. We show that different representations (image transforms) of multiple data (regions of interest) are important to understand printing artifacts with data-driven approaches for laser printer attribution as we discussed in Chapter 4.

We consider that the main contribution of this thesis for the digital image forensic area is the introduction of a new idea to be considered when facing an investigation of questioned documents. Notwithstanding, the proposed approach must be considered also with a grain of salt. In the case of uncompressed image analysis while detecting median filtering, for instance, multi-analyses techniques were not useful at pinpointing additional features for improving the classifications results, as discussed in Chapter 5. Moreover, if efficiency is an issue, the multi-analyses techniques considered herein must be taken very carefully. Normally, adding a series of data transformations, neighborhood analyses, multiple perturbations and inspections at different resolutions of the input data when seeking for different structural artifacts, can represent a computational burden someone is not willing to pay.

7.2 Publications Related to the Thesis

As for the publication of the ideas in this thesis we had until this moment three publications and two submissions to top-tier Journals and Conferences in Forensics, Computer Vision and Machine Learning. The writing of this thesis was inspired in the following publications:

- Contents in Chapter 3 led to the publication of an article in the Elsevier Forensic Science International (FSI) journal [152].
- Chapter 4 led to an article (currently under review) submitted to the IEEE Transactions on Image Forensic and Security (T-IFS) journal [180];
- Chapter 5 contents led to two articles; one published in the Iberoamerican Congress on Pattern Recognition (CIARP) congress [181] and its extension published in the Journal of Intelligent Data Analysis (J-IDA) [182].
- Chapter 6 led to an article (currently under review) submitted to the IEEE Transactions on Image Processing (T-IP) [183].

Table 7.1 shows a summary of all algorithms developed in this thesis. All of them are available for download, with the links being provided within the chapters of this thesis.

Method	STEP OF QUESTIONED IMAGE ANALYSIS			MULTI-ANALYSES CONSIDERED					APPLICATION		
	Pre-Processing	Description	Classification	Multi-Perturbation	Multi-Scale	Multi-Directionality	Multi-Representation	Multi-Data	Source Attribution	Median Filtering Detection	Copy-Move Detection
CTGF_GLCM_MD_MS	✓	✓			✓	✓			✓		
GLCM_MD		✓				✓			✓		
GLCM_MD_MS	✓	✓			✓	✓			✓		
CTGF		✓				✓			✓		
DL_EARLY_FUSION	✓						✓		✓		
DL_LATE_FUSION	✓						✓	✓	✓		
TPOW	✓			✓		✓				✓	
TPMW	✓			✓	✓	✓				✓	
FPMW	✓			✓	✓	✓				✓	
MULTISCALE BKS-RF-LVT			✓		✓	✓					✓
MULTISCALE BKS-SVR-LVT			✓		✓	✓					✓

Table 7.1: Summary of the 11 main approaches based on multi-analysis discussed in this thesis.

7.3 Future Work

We envision a wide range of possible future works based on open issues discussed in this thesis, as our scenarios can be easily adapted for being applied in several other digital image forensic applications. We aim at investigating new scenarios that can be used in the multi-analysis, such as the *multiple classification*, which involves the use of open set classifiers in several classification steps. We also aim at proposing handcrafted feature approaches for other digital image forensic applications, such as the use of multi-scale multi-perturbations for image splicing detection and source attribution of other devices; proposing new multi-analysis data-driven approaches focused on median filtering detection; studying multi-analysis classification procedures that can be used for open-set classification and proposing multi-analysis procedures by multiple representation of multiple data for identifying criminal sites in remote sensing images. Finally, we also intend to act in the reverse direction, proposing new anti-forensic techniques using the proposed scenarios to erase or minimize the structural inconsistencies present in criminal-interest documents and, in doing so, gather some knowledge on how to improve new detection techniques being developed.

Appendix A

Supplementary Information

A.1 Gray-Level Co-Occurrence Matrices Features

The work of Miklineni et al [68] proposed a set of features calculated on top of Gray-Level Co-Occurrence matrices. We use this same set of features in this work.

Before the features are calculated a set of definitions are extracted from the image: (1) Number of pixels R in a Region of Interest (ROI), which is the set of all pixels within the printed area of the character; (2) The gray-level co-occurrence matrices $glcm(n, m)$, which are two-dimensional histograms per neighborhood direction (dr, dc) showing the occurrence of pixels n and m in a given distance (dr, dc); (3) The number of neighboring ROI pixels distant by a (dr, dc) offset R_{glcm}; (4) GLCM probability estimates p_{glcm}; (5) marginal probability densities in the row and column directions p_r and p_c; (6) histograms of differences $D(k)$; (7) Histograms of sums $S(k)$ and its mean μ_S; (8) Mean pixel of a ROI and (9) density of a ROI. Equations A.1 to A.11 formalize these calculations.

$$R = \sum_{(i,j)\in ROI} 1 \tag{A.1}$$

$$glcm(n, m) = \sum_{(i,j),(i+dr,j+dc)\in ROI} 1_{\{I(i,j)=n, I(i+dr,j+dc)=m\}} \tag{A.2}$$

$$R_{glcm} = \sum_{(i,j),(i+dr,j+dc)\in ROI} 1 \tag{A.3}$$

$$p_{glcm}(n, m) = \frac{1}{R_{glcm}} glcm(n, m) \tag{A.4}$$

$$p_r(n) = \sum_{m=0}^{255} p_{glcm}(n, m) \tag{A.5}$$

$$p_c(m) = \sum_{n=0}^{255} p_{glcm}(n, m) \tag{A.6}$$

$$D(k) = \sum_{\substack{0 \le n \le 255 \\ 0 \le m \le 255 \\ |n-m|=k}} p_{glcm}(n,m) \tag{A.7}$$

$$S(k) = \sum_{\substack{0 \le n \le 255 \\ 0 \le m \le 255 \\ n+m=k}} p_{glcm}(n,m) \tag{A.8}$$

$$\mu_S = \sum_{k=0}^{510} kS(k) \tag{A.9}$$

$$\mu_{ROI} = \frac{1}{R} \sum_{(i,j) \in ROI} I(i,j) \tag{A.10}$$

$$p_{ROI}(k) = \frac{1}{R} 1_{\{I(i,j)=k\}} \tag{A.11}$$

Eleven features are calculated from the data in Equations A.1 to A.6. The first four are marginal means and variances defined in Equations A.12 to A.15.

$$\mu_r = \sum_{n=0}^{255} np_r(n) \tag{A.12}$$

$$\mu_c = \sum_{m=0}^{255} mp_c(m) \tag{A.13}$$

$$\sigma_r^2 = \sum_{n=0}^{255} n^2 p_r(n) - \mu_r^2 \tag{A.14}$$

$$\sigma_c^2 = \sum_{m=0}^{255} m^2 p_c(m) - \mu_c^2 \tag{A.15}$$

The next seven features are the energy of the normalized GLCM, three entropy measurements, the maximum entry in the GLCM, and two correlation metrics.

$$E = \sum_{n=0}^{255} \sum_{m=0}^{255} p_{glcm}^2(n,m) \tag{A.16}$$

$$H_{rc1} = -\sum_{n=0}^{255} \sum_{m=0}^{255} p_{glcm}(n,m) \log_2(p_r(n)p_c(m)) \tag{A.17}$$

$$H_{rc2} = -\sum_{n=0}^{255} \sum_{m=0}^{255} p_r(n)p_c(m) \log_2(p_r(n)p_c(m)) \tag{A.18}$$

$$H_{glcm} = -\sum_{n=0}^{255} \sum_{m=0}^{255} p_{glcm}(n,m) \log_2(p_{glcm}(n,m)) \tag{A.19}$$

$$P_{max} = \max\{p_{glcm(n,m)}\} \tag{A.20}$$

$$\rho_1 = \sum_{n=0}^{255}\sum_{m=0}^{255} \frac{(n-\mu_r)(m-\mu_c)p_{glcm}(n,m)}{\sigma_r\sigma_c} \tag{A.21}$$

$$\rho_2 = \sum_{n=0}^{255}\sum_{m=0}^{255} |n-m|(n+m-\mu_r-\mu_c)p_{glcm}(n,m) \tag{A.22}$$

Four features, Equations A.23 to A.26, are obtained from the difference histogram $D(k)$ defined by Equation A.7. They are the energy, entropy, inertia, and local homogeneity of $D(k)$ respectively.

$$E_D = \sum_{k=0}^{255} D^2(k) \tag{A.23}$$

$$H_D = -\sum_{k=0}^{255} D(k)\log_2 D(k) \tag{A.24}$$

$$I_D = \sum_{k=0}^{255} k^2 D(k) \tag{A.25}$$

$$h_D = \sum_{k=0}^{255} \frac{D(k)}{1+k^2} \tag{A.26}$$

Five features, Equations A.27 to A.31, are obtained from the sum $S(k)$ histogram defined by Equation A.8 and A.9. They are the energy, entropy, variance, cluster shade, and cluster prominence of $S(k)$, respectively.

$$E_S = \sum_{k=0}^{510} S^2(k) \tag{A.27}$$

$$H_S = -\sum_{k=0}^{510} S(k)\log_2 S(k) \tag{A.28}$$

$$sigma_S^2 = \sum_{k=0}^{510} (k-\mu_S)^2 S(k) \tag{A.29}$$

$$A = \sum_{k=0}^{510} \frac{(k-\mu_r-\mu_c)^3 S(k)}{(\sigma_r^2-\sigma_c^2+2\sigma_r\sigma_c)^{\frac{3}{2}}} \tag{A.30}$$

$$B = \sum_{k=0}^{510} \frac{(k-\mu_r-\mu_c)^4 S(k)}{(\sigma_r^2-\sigma_c^2+2\sigma_r\sigma_c)^2} \tag{A.31}$$

The last two features use data in Equations A.10 and A.11. These are the ROIs

variance and entropy, as shown in Equations A.32 and A.33.

$$sigma^2_{ROI} = \frac{1}{R} \sum_{(i,j) \in ROI} (I(i,j) - \mu_{ROI})^2 \tag{A.32}$$

$$H_{ROI} = -\sum_{k=0}^{255} p_{ROI}(k) \log_2 p_{ROI}(k), \tag{A.33}$$

which completes the the set of 22 GLCM features considered.

Bibliography

[1] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An Evaluation of Popular Copy-Move Forgery Detection Approaches," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 6, pp. 1841–1854, December 2012.

[2] R. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *Transactions on Systems, Man and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[3] B. L. Shivakumar and S. Baboo, "Detection of Region Duplication Forgery in Digital Images Using SURF," *International Journal of Computer Science Issues*, vol. 8, pp. 199–205, 2011.

[4] E. Silva, T. Carvalho, A. Ferreira, and A. Rocha, "Going deeper into copy-move forgery detection: Exploring image telltales via multi-scale analysis and voting processes," *Journal of Visual Communication and Image Representation*, vol. 29, no. 0, pp. 16–32, 2015.

[5] J. Dong, W. Wang, and T. Tan, "CASIA Tampered Image Detection Evaluation Database," 2010, database avaiable at http://forensics.idealtest.org/.

[6] Youtube, "Statistics," https://www.youtube.com/yt/press/statistics.html.

[7] A. Walker, "Active Users and Other Statistics on Social Media in 2014 (Infographic)," http://www.nukesuite.com/active-users-and-other-statistics-on-social-media-in-2014/, November 2014.

[8] Zephoria, "The Top 20 Valuable Facebook Statistics," https://zephoria.com/top-15-valuable-facebook-statistics/, December 2015.

[9] D. L. M. Sacchi, F. Agnoli, and E. F. Loftus, "Changing History: Doctored Photographs Affect Memory for Past Public Events," *Applied Cognitive Psychology*, vol. 21, no. 8, pp. 249–273, August 2007.

[10] B. Crumley, "France may put warning labels on airbrushed photos," http://www.time.com/time/world/article/0,8599,1927227,00.html, 2009.

[11] C. Hun, "Disgraced cloning in south korea," http://www.nytimes.com/2009/10/27/world/asia/27clone.html, 2006.

[12] E. Kavanaugh, "Editorial expression of concern," *Science*, no. 314, pp. 592–594, 2006.

[13] D. Parrish and B. Noonan, "Image Manipulation as Research Misconduct," *Science and Engineering Ethics (SEE)*, vol. 15, pp. 161–167, January 2009.

[14] Internet Watch Foundation, "IWF Operational Trends 2014: Overview," https://www.iwf.org.uk/resources/trends, 2014.

[15] L. Kuntz, "Pirates and the Paper Chase," *UNESCO Courier*, vol. 3, no. 54, p. 41, 2001.

[16] X.-Y. Luo, D.-S. Wang, P. Wang, and F.-L. Liu, "A review on blind detection for image steganography," *Signal Processing*, vol. 88, no. 9, pp. 2138–2157, 2008.

[17] A. Cheddad, J. Condel, K. Curran, and P. McKevitt, "Review: Digital Image Steganography: Survey and Analysis of Current Methods," *Signal Processing*, vol. 90, no. 3, pp. 727–752, Mar. 2010.

[18] Y. Huo, H. He, and F. Chen, "A Semi-fragile Image Watermarking Algorithm with Two-stage Detection," *Multimedia Tools and Applications*, vol. 72, no. 1, pp. 123–149, 2014.

[19] G. Birajdar and V. Mankar, "Digital Image Forgery Detection Using Passive Techniques: A Survey," *Digital Investigation*, vol. 10, no. 3, pp. 226–245, 2013.

[20] W. Wang, J. Dong, and T. Tan, *Proceedings of Intl. Workshop on Digital Watermarking*. Springer, 2009, ch. A Survey of Passive Image Tampering Detection, pp. 308–322.

[21] Z. Zhang, Y. Ren, X.-J. Ping, Z.-Y. He, and S.-Z. Zhang, "A survey on passive-blind image forgery by doctor method detection," in *Intl. Conference on Machine Learning and Cybernetics*, vol. 6, 2008, pp. 3463–3467.

[22] P.-J. Chiang, N. Khanna, A. Mikkilineni, M. Segovia, J. Allebach, G. Chiu, and E. Delp, *Intelligent Multimedia Analysis for Security Applications*. Springer, 2010, ch. Printer and Scanner Forensics: Models and Methods, pp. 145–187.

[23] T. Lanh, K.-S. Chong, S. Emmanuel, and M. Kankanhalli, "A Survey on Digital Camera Image Forensic Methods," in *IEEE Intl. Conference on Multimedia and Expo*, 2007, pp. 16–19.

[24] T. Huynh, K. Huynh, T. Le-Tien, and S. Nguyen, "A survey on Image Forgery Detection techniques," in *IEEE Intl. Conference on Computing Communication Technologies - Research, Innovation, and Vision for the Future (RIVF)*, Jan 2015, pp. 71–76.

[25] R. Granty, T. Aditya, and S. Madhu, "Survey on passive methods of image tampering detection," in *Intl. Conference on Communication and Computational Intelligence (INCOCCI)*, Dec 2010, pp. 431–436.

[26] H. Li and J. Zheng, *Intelligent Science and Intelligent Data Engineering: Second Sino-foreign-interchange Workshop, IScIDE 2011, Xi'an, China, October 23-25, 2011, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Blind Detection of Digital Forgery Image Based on the Edge Width, pp. 546–553.

[27] A. Kashyap, B. Suresh, M. Agrawal, H. Gupta, and S. Joshi, "Detection of splicing forgery using wavelet decomposition," in *Intl. Conference on Computing, Communication Automation (ICCCA)*, 2015, pp. 843–848.

[28] M. Qureshi and M. Deriche, "A review on copy move image forgery detection techniques," in *Intl. Multi-Conference on Systems, Signals Devices (SSD)*, Feb 2014, pp. 1–5.

[29] T.-T. Ng, S.-F. Chang, J. Hsu, L. Xie, and M.-P. Tsui, "Physics-motivated features for distinguishing photographic images and computer graphics," in *ACM Intl. Conference on Multimedia*, ser. MULTIMEDIA '05, 2005, pp. 239–248.

[30] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *ACM Transactions on Graph.*, vol. 23, no. 3, pp. 303–308, Aug. 2004.

[31] ——, "Lazy snapping," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04. New York, NY, USA: ACM, 2004, pp. 303–308.

[32] J. Sun, J. Jia, and H. Shum, "Poisson matting for images," Dec. 22 2009, uS Patent 7,636,128. [Online]. Available: http://www.google.com/patents/US7636128

[33] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum, "Poisson matting," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04. New York, NY, USA: ACM, 2004, pp. 315–321.

[34] ——, "Poisson matting," *ACM Transactions on Graph.*, vol. 23, no. 3, pp. 315–321, Aug. 2004.

[35] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image Completion with Structure Propagation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 861–868, Jul. 2005.

[36] ——, "Image completion with structure propagation," in *ACM SIGGRAPH 2005 Papers*, ser. SIGGRAPH '05. New York, NY, USA: ACM, 2005, pp. 861–868.

[37] W. Fan, K. Wang, F. Cayre, and Z. Xiong, "Median Filtered Image Quality Enhancement and Anti-Forensics via Variational Deconvolution," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 10, no. 5, pp. 1076–1091, 2015.

[38] G. Valenzise, M. Tagliasacchi, and S. Tubaro, "Revealing the Traces of JPEG Compression Anti-Forensics," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 8, no. 2, pp. 335–349, 2013.

[39] M. Stamm, S. Tjoa, and W. L. K. Liu, "Undetectable image tampering through JPEG compression anti-forensics," in *Intl. Conference on Image Processing*, Sept 2010, pp. 2109–2112.

[40] B. Peng, W. Wang, J. Dong, and T. Tan, "Improved 3D lighting environment estimation for image forgery detection," in *IEEE Intl. Workshop on Information Forensics and Security (WIFS)*, Nov 2015, pp. 1–6.

[41] M. Johnson and H. Farid, "Exposing Digital Forgeries in Complex Lighting Environments," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 2, no. 3, pp. 450–461, Sept 2007.

[42] T. Carvalho, F. Faria, H. Pedrini, R. Torres, and A. Rocha, "Illuminant-Based Transformed Spaces for Image Forensics," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 11, no. 4, pp. 720–733, April 2016.

[43] F. Chunhui, X. Zhengquan, and Z. Xinghui, "An Anti-Forensic Algorithm of JPEG Double Compression Based Forgery Detection," in *International Symposium on Information Science and Engineering (ISISE)*, Dec 2012, pp. 159–164.

[44] T. Bianchi and A. Piva, "Reverse engineering of double JPEG compression in the presence of image resizing," in *IEEE Intl. Workshop on Information Forensics and Security (WIFS)*, Dec 2012, pp. 127–132.

[45] A. Mire, S. Dhok, P. Porey, and N. Mistry, "Digital Forensic of JPEG Images," in *Intl. Conference on Signal and Image Processing (ICSIP)*, Jan 2014, pp. 131–136.

[46] F. Peng and X.-l. Wang, "Digital Image Forgery Forensics by Using Blur Estimation and Abnormal Hue Detection," in *Symposium on Photonics and Optoelectronic (SOPO)*, June 2010, pp. 1–4.

[47] D.-Y. Hsiao and S.-C. Pei, "Detecting digital tampering by blur estimation," in *International Workshop on Systematic Approaches to Digital Forensic Engineering*, Nov 2005, pp. 264–278.

[48] O. Bulan, J. Mao, and G. Sharma, "Geometric distortion signatures for printer identification," in *Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 1401–1404.

[49] Y. Wu, X. Kong, X. You, and Y. Guo, "Printer forensics based on page document's geometric distortion," in *Intl. Conference on Image Processing (ICIP)*, Nov 2009, pp. 2909–2912.

[50] H.-Y. Lee and J.-H. Choi, "Identifying color laser printer using noisy feature and support vector machine," in *Intl. Conference on Ubiquitous Information Technologies and Applications*, 2010, pp. 1–6.

[51] A. Lawgaly, F. Khelifi, and A. Bouridane, "Weighted averaging-based sensor pattern noise estimation for source camera identification," in *IEEE Intl. Conference on Image Processing (ICIP)*, Oct 2014, pp. 5357–5361.

[52] A. Kharboutly, W. Puech, G. Subsol, and D. Hoa, "CT-Scanner identification based on sensor noise analysis," in *European Workshop on Visual Information Processing (EUVIP)*, Dec 2014, pp. 1–5.

[53] A. Mikkilineni, P.-j. Chiang, G. Ali, G. Chiu, J. Allebach, and E. Delp, "Printer identification based on textural features," in *Intl. Conference on Digital Printing Technologies*, 2004, pp. 306–311.

[54] Y. Hu, C.-T. Li, X. Lin, and B.-b. Liu, "An Improved Algorithm for Camera Model Identification Using Inter-channel Demosaicking Traces," in *Intl. Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, July 2012, pp. 325–330.

[55] P.-J. Chiang, N. Khanna, A. Mikkilineni, M. Segovia, S. Suh, J. Allebach, G.-C. Chiu, and E. Delp, "Printer and scanner forensics," *Signal Processing Magazine*, vol. 26, no. 2, pp. 72–83, 2009.

[56] R. W. Floyd and L. Steinberg, "An Adaptive Algorithm for Spatial Greyscale," *Society for Information Display*, vol. 17, no. 2, pp. 75–77, 1976.

[57] R. Ulichney, *Digital Halftoning*. MIT Press, 1987.

[58] N. Khanna, A. K. Mikkilineni, G. T. Chiu, J. P. Allebach, and E. J. Delp, "Survey of Scanner and Printer Forensics at Purdue University," in *Intl. Workshop on Computational Forensics*, 2008, pp. 22–34.

[59] G. N. Ali, P.-J. Chiang, A. K. Mikkilineni, J. P. Allebach, G. T.-C. Chiu, and E. J. Delp, "Intrinsic and Extrinsic Signatures for Information Hiding and Secure Printing with Electrophotographic Devices," in *Intl. Conference on Digital Printing Technologies*, 2003, pp. 511–515.

[60] J. E. Girard, *Criminalistics: Forensic Science, Crime and Terrorism*, 3rd ed. Jones and Bartlett Publishers, 2013.

[61] A. Braz, M. Lopez-Lopez, and C. Garcia-Ruiz, "Raman spectroscopy for forensic analysis of inks in questioned documents," *Forensic Science International*, vol. 232, no. 1-3, pp. 206–212, 2013.

[62] L. Gal, M. Belovicova, M. Oravec, M. Palkova, and M. Ceppan, "Analysis of Laser and Inkjet Prints Using Spectroscopic Methods for Forensic Identification of Questioned Documents," in *Symposium on Graphic Arts*, vol. 10, 1993, pp. 1–8.

[63] P.-C. Chu, B. Y. Cai, Y. K. Tsoi, R. Yuen, K. S. Leung, and N.-H. Cheung, "Forensic Analysis of Laser Printed Ink by X-ray Fluorescence and Laser-Excited Plume Fluorescence," *Analytical Chemistry*, vol. 85, no. 9, pp. 4311–4315, 2013.

[64] M. Gaubatz and S. Simske, "Printer-scanner identification via analysis of structured security deterrents," in *Intl. Workshop on Information Forensics and Security (WIFS)*, 2009, pp. 151–155.

[65] S. J. Simske, J. S. Aronoff, M. Sturgill, and J. C. Villa, "Spectral Pre-Compensation and Security Print Deterrent Authentication," *NIP and Digital Fabrication Conference*, vol. 2008, no. 2, pp. 792–795, 2008.

[66] G. N. Ali, P.-j. Chiang, A. K. Mikkilineni, G. T. Chiu, E. J. Delp, and J. P. Allebach, "Application of principal components analysis and gaussian mixture models to printer identification," in *Intl. Conference on Digital Printing Technologies*, 2004, pp. 301–305.

[67] M.-J. Tsai, J.-S. Yin, I. Yuadi, and J. Liu, "Digital forensics of printed source identification for Chinese characters," *Multimedia Tools and Applications*, pp. 1–27, 2013.

[68] A. K. Mikkilineni, P.-j. Chiang, G. N. Ali, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "Printer identification based on graylevel co-occurrence features for security and forensic applications," in *Intl. Conference on Security, Steganography, and Watermarking of Multimedia Contents*, 2005, pp. 430–440.

[69] A. K. Mikkilineni, O. Arslan, P.-j. Chiang, R. M. Kumontoy, J. P. Allebach, and G. T. c, "Printer forensics using svm techniques," in *Intl. Conference on Digital Printing Technologies*, 2005, pp. 223–226.

[70] A. K. Mikkilineni, N. Khanna, and E. J. Delp, "Forensic printer detection using intrinsic signatures," in *Intl. Society for Optics and Photonics (SPIE)*, vol. 7880, 2011, pp. 78 800R–78 800R–11.

[71] J.-H. Choi, H.-K. Lee, H.-Y. Lee, and Y.-H. Suh, "Color Laser Printer Forensics with Noise Texture Analysis," in *ACM Workshop on Multimedia and Security*, 2010, pp. 19–24.

[72] S. Elkasrawi and F. Shafait, "Printer Identification Using Supervised Learning for Document Forgery Detection," in *Intl. Workshop on Document Analysis Systems*, April 2014, pp. 146–150.

[73] M. U. Devi, C. R. Rao, and A. Agarwal, "A Survey of Image Processing Techniques for Identification of Printing Technology in Document Forensic Perspective," *International Journal of Computer Applications (IJCA)*, vol. 1, pp. 9–15, 2010.

[74] N. Khanna, A. K. Mikkilineni, A. F. Martone, G. N. Ali, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "A survey of forensic characterization methods for physical devices," *Digital Investigation*, vol. 3, no. 0, pp. 17–28, 2006.

[75] S.-J. Ryu, H.-Y. Lee, D.-H. Im, J.-H. Choi, and H.-K. Lee, "Electrophotographic printer identification by halftone texture analysis," in *IEEE Intl. Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 1846–1849.

[76] J.-H. Choi, D.-H. Im, H.-Y. Lee, J.-T. Oh, J.-H. Ryu, and H.-K. Lee, "Color laser printer identification by analyzing statistical features on discrete wavelet transform," in *Intl. Conference on Image Processing (ICIP)*, 2009, pp. 1505–1508.

[77] M.-J. Tsai, J. Liu, C.-S. Wang, and C.-H. Chuang, "Source color laser printer identification using discrete wavelet transform and feature selection algorithms," in *Intl. Symposium on Circuits and Systems (ISCAS)*, 2011, pp. 2633–2636.

[78] N. Khanna, G. T. C. Chiu, J. P. Allebach, and E. J. Delp, "Scanner identification with extension to forgery detection," in *Proceedings on Security, Forensics, Steganography, and Watermarking of Multimedia Contents (SPIE)*, vol. 6819, 2008, pp. 68 190G–68 190G–10.

[79] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan 1979.

[80] D.-G. Kim and H.-K. Lee, "Color laser printer identification using photographed halftone images," in *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*, Sept 2014, pp. 795–799.

[81] ——, "Colour laser printer identification using halftone texture fingerprint," *Electronics Letters*, vol. 51, no. 13, pp. 981–983, 2015.

[82] H. Wu, X. Kong, and S. Shang, "A printer forensics method using halftone dot arrangement model," in *Intl. Conference on Signal and Information Processing (ChinaSIP)*, July 2015, pp. 861–865.

[83] P. Micka, "Letter frequency (English)," http://en.algoritmy.net/article/40379/Letter-frequency-English.

[84] W. Jiang, A. T. S. Ho, H. Treharne, and Y. Q. Shi, "A Novel Multi-size Block Benford's Law Scheme for Printer Identification," in *Pacific Rim Conference on Advances in Multimedia Information Processing*, ser. PCM'10, 2010, pp. 643–652.

[85] E. Kee and H. Farid, "Printer Profiling for Forensics and Ballistics," in *ACM Workshop on Multimedia and Security*, 2008, pp. 3–10.

[86] R. Duda and P. Hart., *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[87] M. Schreyer, "Intelligent Printing Technique Recognition and Photocopy Detection for Forensic Document Examination." in *Informatiktage*, L. Porada, Ed., vol. S-8, 2009, pp. 39–42.

[88] W. Mazzella and R. Marquis, "Forensic Image Analysis of Laser-Printed Documents," *Journal of the American Society of Questioned Document Examiners*, vol. 10, no. 1, 2007.

[89] M.-J. Tsai and J. Liu, "Digital forensics for printed source identification," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2013, pp. 2347–2350.

[90] M.-J. Tsai, C.-L. Hsu, J.-S. Yin, and I. Yuadi, "Japanese character based printed source identification," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 2800–2803.

[91] E. Arias-Castro and D. Donoho, "Does median filtering truly preserve edges better than linear filtering?" *The Annals of Statistics*, vol. 37, no. 3, pp. 1172–1206, 06 2009.

[92] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1992.

[93] E. Kee and H. Farid, "A perceptual metric for photo retouching," *Proceedings of the National Academy of Sciences*, vol. 108, no. 50, pp. 19 907–19 912, 2011.

[94] M. Johnson and H. Farid, "Exposing digital forgeries through specular highlights on the eye," in *ACM Intl. Conference on Information Hiding (IH)*, 2007, pp. 311–325.

[95] P. Saboia, T. Carvalho, and A. Rocha, "Eye Specular Highlights telltales for digital forensics: a machine learning approach," in *Intl. Conference on Image Processing (ICIP)*, 2011, pp. 1977–1980.

[96] A. Popescu and H. Farid, "Statistical tools for digital forensics," in *ACM Intl. Workshop on Information Hiding (IH)*, 2004, pp. 128–147.

[97] M. Kirchner and R. Bohme, "Hiding Traces of Resampling in Digital Images," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 3, no. 4, pp. 582–592, 2008.

[98] A. Bovik, "Streaking in median filtered images," *IEEE Transactions on Acoustic Speech and Signal Processing*, vol. 35, no. 4, pp. 493–503, 1987.

[99] M. Kirchner and J. Fridrich, "On detection of median filtering in digital images," in *SPIE Media Forensics and Security II*, 2010, pp. 754 110–754 110–12.

[100] G. Cao, Y. Zhao, R. Ni, L. Yu, and H. Tian, "Forensic detection of median filtering in digital images," in *IEEE Intl. Conference on Multimedia and Expo (ICME)*, 2010, pp. 89–94.

[101] H.-D. Yuan, "Blind Forensics of Median Filtering in Digital Images," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 6, no. 4, pp. 1335–1345, 2011.

[102] C. Chen and J. Ni, "Median filtering detection using edge based prediction matrix," in *Intl. Conference on Digital-Forensics and Watermarking*, 2012, pp. 361–375.

[103] C. Chen, J. Ni, R. Huang, and J. Huang, "Blind median filtering detection using statistics in difference domain," in *ACM Intl. Conference on Information Hiding (IH)*, 2013, pp. 1–15.

[104] C. Chen, J. Ni, and J. Huang, "Blind Detection of Median Filtering in Digital Images: A Difference Domain Based Approach," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4699–4710, to appear in December, 2013.

[105] X. Kang, M. Stamm, A. Peng, and K. J. R. Liu, "Robust Median Filtering Forensics Using an Autoregressive Model," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 8, no. 9, pp. 1456–1468, 2013.

[106] ——, "Robust median filtering forensics based on the autoregressive model of median filtered residual," in *IEEE Signal Information Processing Association Annual Summit and Conference (APSIPAASC)*, 2012, pp. 1–9.

[107] Y. Zhang, S. Li, S. Wang, and Y. Q. Shi, "Revealing the Traces of Median Filtering Using High-Order Local Ternary Patterns," *IEEE Signal Processing Letters*, vol. 21, no. 3, pp. 275–279, March 2014.

[108] J. Fridrich, D. Soukal, and J. Lukas, "Detection of Copy-Move Forgery in Digital Images," in *Digital Forensic Research Workshop (DFRWS)*, Cleveland, USA, 2003, pp. 134–137.

[109] A. C. Popescu and H. Farid, "Exposing Digital forgeries by Detecting Duplicated Image Regions," Dept. of Computer Science – Dartmouth College, Hanover, USA, Tech. Rep. TR 2004-515, 2004.

[110] W. Luo, J. Huang, and G. Qiu, "Robust Detection of Region-Duplication Forgery in Digital Image," in *Intl. Conference on Pattern Recognition (ICPR)*, 2006, pp. 746–749.

[111] B. Mahdian and S. Saic, "Detection of Copy-Move Forgery Using a Method Based on Blur Moment Invariants," *Forensic Science Intl.*, pp. 180–189, September 2006.

[112] J. Zhang, Z. Feng, and Y. Su, "A New Approach for Detecting Copy-Move Forgeries in Digital Images," in *IEEE Intl. Conference on Communication Systems (ICCS)*, 2010, pp. 362–366.

[113] G. Li, Q. Wu, D. Tu, and S. Sun, "A sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on DWT and SVD," in *IEEE Intl. Conference on Multimedia and Expo. (ICME)*, 2007, pp. 1750–1753.

[114] X. B. Kang and S. M. Wei, "Identifying Tampered Regions Using Singular Value Decomposition in Digital Image Forensics," in *Intl. Conference on Computer Science and Software Engineering*, vol. 3, Dec 2008, pp. 926–930.

[115] S.-J. Ryu, M.-J. Lee, and H.-K. Lee, "Detection of copy-rotate-move forgery using Zernike moments," in *Intl. Workshop in Information Hiding (IHW)*, 2010, pp. 51–65.

[116] S.-J. Ryu, M. Kirchner, M.-J. Lee, and H.-K. Lee, "Rotation Invariant Localization of Duplicated Image Regions Based on Zernike Moments," *Transactions on Information Forensics and Security (T.IFS)*, vol. 8, no. 8, pp. 1355–1370, 2013.

[117] S. Bravo-Solorio and A. K. Nandi, "Exposing duplicated regions affected by reflection, rotation and scaling," in *Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 1880–1883.

[118] M. Bashar, K. Noda, N. Ohnishi, and K. Mori, "Exploring Duplicated Regions in Natural Images," *IEEE Transactions on Image Processing (T.IP)*, 2010.

[119] S. Bayram, H. T. Sencar, and N. Memon, "An Efficient and Robust Method for Detecting Copy-Move Forgery," in *IEEE Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 1053–1056.

[120] J. Wang, G. Liu, H. Li, Y. Dai, and Z. Wang, "Detection of Image Region Duplication Forgery Using Model with Circle Block," in *IEEE Intl. Conference on Multimedia Information Networking and Security (MINES)*, 2009, pp. 25–29.

[121] G. Bradski and A. Kaehler, *Learning OpenCV*, 1st ed. O'Reilly Media, 2008.

[122] J.-W. Wang, G.-J. Liu, Z. Zhang, Y.-W. Dai, and Z.-Q. Wang, "Fast and Robust Forensics for Image Region-duplication Forgery," *Acta Automatica Sinica*, vol. 35, pp. 1488–1495, 2010.

[123] H.-J. Lin, C.-W. Wang, and Y.-T. Kao, "Fast Copy-Move Forgery Detection," *WSEAS Transactions on Signal Processing (WSEAS-TSP)*, pp. 188–197, May 2009.

[124] E. Ardizzone and G. Mazzola, "Detection of Duplicated Regions in Tampered Digital Images by Bit-Plane Analysis," in *Intl. Conference on Image Analysis and Processing (ICIAP)*, 2009, pp. 893–901.

[125] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "The Generalized PatchMatch Correspondence Algorithm," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 29–43.

[126] ——, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing," *ACM Transactions on Graphics (ToG)*, pp. 24:1–24:11, July 2009.

[127] H. Huang, W. Guo, and Y. Zhang, "Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm," in *IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA)*. IEEE Computer Society, 2008, pp. 272–276.

[128] X. Pan and S. Lyu, "Detecting Image Region Duplication Using SIFT Features," in *IEEE Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 1706–1709.

[129] ——, "Region Duplication Detection Using Image Feature Matching," *Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 857–867, Dec 2010.

[130] I. Amerini, L. Ballan, R. Caldelli, A. D. Bimbo, and G. Serra, "A sift-based forensic method for copy-move attack detection and transformation recovery," *Transactions on Information Forensics and Security (T.IFS)*, vol. 6, no. 3, pp. 1099–1110, 2011.

[131] B. Xu, J. Wang, G. Liu, H. Li, and Y. Dai, "Image Copy-Move Forgery Detection Based on SURF," in *IEEE Intl. Conference on Multimedia Information Networking and Security (MINES)*, 2010, pp. 889–892.

[132] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence (T.PAMI)*, vol. 27, no. 10, pp. 1615–1630, 2005.

[133] L. Xu, A. Krzyzak, and C. Suen, "Methods of combining multiple classifiers and their applications to hand-writing recognition." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 22, pp. 418–435, 1992.

[134] Y. Huang and C. Suen, "A method of combining multiple experts for the recognition of unconstrained handwritten numerals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 1, pp. 90–94, January 1995.

[135] F. R. de Siqueira, W. R. Schwartz, and H. Pedrini, "Multi-scale gray level co-occurrence matrices for texture description," *Neurocomputing*, vol. 120, no. 0, pp. 336–345, 2013, image Feature Detection and Description.

[136] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[137] T. Ojala, M. Pietikäinen, and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Feature Distributions," *Pattern Recognition*, vol. 29, pp. 51–59, 1996.

[138] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, pp. 1895–1923, 1998.

[139] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Intl. Conference on Computer Vision & Pattern Recognition*, C. Schmid, S. Soatto, and C. Tomasi, Eds., vol. 2, June 2005, pp. 886–893.

[140] P.-J. Chiang, N. Khanna, A. K. Mikkilineni, M. V. O. Segovia, S. Suh, J. P. Allebach, G. T. C. Chiu, and E. J. Delp, "Printer and scanner forensics," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 72–83, March 2009.

[141] N. Khanna, A. K. Mikkilineni, and E. J. Delp, "Scanner Identification Using Feature-Based Processing and Analysis," *IEEE Transactions on Information Forensics and Security (T.IFS)*, vol. 4, no. 1, pp. 123–139, March 2009.

[142] M. Chen, J. Fridrich, M. Goljan, and J. Lukas, "Determining Image Origin and Integrity Using Sensor Noise," *IEEE Transactions on Information Forensics and Security (T.IFS)*, vol. 3, no. 1, pp. 74–90, March 2008.

[143] J. Lukas, J. Fridrich, and M. Goljan, "Digital Camera Identification from Sensor Noise Sensor," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 1, no. 2, pp. 205–214, 2006.

[144] F. d. O. Costa, E. A. Silva, M. Eckmann, W. J. Scheirer, and A. Rocha, "Open Set Source Camera Attribution and Device Linking," *Elsevier Pattern Recognition Letters (PRL)*, vol. 39, pp. 92–101, 2014.

[145] P. Korzeniowski, "Color Laser Printers Gaining Enterprise Popularity," http://www.technewsworld.com/story/43766.html, June 2005.

[146] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

[147] M. Thomsen, "Microsoft's Deep Learning Project Outperforms Humans In Image Recognition," http://www.forbes.com/sites/michaelthomsen/2015/02/19/microsofts-deep-learning-project-outperforms-humans-in-image-recognition/#38bf25831285, February 2015.

[148] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." in *Neural Information Processing Systems (NIPS).*, 2012, pp. 1106–1114.

[149] J. Chen, X. Kang, Y. Liu, and Z. Wang, "Median Filtering Forensics Based on Convolutional Neural Networks," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1849–1853, Nov 2015.

[150] VLFEAT, "MatConvNet: CNNs for MATLAB," http://www.vlfeat.org/matconvnet/.

[151] A. Rocha and S. K. Goldenstein, "Multiclass From Binary: Expanding One-Versus-All, One-Versus-One and ECOC-Based Approaches," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 289–302, Feb 2014.

[152] A. Ferreira, L. C. Navarro, G. Pinheiro, J. A. dos Santos, and A. Rocha, "Laser printer attribution: Exploring new features and beyond," *Forensic Science International*, vol. 247, pp. 105–125, 2015.

[153] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," in *CVPR 2015*, 2015. [Online]. Available: http://arxiv.org/abs/1409.4842

[154] C. G. M. Snoek, "Early versus late fusion in semantic video analysis," in *ACM Intl. Conference on Multimedia*, 2005, pp. 399–402.

[155] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964.

[156] wikipedia, "Letter Frequency," https://en.wikipedia.org/wiki/Letter_frequency.

[157] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein, "Vision of the unseen: Current trends and challenges in digital image and video forensics," *ACM Computing Surveys*, vol. 43, no. 4, pp. 26:1–26:42, 2011.

[158] E. Hauck, "Data compression using run length encoding and statistical encoding," Dec. 2 1986, uS Patent 4,626,829.

[159] A. Rocha and S. Goldenstein, "Progressive randomization: Seeing the unseen," *Elsevier Computer Vision and Image Understanding (CVIU)*, vol. 114, no. 3, pp. 349–362, 2010.

[160] K.-H. Thung and P. Raveendran, "A survey of image quality measures," in *IEEE Intl. Conference for Technical Postgraduates (TECHPOS)*, 2009, pp. 1–4.

[161] A. Eskicioglu and P. Fisher, "Image quality measures and their performance," *IEEE Transactions on Communications*, vol. 43, no. 12, pp. 2959–2965, 1995.

[162] Z. Wang, A. Bovik, and H. Sheikh, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[163] I. Avcibas, S. Bayram, N. Memon, M. Ramkumar, and B. Sankur, "A classifier design for detecting image manipulations," in *IEEE Intl. Conference on Image Processing (ICIP)*, 2004, pp. 2645–2648.

[164] I. Avcibas, N. Memon, and B. Sankur, "Steganalysis based on image quality metrics," in *IEEE Workshop on Multimedia and Signal Processing*, 2001, pp. 517–522.

[165] G. Schaefer and M. Stich, "UCID - An Uncompressed Colour Image Database," in *Storage and Retrieval Methods and Applications for Multimedia*, ser. Proceedings of SPIE, vol. 5307, 2004, pp. 472–480.

[166] T. Pevny, "SPAM features," http://dde.binghamton.edu/download/spam/, December 2011.

[167] I. D. Gebru, "Median Filter Detection for Digital image forensics," https://isrishblog.wordpress.com/2012/08/07/median-filter-detection-for-digital-image-forensics/, August 2012.

[168] M. Fontani and M. Barni, "Hiding Traces of Median Filtering in Digital Images," in *European Signal Processing Conference (EUSIPCO)*, 2012, pp. 1239–1243.

[169] D. Lowe, "Object recognition from local scale-invariant features," in *Intl. Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157.

[170] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *European Conference on Computer Vision (ECCV)*, 2006, pp. 404–417.

[171] L. Breiman and L. Breiman, "Bagging Predictors," in *Machine Learning*, 1996, pp. 123–140.

[172] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," http://alex.smola.org/papers/2003/SmoSch03b.pdf, accessed 07 June 2015.

[173] "IEEE IFS-TC Image Forensics Challenge," http://ifc.recod.ic.unicamp.br/fc.website/index.py.

[174] M. Friedmann, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, pp. 675–701, 1939.

[175] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 6, no. 1, pp. 80–83, December 1945.

[176] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency." *Annals of Statistics*, no. 29, 2001.

[177] O. J. Dunn, "Estimation of the Medians for Dependent Variables." *Annals of Mathematical Statistics*, vol. 1, no. 30, pp. 192–197, 1959.

[178] S. Holm, "A simple sequentially rejective multiple test procedure." *Scandinavian Journal of Statistics*, no. 6, pp. 65–70, 1979.

[179] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?" *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3133–3181, Jan. 2014.

[180] A. Ferreira, P. Bestagini, L. Bondi, L. Baroffio, J. A. dos Santos, S. Tubaro, and A. Rocha, "Data-Driven Feature Characterization Techniques for Laser Printer Attribution," *IEEE Transactions on Image Forensic and Security (TIFS)*, Submitted.

[181] A. Ferreira and A. Rocha, "A Multiscale and Multi-Perturbation Blind Forensic Technique for Median Detecting," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, 2014, vol. 8827, pp. 302–310.

[182] A. Ferreira, J. A. dos Santos, and A. Rocha, "Multi-directional and Multi-scale Perturbation Approaches for Blind Forensic Median Filtering Detection," *Journal of Intelligent Data Analysis*, To appear in 2016.

[183] A. Ferreira, S. C. Felipussi, C. Alfaro, P. Fonseca, J. E. Vargas, J. A. dos Santos, and A. Rocha, "Behavior Knowledge Space-Based Fusion for Copy-Move Forgery Detection," *IEEE Transactions on Image Processing (TIP)*, Submitted.