U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

# A machine learning based digital forensics application to detect tampered multimedia files

Sara Cardoso Ferreira

Mestrado Integrado em Engenharia de Redes e Sistemas Informáticos

Departamento de Ciência de Computadores

2021

**Orientador**
Manuel Eduardo Carvalho Duarte Correia, Professor Associado, Faculdade de Ciências da Universidade do Porto

**Coorientador**
Mário João Gonçalves Antunes, Professor Adjunto, Escola Superior de Tecnologia e Gestão do Politécnico de Leiria

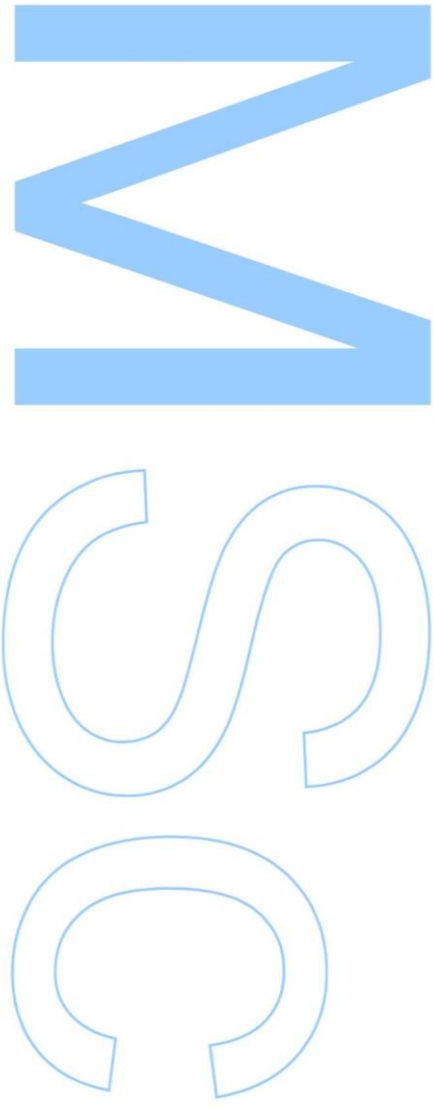
U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_____



# Acknowledgments

I would like to thank my supervisors Professor Mário Antunes and Professor Manuel Correia because without them none of this was possible. Thank you professor Mário Antunes for accepting to go on this adventure with me, for encouraging me to go one step further and for all your availability and enthusiasm for the subject. I would also like to thank Professor Mark McKinnon from Davenport University for his support in clarifying some concepts regarding the development of Autopsy modules. I also want to thank my mother, who, despite not understanding anything about computers, always supported me. Finally, I would like to thank all my friends who have always supported me in all phases of my life.

# Abstract

Computer Forensics, commonly known as Digital Forensics, embodies techniques, tools, and procedures for the collection, preservation and analysis of digital evidence in electronic equipment, including disks, smartphones and other devices with storage capacity. The artifacts collected and with interest for the criminal investigation are diverse as depend on the crime being investigated. Besides the common artifacts, there are several types of digital content that may have potential interest to the investigator, such as files, e-mail addresses, phone contacts or other information that may indicate the existence of criminal activity.

Multimedia content (photos and videos) have a high interest to digital forensics, mainly due to its association with crimes that have a high impact on public opinion. Fake news, misinformation, sexual abuse, and distribution of pornographic content, especially involving under aged, are some examples where seized videos and photos are relevant to the investigation and must necessarily be analyzed by criminal investigation teams. This analysis, usually employs manual and time-consuming techniques.

There are several tools and techniques used to process and analyze photos and videos, with numerous applications in facial recognition and emotion detection. Although advances can also be identified in the detection of manipulated multimedia content, this has not been translated into improvements for forensic analysis and for the investigation of cybercrimes. Autopsy is a fast, easy-to-use, and open source digital forensic platform, which is capable of analyzing a wide range of electronic devices. Besides the built-in modules, it also benefits from the developments made by the community, mainly through the development of custom modules.

This dissertation describes the development of a Support Vector Machines (SVM)-based standalone application and two modules for Autopsy, to automate photos and videos files processing, and to compute their probability of have been digitally manipulated. A dataset comprising both real and manipulated files containing objects and faces, was also created to train and test the model. Using a ten-fold cross-validation it was achieved an F1-score above 99.5% for tampered photos detection, 79.8% for videos, and 89.2% for a mixture of manipulated photos and videos. The method used was compared with a Convolutional Neural Network (CNN)-based method, to benchmark both models. Although investigators' manual analysis is still needed, the developed modules indicate the examples that should deserve the investigator's attention, that is those that have a higher probability of have been manipulated.

# Resumo

A Informática Forense, vulgarmente designada por Análise Digital Forense, emprega técnicas, aplicações e procedimentos para a recolha, preservação e análise de evidências digitais em equipamentos electrónicos, nomeadamente discos, *smartphones* e outros dispositivos com capacidade de armazenamento. Os artefactos recolhidos e de interesse para a investigação criminal são diversos, dependendo do crime em causa. Para além dos artefactos comuns, há outros conteúdos digitais que podem ter potencial interesse para o investigador, como ficheiros, endereços de e-mail, contactos e outra informação que pode indicar a existência de aividade criminal.

Os conteúdos multimédia (fotos e vídeos) têm um interesse acrescido pela sua associação a crimes com um elevado impacto na opinião pública. As noticias falsas, o abuso sexual e a distribuição de conteúdo pornográfico, especialmente envolvendo menores, são alguns exemplos onde as fotos e videos apreendidos são relevantes para a investigação e têm de ser analisados pelas equipas de investigação criminal. A análise deste tipo de conteúdos recorre normalmente a técnicas manuais e pouco sistematizadas.

Existem várias aplicações e técnicas para processar e analisar fotos e vídeos, com inúmeras aplicações no reconhecimento facial e na deteção de emoções. Embora se possam identificar avanços também na deteção de conteúdos multimédia manipulados, tal não se tem traduzido em melhorias para a informática forense e para a investigação de crimes informáticos. Autopsy é uma plataforma de análise digital forense *open-source*, rápida, fácil de usar e capaz de analisar uma vasta gama de dispositivos. Além dos módulos nativos, o Autopsy beneficia também do desenvolvimento feito pela comunidade, nomeadamente pela adição de módulos personalizados.

Esta dissertação descreve uma aplicação baseada em Support Vector Machines (SVM) e de dois módulos para o Autopsy, para automatizar o processamento de fotos e vídeos, e calcular a probabilidade de terem sido manipulados digitalmente. Foi também criado um *dataset* contendo fotos e vídeos reais e manipulados, com objectos e faces, para treinar e testar o modelo. Os testes realizados com *ten-fold cross validation*, revelaram um *F1* acima de $99,5\%$ na deteção de fotos manipuladas, $79,8\%$ na deteção de vídeos, e $89,2\%$ na deteção de uma mistura de fotos e vídeos. O método utilizado foi comparado com um método baseado em *Convolutional Neural Network* (CNN), para avaliar ambos os modelos. Embora a análise manual ainda seja necessária, os módulos desenvolvidos indicam os ficheiros que devem merecer maior atenção pelo investigador, ou seja, os que têm maior probabilidade de terem sido manipulados.

# Contents

# List of Tables

# List of Figures

# Listings

# Acronyms

**OPC** Orgãos de policia criminal

**JVM** Java virtual machine

**DOG** Difference of gaussian

**ORB** Oriented Fast and Rotated Brief

**SVM** Support Vector Machines

**CNCS** Centro Nacional de Cibersegurança

**RASI** Relatório Anual de Segurança Interna

**UNC3T** Unidade Nacional de Combate ao Cibercrime e à Criminalidade Tecnológica

**ML** Machine Learning

**DFT** Discrete Fourier Transform

**ROC** Receiver Operating Characteristics

**ML** Machine Learning

**PJ** Polícia Judiciária

**CNN** Convolution Neural Networks

**AI** Artificial Intelligence

**FTK** Forensic Toolkit

**DAM** Digital assets management system

**GAN** Generative Adversarial Networks

**GNR** Guarda Nacional Republicana

**PSP** Polícia de Seguranca Pública

**GUI** Graphical User Interface

**AI** Artifical Intelligence

**CRF** Camera Response Function

**CFA** Color Filter Array

**FAST** Features from Accelerated Segment Test

**BRIEF** Binary Robust Independent Elementary Features

**SIFT** Scale Invariant Feature Transform

**SURF** Speeded-Up Robust Features

**RBF** Radial Basis Function

**TP** True Positive

**FP** False Positive

**FN** False Negative

**TN** True Negative

**OSDF** Open Source Digital Forensics

**KML** Keyhole Markup Language

**ROC** Receiver Operating Characteristic

**RNN** Recurrent Neural Networks

# Chapter 1

# Introduction

Cybercrime has challenged national security systems all over the world. In the last five years, there has been an increase of 67% in the incidence of security breaches worldwide [6], with malicious activities like phishing, ransomware, and cryptojacking being the most popular threats to cybersecurity [4, 49, 69]. In Portugal, there was an increase of 88% in security incidents with vulnerabilities [29]. This growth is directly linked to the Covid-19 pandemic we are going through, which has had an undeniable influence on the increasing of cybercrimes. The curfew and the acceleration in the migration to online encouraged cybercriminals' sense of opportunity, leading to the exploitation of the most exposed technical and human vulnerabilities. Activities we took for granted in our daily lives like work, read, talk, study, and shop, are now highly dependent on digital services. This creates a perfect context for an increase in online fraud and other criminal activities in cyberspace [9, 51]. The widespread global reach of cyberattacks, their impact, sophistication, and dire consequences for society can be analyzed within several distinct dimensions, namely economic disruption, psychological disorder, and other threats to national defense [10, 12].

However, not every illicit act made through a computer is necessarily a cybercrime act. The term cybercrime emerged in the late 1990s in a political and intra-governmental group called the "Lyon Group", which aimed to analyze crimes committed using electronic equipment or through the Internet. Later, the term started being used to identify, in a general way, all types of crime committed on communication networks, namely the Internet. These crimes also include computer crimes that are committed against computer systems plus the data and information housed in the information systems [11].

The CERT.PT (https://www.cncs.gov.pt/certpt/, accessed on 7 March 2021) is a service within the Centro Nacional de Cibersegurança (CNCS) that coordinates incident response involving state entities, operators of essential services, operators of national critical infrastructures and digital service providers. In general, CERT.PT coordinates the response to incidents into national cyberspace, including any device belonging to a network or IP addressing block assigned to an electronic communications operator, institution, legal or natural person headquartered or physically located in Portuguese territory.

In 2020, CERT.PT received and processed 6525 notifications, which corresponds to an increasing of 93%, when comparing with the year of 2019. From the amount of notifications, about 22% resulted in the opening of incidents, that were analyzed and successfully resolved. According to the Relatório Anual de Segurança Interna (RASI) 2020 [8], 31% of the ground total of incidents affected public administration entities, which means that there was an increasing in this area over the previous year [7].

From the incidents reviewed in RASI 2020, four classes can be highlighted: fraud, malicious code, intrusion, and information security, as depicted in Figure 1.

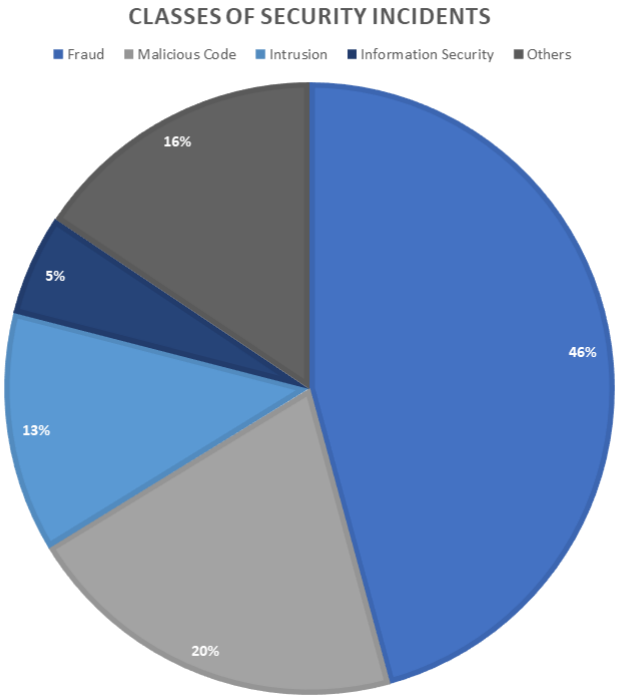



Figure 1.1: Security incidents in Portugal in 2020, according to RASI.

The fraud concerns essentially to phishing, smishing (SMS phishing) and spearphishing attacks. Its primary goal is to collect information about the victim and to induce him or her to perform an unwanted action. The financial and banking sectors have been more prone to this type of malicious activity, where a higher incidence was observed. The second type that has recorded more incidents is malicious code, which includes mainly the distribution of malware. The pandemic situation has also favored some intrusion actions, taking advantage of technical vulnerabilities and the circumstances of having the vast majority of people in remote work [29]. Ransomware malicious code is however included in information security category. Many other cybercrimes, such as illegitimate use of a third party's name, were carried out and every day new cases are being analyzed. It is of utmost importance to find tools that can somehow mitigate this inevitable increasing of cybercrime.

Digital forensics is based on the collection and analysis of artifacts that may constitute

digital evidence in crimes, and it becomes an indispensable tool for criminal investigation police. Digital forensics analysis embodies techniques and procedures for the collection, preservation, and analysis of digital evidences that may exist in electronic equipment [75]. Digital forensics techniques are essential to investigate crimes that are committed with computers (e.g.phishing and bank fraud), as well as those carried on against individuals, where evidence may reside on a computer (e.g.money laundering and child exploitation) [35]. Therefore, it is a crucial tool for law enforcement, namely for their criminal investigation teams. There are several criminal police agencies in Portugal, being the most significant, the Polícia Judiciária (PJ). In PJ, the Unidade Nacional de Combate ao Cibercrime e à Criminalidade Tecnológica (UNC3T) leads the cybercrime investigation and digital forensics of seized electronic equipment.

When the analysis is conducted manually, solely by the means of a human operator, digital forensics can be very time-consuming and highly inefficient in terms of identifying and collecting complete and meaningful digital evidence [21] of cybercrimes or crimes committed with the resource to etectronic equipment. Moreover, the manual analysis of multimedia content, namely for the identification of manipulated videos or photos, often results in the misclassification of files.

Effective forensics tools are essential, as they have the ability to reconstruct evidence left by cybercriminals when they perpetrate a cyberattack [42]. However, there exists an increasing number of highly sophisticated tools that make life much easier for cybercriminals to carry out complex and highly effective digital attacks. The criminal investigator is thus faced with a very difficult challenge on trying to keep up with these cybercriminal operational advantages [67]. Autopsy (https://www.autopsy.com/, accessed on 8 October 2020) is a digital forensics tool that helps to level out this field. It is open-source and widely used by criminal investigator teams to analyze and identify digital evidence and artifacts of suspicious and anomalous activities. It incorporates a wide range of native modules to process digital objects, including images (on raw disks), and also provides a highly effective framework that allows the community to develop more modules for otherwise more specialized forensics tasks.

Considering the exponential growth of technology, it is possible to notice that nowadays there are very few people not editing their photos to make them more appealing, or even to change something in their appearance such as removing a pimple from their face. With the increase of tools for this purpose, such as Photoshop (https://www.adobe.com/pt/products/photoshop.html, accessed on 12 April 2021), it is fairly easy to manipulate photos and increasingly difficult to detect this type of manipulation with the naked eye.

Multimedia content has an increased interest in digital forensics because of its association with crimes that have a high impact on public opinion. Sexual abuse and distribution of pornographic content, especially involving under aged, are some examples where multimedia content is relevant to digital forensics. Another troubling example of a crime using multimedia content is digital kidnapping, which consists of stealing the photos of a person, posing as him/her in order to reveal private information and thus having a negative impact on the victim's life. Digital forensics in this case is indispensable to analyze artifacts collected in the practice of these crimes and

determine responsibilities.

The multimedia content manipulation has also captured the interest of fake news detection, specially on social networks like Twitter or Facebook. The contents are published and read by millions of users, and the social networks providers are barely aware about their veracity. For example, fake videos spreading erroneous information in social networks leverage the citizens' misinformation and may influence (positively or negatively) the elections process.

The tools that allow the creation and manipulation of multimedia content (photos and videos) have nowadays become consequently more sophisticated and accessible to everyone, without leaving traces perceptible to the naked eye, and therefore the authenticity of the images cannot be taken for granted, especially if they are evidence in an investigation of a digital crime. Copy-move, splicing, resampling, and retouching are among the most widely used multimedia manipulation techniques that can be applied in digital photos and videos.

Defacing and deepfake (a type of splicing manipulation) take advantage of multimedia content manipulation techniques to tamper digital photos and videos. These manipulations consists of changing the components of a photo or video, for instance, to change the face of a person to make it looks like someone else or saying something that the person didn't said, which usually inflict severe reputational damages on their victims. These cyber threats use hyper-realistic videos that apply Artifical Intelligence (AI) techniques to change what someone says and does [84]. Coupled with the reach and speed of social media, convincing deepfakes can quickly reach millions of people, negatively impact society in general, and create real havoc on the lives of its victims. A deepfake attack may have different motivations. Fake news [18], revenge porn [37], digital kidnapping, usually involving under-aged or otherwise vulnerable victims [76], associated with ransomware blackmailing, are among the most relevant forms of deepfaking attacks that can create havoc on the lives of its victims.

Machine Learning (ML) has boosted the automated detection and classification of digital artifacts for forensics investigations. Existing ML techniques to detect manipulated photos and videos [79] are seldom not fully integrated into forensics applications. Therefore ML-based Autopsy modules, capable of detecting deepfakes and even other types of multimedia content manipulations, are relevant and will most certainly be very appreciated by the criminal investigation teams. The good results already observed by the reported ML methods for detecting multimedia content manipulation have not yet been fully translated into substantial gains for cybercrime investigation, as those methods have not often been incorporated into the most popular state-of-the-art digital forensics tools [14].

There are several methods of manipulation detection suggested by researchers. In this dissertation some focus was given to two methods, namely the detection of copy-move [59] and deepfake [32] manipulation types. The first method reviewed, "Digital image forensics technique for copy-move forgery detection using dog and orb", is used to detect copy-move and consists in three main steps: boundary detection, features extraction, and features matching. With this method, Patrick Niyishaka and Chakravarthy Bhagvati of the University of Hyderabad

in India were able to achieve an F1-score of 93.82% when detecting manipulated photos. The second method studied and the one used in the development of this dissertation "Unmaking deepfakes with simple features", is based on a classical frequency domain analysis followed by a ML based classifier. Compared to "Digital image forensics technique for copy-move forgery detection using dog and orb" method, which need to be fed with large amounts of data, this method achieved very good results using only a few training samples and a good accuracy in unsupervised scenarios. Ricard Durall et al [32] achieved an accuracy of 100% detecting deepfake manipulations in high-resolution photos and 91% in low-resolution videos. The scores obtained by this second method led us to choose it as the basis for the development in the scope of this dissertation.

Despite the developments made so far on the detection of tampered multimedia content, there are however some limitations and open issues. Deepfake videos in existing datasets reveals some severe contrasts in visual quality to the actual deepfake videos circulating on the Internet. Several common visual artifacts can be found in the latter, namely low-quality synthesized faces, visible splicing boundaries, and color mismatch. These artifacts are usually not visible in the videos circulating on the Internet, otherwise they would not be credible. Furthermore, high detection performance on these datasets may not bear strong relevance [66]. Although existing video manipulations detection methods are mostly based on binary classification at the frame level, and the temporal consistency among frames are not explicitly considered. That is, as many videos exhibit temporal artifacts and real or fake frames tend to appear in continuous intervals, it gives a potentially wrong score when detecting manipulations in videos.

Even though the literature review covers a significant amount of research papers on multimedia contents manipulation detection, emergent research has provided additional detection methods.

Many forensics investigators do not have a background in the area, and consequently are not prepared to develop and implement algorithms for the detection of manipulated multimedia content. Nowadays it is consensual that there is a need to invest in the development of tools that may automate the detection process, to further enhance the efficiency of analysing artifacts in a digital forensics investigation. By knowing the probability that certain artifacts may or may not have been manipulated, it is possible for investigators to focus on the most relevant artifacts to the case, and decrease the analysis time to quickly find the perpetrators.

The whole community related to the detection of manipulated multimedia content, namely the criminal investigation teams and the victims, will benefit from the contributions present in this dissertation by increasing the effectiveness with which they may analyze digital evidence related to multimedia content and decreasing the time for solving cases related to these manipulations.

## 1.1 Main goals

The following objectives were defined for the research work presented in this dissertation:

- To identify and evaluate existing algorithms in the literature, which are related to the detection of photos and videos manipulation, namely those that were subjected to copy-move, splicing, and deepfake tampered techniques.
- To study the Autopsy tool functioning, particularly the API available in Python for the development of new modules.
- To deploy and develop a standalone application to ingest and process photos and video frames. For each input file, the application evaluates the probability of has been manipulated. The algorithms identified in the literature to extract the features from files and to classify them, were respectively Discrete Fourier Transform (DFT) and Support Vector Machines (SVM).
- To setup a dataset according to the literature review. The resulting dataset was made available in its original form, as well as the ready-to-use version with the features extracted through the DFT method.
- To perform tests using the resulting dataset and to evaluate and discuss the results obtained.
- To benchmark the proposed method with the state-of-the-art cutting edge ML methods described in the literature for image forensics, namely deep Convolution Neural Networks (CNN) based learning approaches.
- To deploy and develop two Autopsy modules, based on the standalone application. The modules process separately photos and video files.
- To publish the Autopsy modules on the 2021 Autopsy Module Development Contest (https://www.osdfcon.org, accessed on 10 November 2020), which is included on the Open Source Digital Forensics (OSDF) Conference. The submission of the work to the Autopsy Module Development Contest is schedule to 15 September 2021 (https://www.osdfcon.org/2021-event/2021-module-development-contest/).
- To perform acceptance and usability tests for the developed modules.

The goals initially defined for this dissertation were totally attained. The detection of manipulated videos was not initially full defined, but throughout the research this feature was also included on the list. The software and the dataset produced was publicly made available on GitHub repositories (described in Section 1.2), under a MIT license.

## 1.2 Contributions

The research work leading to the realization of this dissertation, has achieved a two-fold contribution, that derived from the goals initially defined, namely: 1) two Python modules for the Autopsy tool were developed, which have contributed to automate the detection of tampered

photos and videos files; 2) a processing framework of multimedia files, which was deployed to process and classify the files as being genuine or manipulated. The overall architecture is composed of a pre-processing phase to ingest the input files and to produce the learning SVM models, a processing phase to calculate the probability of a file has been manipulated, and a results analysis phase.

A standalone application was firstly developed, to study the detection method and evaluate the results, before starting the development of the Autopsy modules. A benchmark with CNN approaches, namely by evaluating both classification performance and processing time, gave confidence to carry on with the adoption of the SVM classifier.

Regarding software development, the following items were publicly made available into two GitHub repositories:

- A labeled dataset composed of about 50,000 multimedia files was produced. It incorporates state-of-the-art datasets of both normal examples and those subjected to some kind of manipulation, namely splicing, copy-move and deepfaking. The dataset is composed of the files resulting of DFT features extraction processing, and is available at the following GitHub link: https://github.com/saraferreirascf/Photos-Videos-Manipulations-Dataset.
- The development of two ready-to-use Autopsy modules: one module was designed to detect the fakeness level of digital photos; the other module was designed to detect the fakeness level of input video files. The modules take advantage of the SVM-based model implemented as a standalone application and have been made available in the following GitHub link: https://github.com/saraferreirascf/Photo-and-video-manipulations-detector. The Python modules are ready to be installed and used on Autopsy tool.

The following papers were published during the development of the research:

- Sara Ferreira, Mário Antunes, Manuel E. Correia; "Forensic analysis of tampered digital photos"; $25^{th}$ Iberoamerican Congress on Pattern Recognition (CIARP); May 2021; Porto; Portugal; to be published in Springer Lecture Notes on Computer Science. Paper is available in the proceedings of the conference, at https://ciarp25.org/wp-content/uploads/sites/10/2021/05/CIARP25-Papers.pdf (accessed on 16 June 2021), pp.402-411.
- Sara Ferreira, Mário Antunes, Manuel E. Correia; "Exposing Manipulated Photos and Videos in Digital Forensics Analysis"; J. Imaging 2021; on review
  - By suggestion of the journal's (https://www.mdpi.com/journal/jimaging/special_issues/image_video_forensics, accessed on 14 June 2021) reviewers, a benchmark with CNN-based methods was added, a proofreading was made to the text, to correct some typos, and grammatical issues and it was also included the recent published works which employed Generative Adversarial Networks (GAN), CNN and Recurrent Neural Networks (RNN) to detect manipulated multimedia content.

Bearing in mind all the contributions produced by this dissertation, it can be seen that improvements have been achieved in the field of automatic detection of manipulated multimedia content directly linked to digital forensics. There were also improvements taking into account the dataset created. Until now there was no dataset that simultaneously contained faces and objects in both photos and videos, aggregating several types of manipulations. By creating a more complete dataset, it is possible to improve new and forthcoming ML models for the detection of multimedia content, with higher accuracy on detecting manipulated multimedia content.

## 1.3 Dissertation layout

This dissertation is divided into six chapters, which are described as follows. The fundamental concepts necessary to understand the work being presented, namely digital forensics fundamentals and tools, machine learning methods, and techniques to manipulate and detect tampered multimedia content are presented in Chapter 2.

In Chapter 3 a comprehensive state-of-the-art and literature review is presented, to sustain the decisions made. The techniques available to analyse and detect photos and videos content are highlighted. Regarding ML techniques applied to detect tampered photos and videos, SVM and CNN are explained and the most relevant research is detailed.

Chapter 4 explains the overall process of digital forensic analysis using Autopsy tool. The most common and well-known modules for file ingestion and analysis are described, emphasising those related with artifacts detection and classification. In this Chapter is also described the two modules developed for Autopsy tool, namely one for the detection of manipulated photos and one for the detection of manipulated videos. This chapter outlines the architecture used in the development, namely its main components.

Chapter 5 presents the metrics used to evaluate the performance of the modules and the datasets used for the subsequent modules testing are also presented. The results obtained and their corresponding analysis are described and evaluated and a benchmark with a CNN-based approach is detailed.

Finally, in Chapter 6, the conclusions that were reached are presented and some lines of development that can be used in future work are proposed too.

# Chapter 2

# Fundamental concepts

In this Chapter, the fundamental concepts for the comprehension of this document will be presented, namely the definition of digital forensics, the most commonly used digital forensics applications, the description of the development process of modules for Autopsy (https://www.autopsy.com/, accessed on 8 October 2020), the fundamentals of photos and video files, the main multimedia content manipulation methods, and an introduction to Machine Learning (ML).

## 2.1 Digital forensics

Computer Forensics, also commonly referred to as digital forensics, is a sub-area of cybersecurity which aims to integrate techniques and procedures for collecting, preserving, and analyzing evidence on electronic equipment, namely disks, smartphones, and other storage-capable devices [30].

It is an indispensable tool for law enforcement, through criminal investigation teams, namely in the analysis and identification of artifacts, and digital evidence related to criminal activity using electronic equipment. The Portuguese criminal police agencies, commonly named Orgãos de policia criminal (OPC), are institutions under the authority of the Government through its Ministries of Justice and Defense, which cooperate with the judicial authorities in criminal investigations. The most relevant criminal police agencies present in Portugal are the Polícia Judiciária (PJ) (http://policiajudiciaria.pt/, accessed on 4 March 2021), Guarda Nacional Republicana (GNR) (http://www.gnr.pt/, accessed on 4 March 2021), and Polícia de Seguranca Pública (PSP) (http://www.psp.pt, accessed on 4 March 2021). PJ has recently created the Unidade Nacional de Combate ao Cibercrime e à Criminalidade Tecnológica (UNC3T) meant to lead the cybercrime and digital forensics investigations of seized electronic equipment.

The main goal of digital forensics analysis is to produce a scientifically supported reconstruction of events, helping in the process of collecting evidences that may dictate the innocence or guilt of a suspect. As a useful tool to help in the decision making process, digital forensics has a growing presence in the Portuguese judicial paradigm and in the organizational context. Digital forensics can be defined as the systematic, technical inspection of a computer system and its components

(data, hardware) to obtain digital proofs, or evidences, of a crime that needs to be investigated. These digital evidences, need to respect the same fundamental principles as physical evidences, namely:

- **Admissibility:** The proof must be in accordance with the existing legislation.
- **Authenticity:** Must be authentic in the relationship between the clue and the incident.
- **Completeness:** It has to be whole, with nothing missing.
- **Reliability:** In the description of all actions performed in handling evidences needs to be reliable.
- **Credibility:** Evidence has to be understandable and plausible.

Accordingly, digital evidence must be legally admissible and the way it was obtained should be clearly known. It should also be technically incontestable, its origin must be verifiable, and its integrity must also be demonstrable.

The Portuguese Law No.109/2009 commonly known as the "Cybercrime Law", emerged in Portugal, to frame two major groups of illicit activities made by means of a computer and through the Internet infrastructure: activities related with traditional crime, like robbery or drug trafficking, that uses computers and other digital devices; activities that explore vulnerabilities in information systems, such as remote access, malware spreading, among others. The 109/2009 law updated some existing norms to the digital context, and created new ones adequate to the digital society in which we are living. For example, norms related with violation of correspondence, that are related with post mail, were updated to accommodate the same subject but in a digital format. For those scenarios that are not covered in 109/2009 law, the Code of Criminal Procedure continues to apply [11].

Cybersecurity professionals and digital forensic investigators understand the value of the information extracted from seized equipment and need to respect the fact that it can be easily compromised if not properly handled and protected. For this reason, it is crucial to establish and follow strict guidelines and protocols in digital forensic investigations, to preserve digital evidences. These procedures include detailed instructions about the time frame digital forensics investigators are authorized to recover potential digital evidence, how to properly prepare systems for evidence recovery, where to store any recovered evidence, and how to document these activities to help ensure the authenticity of the data [61]. The result of digital forensics activities is a technical report which details the artifacts found in an analyzed equipment [11]. Section 3.1 details the whole process of digital forensics analysis.

The general methodology of a forensics investigation is divided into four main phases [11]:

1. **Preparation:** Physical labelling and inventory of the digital equipment. In this phase, the best analysis approach is also identified.

2. **Acquisition:** Extraction of data stored in the equipment that is being object of digital expertise. In this phase it is crucial to perform all the procedures that ensure the preservation of the digital evidence.

3. **Analysis:** Investigation of the available data extracted in the acquisition phase.

4. **Presentation:** Production of technical reports in which the identification of the equipment, the results of the investigations, and the conclusions drawn from the analysis are stated.

It is possible to observe that digital evidences are subjected to several phases during the digital expertise. Figure 2.1 depicts the digital evidence path, from the incidence occurrence, to the acquisition and validation of digital evidences.

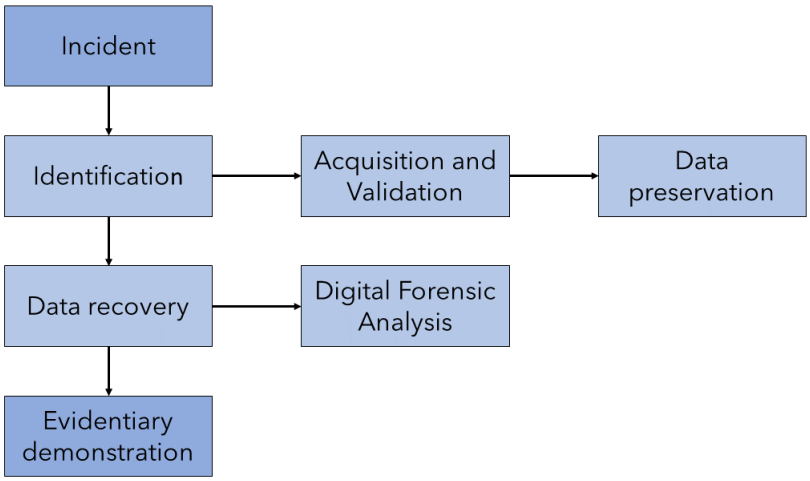



Figure 2.1: Digital evidence path.

From the incident or criminal act committed, to the probatory demonstration, the evidence is subject to several phases. During the first phase, the evidence is identified. This identification can be done directly, through the equipment where it is stored, or, if the evidence is not accessible, an authorization is required to handle it. After authorization, the evidence is sorted and identified. If neither the location of the evidence is known nor permission to handle it has been granted, the evidence is immediately preserved. Following this initial phase, in a laboratory, data recovery and digital forensic analysis is performed.

## 2.2 Digital forensics applications

Digital forensics software applications are a helping hand for criminal investigators, especially in data extraction and analysis phases. This Section introduces some well-known digital forensics tools that are widely used by criminal police departments. Only the forensics tools fully dedicated

to extraction and/or analysis are described in this Section, namely Autopsy, EnCase, Forensic Toolkit (FTK), XRY and Cellebrite.

Autopsy (https://www.autopsy.com/, accessed on 8 October 2020) is an open source, fast, easy-to-use forensic platform capable of analyzing all types of mobile devices and digital storage media. This tool has a graphical interface to the command-line digital investigative analysis tools of The Sleuth Kit (https://www.sleuthkit.org/, accessed on 8 October 2020), and allows the use of community-developed modules and the development of custom ones.

For a better understanding of this tool, it is important to understand its key concepts:

1. Autopsy defines a case as a digital container, which is composed of one or more data sources. Only one case can be opened at a time, and to start an investigation one case must be opened.
2. *Data source* is the term used to refer to disk images and logical files that are added to a case, and correspond to the extraction previously made from an electronic equipment (e.g. disk).
3. Autopsy maintains a central database, usually SQlite or PostgreSQL, to persistently store all files, their metadata, and results analysis. These artifacts are attached to the Autopsy case for a continuous analysis during the lifetime of the case in Autopsy.
4. The results obtained in the analysis phase are usually posted on a blackboard section available on Autopsy. These collected artifacts are persistently available to be accessed by the user through a Graphical User Interface (GUI).

Autopsy has four types of modules, which are detailed below, namely Ingest modules, Report modules, Content viewers and Result viewers [5].

1. **Ingest Modules:** Ingest modules analyze data present in data sources. This type of modules perform all of the analysis of the files and parse their contents. There are two types of ingest modules:
    - **File Ingest Modules:** During their lifetime, they are passed in each file in the data source. This includes files that are found via carving or inside of ZIP files. This type of modules can be used, for example, to do hash calculation and lookup in order to prove the authenticity of evidences.
    - **Data Source Ingest Modules:** Data source-level ingest modules gets passed in a reference to a data source and it is up to the module to search for the files that it wants to analyze. These modules can use a database to query one or more files and perform analysis on them. This type of modules can be used, for example, to do web analytics where the name and path of the databases are known, as well as registry analysis where the path of the hives are known.

2. **Report Modules:** These modules are typically executed after a user has examined all results. The purpose of this module is to report the results, that is, the artifacts found, however they can also be used to perform analysis. Autopsy comes with modules to generate HTML and Excel artifact reports, a tab delimited File report, Keyhole Markup Language (KML) report for Google Earth data, and a body file for timeline creation. There are three types of report modules:

    - **Table report modules:** In this module the data is organized into tables.
    - **File report modules:** With this module, the data is also organized into tables but they deal specifically with reporting on the files in the case, not artifacts.
    - **General report modules:** By using this type of module it is possible to organize the data in in a customizable and free-form format.

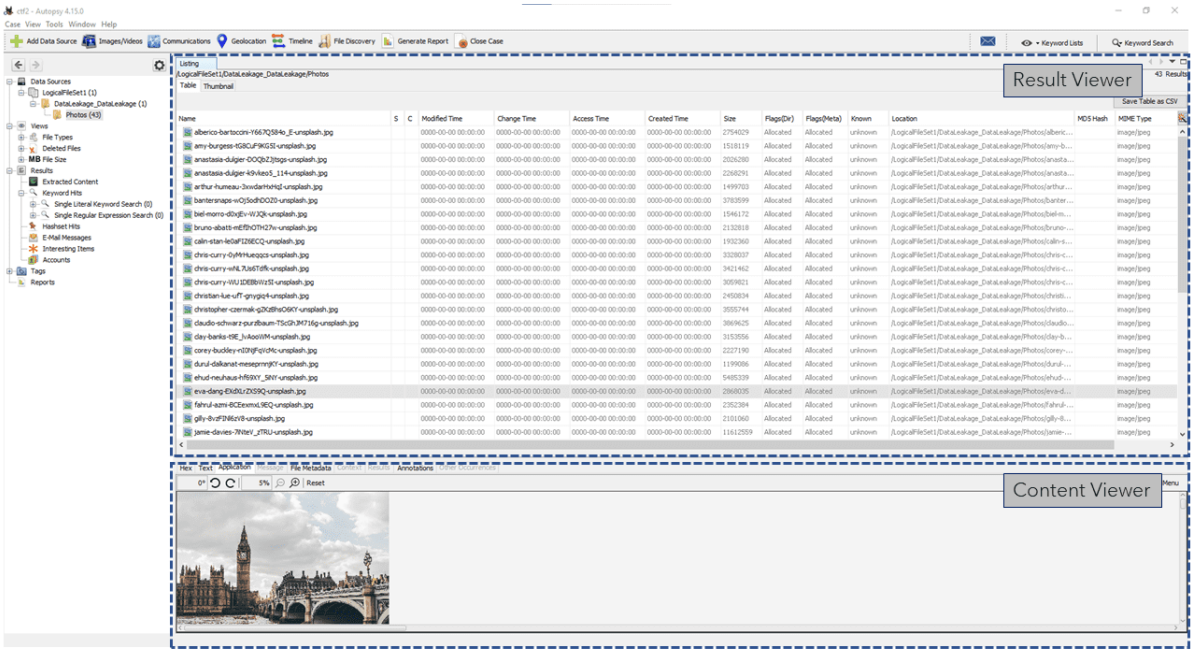



Figure 2.2: Location of content viewers and result viewers in Autopsy GUI.

3. **Content Viewers:**

    - These modules are graphic and focused on displaying a file in a specific form. The Content Viewer exists in the lower right side of the Autopsy main screen (Figure 2.2) and shows pictures, video, hex, text, extracted strings, metadata found in data sources.
    - The Content Viewer is context-aware, meaning different tabs will be enabled depending on the type of content selected and which ingest modules have been run. For example, selecting a `JPG` file will cause the Content Viewer to automatically select the "Application" tab and will display the image there.

4. **Result Viewers:**

- This kind of modules exist in the upper-right area of the default Autopsy interface (Figure 2.2).
- These modules display information about a set of files that are passed into the viewer from the tree on the left, keyword searching, or other searches. The main idea is that the same set of files can be viewed in a table format, thumbnail form, or any other form.

There are some modules directly available through Autopsy, like `FileType Identification`, `Email Parser`, and `Encryption Detection`. In Figure 2.3 it is possible to see some of the available modules for Autopsy.

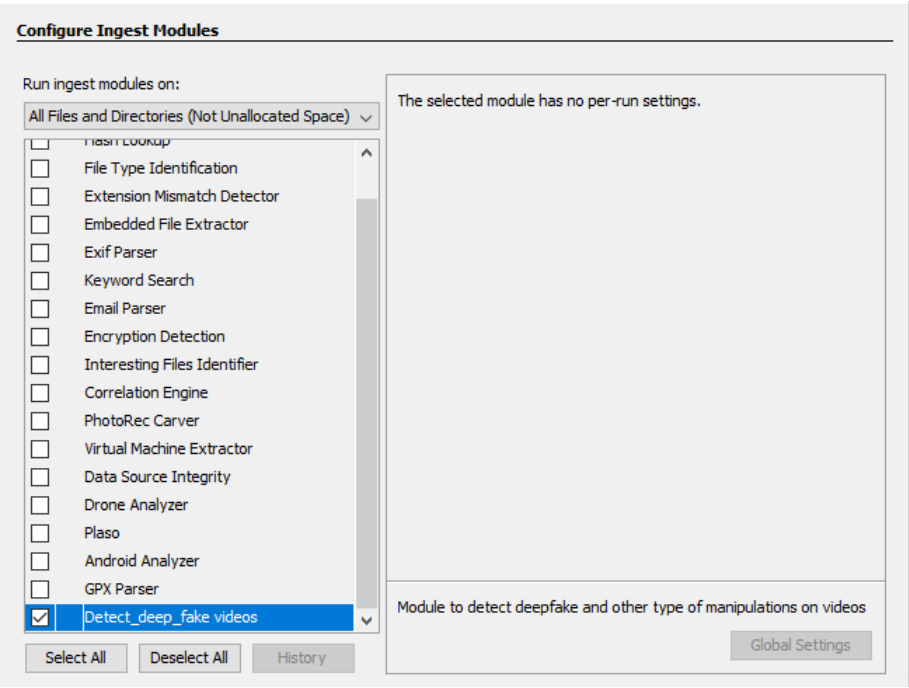



Figure 2.3: Available modules through Autopsy.

`FileType Identification` module identifies files based on their internal signatures and does not depend on file extensions. For example, it can be used to find digital evidences. For example, photos in a data source even if the extension is wrong.

`Email Parser` module identifies files in `MBOX`, `EML` and `PST` formats, based on file signatures. It extracts e-mail messages from the files, and adds the results attained to the Autopsy blackboard. This module skips known files and creates a blackboard artifact for each message. It also adds e-mail attachments as derived files. The `Encryption Detection` module looks for files that can be potentially encrypted using both a general entropy calculation and additional specialized tests for certain types of files. This module can be helpful, for example, if an image extracted from a suspect's disk is being analyzed, revealing interesting files for the investigation, because encrypted files are more likely to contain sensible information.

In Section 4.1 a more complete description on how to use this forensic tool in a digital forensics

investigation is provided. Two new modules were developed in the scope of this dissertation. One module for the detection of manipulated photo and another for the detection of manipulated videos. Their architecture is described in Section 4.3.

EnCase is a digital forensic tools delivered by Opentext Security https://security.opentext.com/encase-forensic, accessed on 8 October 2020), which has other software products designed for forensics, security analytic, and e-discovery. Encase is traditionally used to recover evidence from seized hard disk drives and support mobile devices extraction and analysis. It allows the investigator to conduct an in-depth analysis of user files to collect evidence such as documents, pictures, Internet history, and Windows Registry information, among other features. This software is proprietary and the license has associated costs.

FTK is an open source software delivered by AccessData https://accessdata.com/products-services/forensic-toolkit-ftk, accessed on 8 October 2020). It provides real-world capabilities to help digital forensics investigation teams to separate critical data from trivial details, and protect digital information while complying with digital regulations. This tool can be used by both criminal police and the private sector to perform complete forensic examinations of an electronic equipment. It includes customizable filters that allows the examiner to inspect a large amount of files, including locating e-mails purportedly excluded from a computer. This feature is compatible with Outlook, AOL, Outlook Express, Netscape, Earthlink, Yahoo, Hotmail, Eudora, and MSN e-mail services.

XRY, from MSAB (https://www.msab.com/products/xry/, accessed on 8 October 2020), is an intuitive and efficient software for Windows. It allows a fast and secure high-quality data extraction from mobile devices while maintaining the integrity of the evidence. XRY allows a rapid logical and physical extraction of files and its file format maintains secure and accountable evidence at all times, with complete forensic auditing and protection of evidence from the moment the extraction begins. It is a proprietary software and has associated licensing and acquisition costs.

Cellebrite (https://www.cellebrite.com, accessed on 8 October 2020) is an Israeli toolset used for the collection, analysis, and management of digital data. It is a competitor of XRY for mobile devices extraction and analysis, providing a wide set of features to extract data from digital devices.

## 2.3 Digital photos and videos fundamentals

The word *image* comes from the Latin *imāgo*. An image is an artifact that represents the visual perception of a physical object, providing us with a representation of it [24]. Broadly speaking, images can be classified as having two dimensions (Figure 2.4(a)) like a photograph, or three dimensions (Figure 2.4(b)) like a hologram.

(a) Photo. (b) Hologram.

Figure 2.4: Examples of 2D and 3D images.

In addition, an image can be static as in the photos, or it can have movement, as typically found in videos. The term "photos" will be used throughout the document, instead of "images", not to confuse the meaning of "photo" with disk images.

A digital photo is an image composed of pixels. The word *pixel* comes from the words *picture* and *element* and means an *"element of the image"* to which it is possible to assign colors and other information. The combination of all the pixels in an image is called *resolution*.

Each photo also contains a set of relevant informational data, often called *metadata*, which describes and provides information about the photo [27] as depicted in Figure 2.5. Some examples are the file type and image width and height.

| File | |
|---|---|
| File Type | JPEG |
| File Type Extension | jpg |
| MIME Type | image/jpeg |
| Image Width | 1920 |
| Image Height | 1079 |
| Encoding Process | Baseline DCT, Huffman coding |
| Bits Per Sample | 8 |
| Color Components | 3 |
| Y Cb Cr Sub Sampling | YCbCr4:2:0 (2 2) |
| **JFIF** | |
| JFIF Version | 1.01 |
| Resolution Unit | None |
| X Resolution | 1 |
| Y Resolution | 1 |
| **Composite** | |
| Image Size | 1920x1079 |
| Megapixels | 2.1 |

Figure 2.5: Metadata analysis with Fotoforensics (http://fotoforensics.com/, accessed on 15 January 2021).

It is possible to extract the metadata associated with a photo file, in a way that can be perceivable by both software applications and humans. This information is crucial for easy location and identification of digital photos.

Metadata is stored in two distinct places: internally, embedded in the photo file, or externally as in a Digital assets management system (DAM) (system that stores, shares and organizes digital assets in a central location) like MediaValet (https://www.mediavalet.com/product-tour/, accessed on 6 June 2021).

Metadata is divided into the following three main categories:

- **Descriptive:** Information about the visual content of the photo, such as title, caption, and keyword.
- **Rights:** Information about the creator, copyright, and terms of use rights.
- **Administrative:** Date, and place of creation.

Video is an electronic media for recording, copying, reproducing, broadcasting and displaying moving visual media [41]. It was initially developed for mechanical television systems, which were quickly replaced by cathode ray tube (CRT) systems, which were later replaced by flat screens of various types, like the ones that we have nowadays.

Video systems range in display resolution, aspect ratio, refresh rate, color capabilities and other features. There are analog and digital variants that can carried on a variety of media, including radio broadcast, magnetic tape, optical discs, computer files, and network streaming. It is also possible to get the video metadata using tools like `exiftool` (https://exiftool.org/, accessed on 2 May 2021) as despicted on Figure 2.6.

```
saraferreira@MacBook-Pro-de-Sara Documents % exiftool 00000.mp4
ExifTool Version Number         : 12.00
File Name                       : 00000.mp4
Directory                       : .
File Size                       : 2.3 MB
File Modification Date/Time     : 2021:06:01 12:01:28+01:00
File Access Date/Time           : 2021:06:01 12:01:28+01:00
File Inode Change Date/Time     : 2021:06:01 12:01:28+01:00
File Permissions                : rw-r--r--
File Type                       : MP4
File Type Extension             : mp4
MIME Type                       : video/mp4
Major Brand                     : MP4  Base Media v1 [IS0 14496-12:2003]
Minor Version                   : 0.2.0
Compatible Brands               : isom, iso2, mp41
Media Data Size                 : 2406376
Media Data Offset               : 44
Movie Header Version            : 0
Create Date                     : 0000:00:00 00:00:00
Modify Date                     : 0000:00:00 00:00:00
Time Scale                      : 1000
Duration                        : 15.00 s
```

Figure 2.6: Metadata analysis with exiftool.

When talking about videos it is important to mention frames. A frame is one of the many still images which compose the complete moving picture. When the moving picture (video) is displayed, each frame is flashed on a screen for a short time and then immediately replaced by the next one. Persistence of vision blends the frames together, producing the illusion of a moving photo, that is, a video.

## 2.4 Multimedia manipulations

Digital photos and videos manipulation is a very appealing and highly effective medium to spread misinformation. There are four popular main types of manipulation that can be applied to a multimedia file, namely copy-move, splicing, retouching and resampling. Despite the similarity of the overall final result, as it consists mainly of manipulating objects, faces, or voices in multimedia files, the methods to produce and further enhance the manipulation and then the ML techniques employed to detect these manipulations, are very distinct and can be highly challenging for a fully automated detection system.

The multimedia manipulation can be generally achieved by using specific software applications, like Adobe Photoshop (https://www.adobe.com/pt/products/photoshop.html, accessed on 12 April 2021) or GNU Gimp (https://www.gimp.org/, accessed on 28 May 2021). There are also specific algorithms that can be used to manipulate images, namely Content-Aware Fill e PatchMatch for copy-move, Content-Aware Healing for copy-move and splicing and Clone-Stamp and Alpha Matting for splicing [74].

(a) Original photo. (b) Manipulated photo.

Figure 2.7: Iranian missile test

There are good examples of how copy-move manipulation can be used to provoke great impact on spreading fake information and wrest the original photo from its context. Nearly a decade ago, Iran was accused of doctoring a photo of its missile tests. The photo depicted in Figure 2.7 was released on Iran's Revolutionary Guards official website, which claimed that four missiles were heading skyward simultaneously [36], when in fact only three missiles were launched. More recently, in July 2017, a fake image of Russian President Vladimir Putin was distributed on social

media related to his meeting with US President Donald Trump during the 2017 G20 summit. This fake image garnered several thousand likes and retweets [78].

**Copy-move** manipulation is a technique that consists on copying or moving parts of a photo to another place in the same photo. The goal is to give the illusion of having more elements in the photo than those that are really there. Figure 2.8 depicts an example of copy-move, in which it is possible to observe the addition of a new element in the resulting photo (Figure 2.8b).

The main reason behind the increase of copy-move manipulation is the simplicity of the method. Textured areas such as grass, foliage, gravel, or fabric with irregular patterns, are ideal for this purpose because the copied areas will probably blend into the background and the human eye will not be able to easily discern any suspicious artifacts. As the copied parts came from the same digital photo, their noise, color palette, dynamic range, and other important properties, will be compatible with the rest of the photo. This type of manipulation is difficult to detect by methods that look for statistical measurements incompatibilities in different regions of the photo [34].

(a) Original image. (b) Manipulated image.

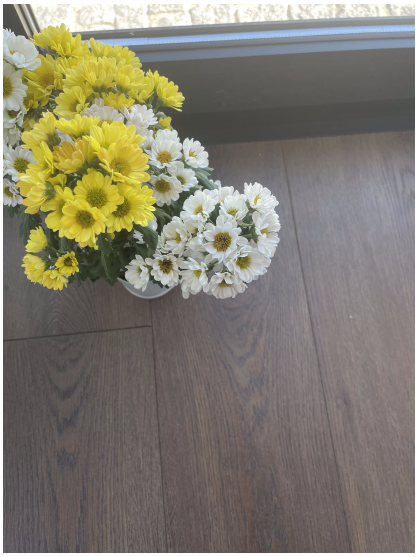
Figure 2.8: Copy-move manipulation.

**Splicing** (Figure 2.9) consists of superimposing different regions of two photos, being deepfake the most relevant consequence. It is an artificial and automated manipulation of media, usually made by employing Artifical Intelligence (AI) techniques, in which a person's face in an existing photo or video is swiped by someone else's face. This manipulation is often used as an initial step of photo montage, which is very popular in digital photo content editing. The splicing tampered image could be used in news reports, photography contests, key proof in academic papers, and so on, which could bring negative impact.

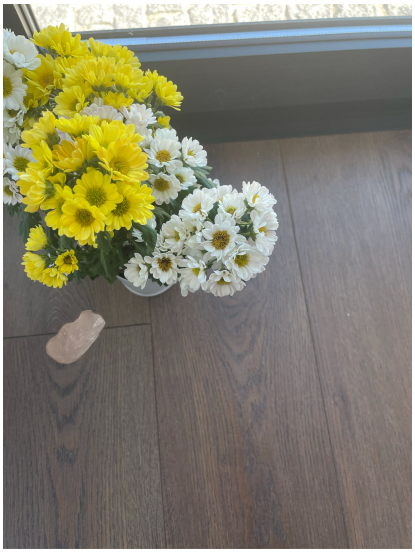
As a result, it is an important issue to develop reliable splicing detection methods. In the forgery process, the manually introduced transitions of edges and corners are different from those in the original photo. The differences are commonly described by the inconsistency and abnormality, and they are used for splicing detection [85].

Splicing is comparatively more difficult to detect than copy-move. While in copy-move it is easy to detect similar outlines of objects in the same image, as they will have the same texture, transitions (curves and lines), and size, in splicing we have different objects with different textures and image characteristics, thus making it difficult to detect them.

(a) Original image.



(b) Original image.



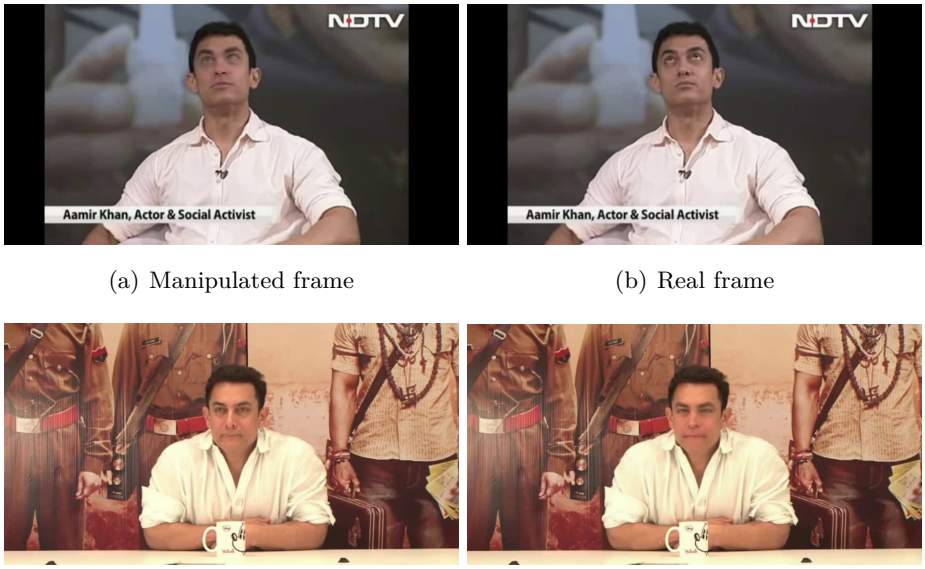
(c) Manipulated image.

Figure 2.9: Splicing manipulation.

The term **deepfake** is the combination of "deep learning" and "fake" [84]. In general, deepfake

is achieved by manipulating realistic videos or photos with the aim to depict people saying or doing things that did not happen. This kind of manipulation is usually difficult to detect, as it uses real footage to make it the closest thing to reality.


Figure 2.10: Example of deepfake manipulation.




(a) Manipulated frame (b) Real frame

(c) Real frame (d) Manipulated frame

Figure 2.11: Comparison between fake and real frames in deepfake videos.

Figure 2.11 depicts four frames extracted from two videos. Figures 2.11b) and 2.11c) are legitimate, while Figures 2.11a) and 2.11d) are manipulated. As can be seen in Figure 2.11a) in comparison with 2.11b), the eyes seem a little foggy and looking in opposite directions. If someone without knowledge of the real frame was looking at the manipulated one, might think it was a cross-eyed person and not notice that it was deepfake. In Figure 2.11d), similarly to Figure 2.11a), the face is a little foggy and blurry even, and the quality is not compatible with the rest of the photo. In some cases, when comparing two photos, it is easy to see which is the deepfake, but having access only to the manipulated image, with the naked eye it is not so perceptible that the photo has been manipulated, further demonstrating the importance of creating automatic and adaptive applications that may be able to identify this type of manipulation.

While deepfake of photos and videos is not new and can be observed in numerous digital

contents, it has leveraged powerful ML and AI techniques to improve contents manipulation [50]. The most common ML methods used to improve deepfake are based on deep learning and involve training generative neural network architectures, such as auto-encoders [58]. By using Generative Adversarial Networks (GAN), two artificial neural networks work together to create a real-looking media. The first neural network, usually called "the generator", tries to create new samples that are good enough to trick the second network training with a real images dataset. In conclusion, a GAN can look at thousands of photos of a person, and produce a new portrait that approximates those photos without being an exact copy of any one of them.

Philip Wang released the website https://www.thispersondoesnotexist.com/, which makes use of an AI system developed by researchers at Nvidia - StyleGan, to create the most realistic-looking faces on nonexistent people that machines have ever produced [56]. Every time the webpage is loaded a new face like the one depicted in Figure 2.12, is shown to the user.

Figure 2.12: Face of a person that does not exists.

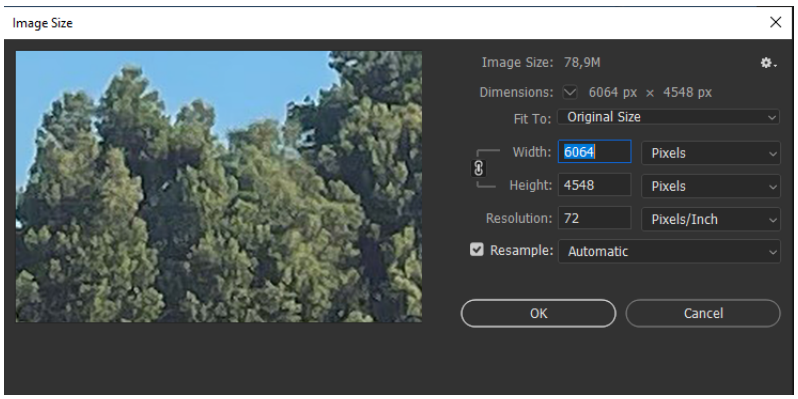
Deepfake has garnered widespread attention, as it has been used in digital campaigns of spreading fake news. This manipulation technique is also responsible for innovative and widely spread digital kidnap, revenge porn, and financial fraud cybercrimes [26, 68].

**Resampling** is the process of geometrically transforming digital photos. Unlike resizing, resampling changes the size of an photo, in terms of resolution, meaning that the number of pixels will be changed too. When an image is resampled, it uses interpolation using Equation 2.1 by using known data to estimate values at unknown points to create pixels. When an image is upscaled (upsampling) as depicted in Figure 2.13, more pixels are added to the image resolution while in a downscaled (downsampling) image there are less pixels. AI techniques help improve upscaling by using ML methods with neural networks [77]. This type of manipulation can be used to recover old photos or even improve the visibility of photos in general.

$$y = y_1 + (x - x_1)\frac{(y_2 - y_1)}{(x_2 - x_1)} \tag{2.1}$$

(a) Original image.

(b) Manipulated image.




(c) Resampling in Photoshop.

Figure 2.13: Resampling manipulation.

**Retouching** is another type of photo forgery, which is most commonly used for commercial and aesthetic applications. Retouching operation is carried out mostly to enhance or reduce the photo features. In Figure 2.14 the author of the blue book was removed. Artists professionally use retouching to restore old historical photos with paintings, while malicious users use this type of manipulation to tamper the digital photos. Therefore, this kind of manipulation can be considered as a less harmful kind of digital photo manipulation. This type of forging is mainly used by magazine photo editors to make the picture more attractive, still meaningful that this kind of image modification is morally wrong [48].


(a) Original image. (b) Manipulated image.

Figure 2.14: Example of retouching where the author of the last book was removed.

As for all other manipulations, almost any editing software, like Adobe Photoshop (https://www.adobe.com/pt/products/photoshop.html,accessed on 12 April 2021) and GNU Gimp (https://www.gimp.org/, accessed on 28 May 2021), can be used to apply this kind of manipulation. In the examples depicted in this section, PicsArt (https://picsart.com/, accessed on 28 May 2021) mobile application was used.

## 2.5 ML fundamentals

ML can be defined as a set of techniques that allows computers to learn without the need of explicit programming, that is, to perform tasks for which an algorithm has not been defined to obtain the result.

Automatic learning in ML techniques can be supervised or unsupervised. Supervised learning is the most common type and involves presenting a set of previously classified (labelled) examples, from which it is possible to build a training model for subsequent application to new unseen examples (testing phase), in order to obtain their classification [72]. Supervised learning problems can be further grouped into regression and classification problems. When the goal is to place data into one or more predefined classes it is a classification problem. When the goal is to give a value (usually numeric) to an input, it is a regression problem (Figure 2.15).

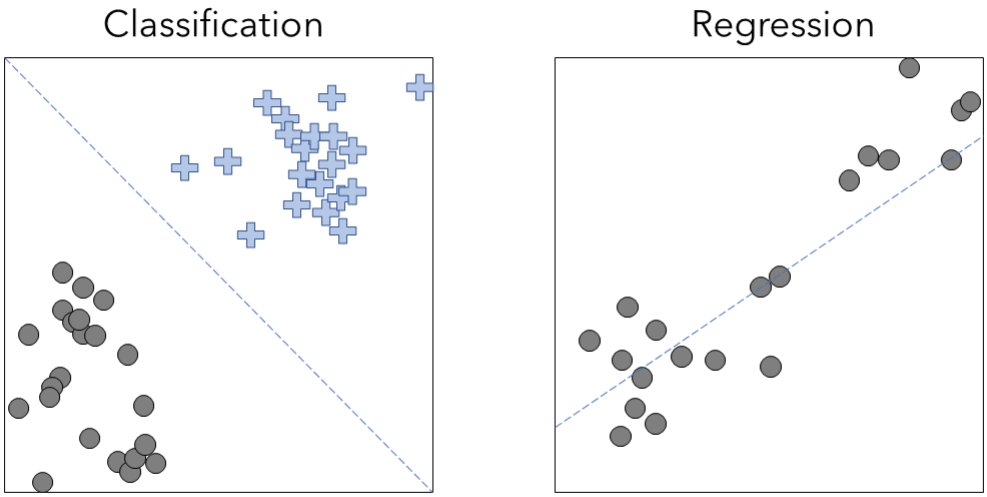



Figure 2.15: Difference between classification and regression.

The problem studied in the scope of this dissertation is essentially a classification problem, since the goal is to define whether a photo or video is manipulated or not. That is, a classification problem can be defined in this dissertation, in which each multimedia file can belong to one of the following two classes: legitimate or manipulated.

Some popular examples of supervised machine learning algorithms are:

- Linear regression for regression problems;
- Random forest for classification and regression problems;
- Support Vector Machines (SVM) for classification problems.

In unsupervised learning, the algorithm has to extract the learning from the available input. The goal here is to model an underlying structure or distribution in the data in order to learn more about the data itself. Many of these algorithms first perform data partitioning (clustering) in order to learn and classify the input into a finite number of classes [72].

Some popular examples of unsupervised learning algorithms are:

- k-means for clustering problems;
- Apriori algorithm for association rule learning problems.

When applying ML, most of the time and regardless the type of problem or algorithm that will be used, an initial data pre-processing step is required. This pre-processing phase is used to check for invalid data, missing values, repeated data, and to filter out the most relevant characteristics for the target problem.

Deep Learning is a subfield of ML concerned with algorithms inspired by the structure and function of the human brain. Deep learning algorithms are similar to how nervous system are structured where each neuron connects to each other and passes information. This type of algorithms works in layers and a typical model has, at least three layers. Each layer accepts the information from the previous one and pass it on to the next one. Deep learning tends to perform well with larges amount of data whereas old ML models stops improving after a saturation point [47].

The performance of most ML algorithms depends upon the accuracy of the features extracted. Trying to obtain high-level features directly from data is a major difference between deep learning and traditional ML algorithms. Deep learning reduces the effort of designing a feature extractor for each problem, but the time it takes to train a deep learning model comparing with a ML model is significantly higher [31].

## 2.6 Summary

This Chapter as introduced the concept of digital forensics, the main digital forensics applications were outlined namely Autopsy, EnCase, FTK, XRY and Cellebrite and the fundamentals of digital photos ans videos were presented. A brief introduction to ML fundamentals were made in this Chapter as well.

In this dissertation an SVM-based method was implemented to classify the multimedia files. In Chapter 3 the process of a digital forensics analysis will be the described along with algorithms to detect manipulated multimedia content. The concepts of SVM and Convolution Neural Networks (CNN) are going to be described as well.

# Chapter 3

# Literature review

Considering that the manipulation of multimedia content is a growing problem with great impact in today's society, there are already some algorithms for the detection of such manipulations. This Chapter starts with presenting the fundamentals behind digital forensics analysis. It continues by describing the detection methods analysed in this research, in order to develop a method that may cope efficiently with manipulated files detection. The Chapter ends with the fundamentals behind the Support Vector Machines (SVM) and Convolution Neural Networks (CNN) functioning.

## 3.1 Digital forensics analysis

Digital forensics on images can be broadly divided into the following six categories [25]:

1. The objective of **source classification** is to classify photos according to their source, such as scans versus digital camera images, Canon versus Kodak, and so forth.
2. **Device identification** aims at proving that a certain photo was taken by a specific device, and also that a certain camera took a certain photo or video.
3. **Device linking**, which connects groups of objects according to their source. For example, given a set of images, we would like to find out which images were taken by the same camera.
4. **Processing history recovery** tries to recover the processing chain applied to the photo. In this task the features extracted are not necessarily used in malicious processing (e.g. lossy compression, filtering, resizing, contrast/brightness adjustment, etc).
5. **Integrity checking** or forgery detection is a procedure to discover malicious processing, such as removal or addition of objects.
6. **Anomaly Investigation** deals with the explanation of anomalies found in photos that may be a consequence of digital processing or other specific phenomena related to digital cameras.

Forgery detection techniques can be classified as **active** or **passive** [64]. In active detection techniques, prior information about a photo or a video is vital to the detection process, as it concerns data obfuscation where some code is embedded in the photo or video at the time of its generation. It is thus mandatory to verify this code to authenticate the originality of the photo or video file. Active detection techniques are further classified into two forms: **digital watermarking** and **digital signatures**. Digital signatures incorporate some kind of secondary information into the message digest form to ensure authentication, integrity and non-repudiation of the photo or video. The main disadvantage of these approaches are the fact that the information must be inserted into the photos or provided at the time of recording with special equipment, making indispensable some kind of prior information about the photo or video.

In passive detection techniques, no prior information of the photo or video is required. These techniques are based on the assumption that even though that forgery due to tampering may not leave any obvious visual trace, it can change the basic statistics leading to inconsistencies in intrinsic characteristics such as noise, Camera Response Function (CRF) , or Color Filter Array (CFA), are used to detect forgeries.

The goal of photo and video forgery detection techniques is to classify images or videos into legitimate or manipulated. Existing approaches to blind image and video forgery detection, work by extracting their features. After that, a suitable classifier should be trained, by using the specific features extracted from photos or videos datasets, and definitively classifying them as legitimate or manipulated. A generalized framework for blind image detection and video forgery detection [15] consists on the following steps:

1. **Pre-processing:** Several operations are performed on photos and videos before the feature extraction process. Some examples are the conversion of RGB to grayscale, cropping, among others.

2. **Feature extraction:** After the pre-processing phase, a set of features are extracted from each multimedia file. The selected features should be constructed with low dimensionality, which will reduce the computational complexity of training.

3. **Classification:** Selection of appropriate classifiers based on an extracted set of features and choice of a large set of photos or videos to plot the classifiers. In this phase, a set of essential features is obtained from the classifiers, which can be used for classification. Pre-processing the features reduces the feature size, helping reducing the complexity of the computation. A classifier discriminates the given photos or videos, and classifies them into two categories, i.e. authentic or manipulated.

4. **Post-Processing:** Some of the forgery techniques, such as copy-move and splicing, require post-processing operations that involve locating forged regions.

Despite the significant literature available on digital forensics analysis of photos and videos, some unexplored research questions are present due to the peculiarities of video signals compared

to images and the wider range of possible changes that can be applied to this type of digital content [57]. In fact, all the potential manipulations concerning digital photos can be operated on both the individual frames of a video sequence and along the time dimension of the target video. This fact aims at hiding or erasing details of a recorded scene, hiding the source, redistributing the original signal without the owner's permission or faking its characteristics (for example, low-quality content recorded with high quality). Furthermore, forensic analysis of video files proved to be more difficult when compared to still photo analysis, since video data is virtually always available in compressed formats and a high compression factor is often used to store it. Strong compression may cancel or fatally compromise existing footprint ratios so that the processing history is, entirely or in part, no longer recoverable. In addition, forensic analysts must now face the problem of anti-forensic techniques, which consist of modifying the forging process so that the unauthorized modifications are transparent to the forgery detection algorithms.

Digital forensics has gained a growing interest in the criminal ecosystem (e.g. attorneys, prosecutors, trial, criminal police), as the number of cybercrimes and crimes using electronic equipment has raised in the past years. Nowadays, the vast majority of crimes, ranging from the most traditional like murder or assault, to cybercrime, takes advantage of electronic devices connected to the Internet. This shift in the *modus operandi* has direct implications on the increasing number of equipment (e.g. PC, laptops, external storage devices, mobile phones, among others) seized by the police in the scope of a process, and consequently on the methodology adopted to analyze those devices.

Criminal police have been challenged to implement emergent methodologies to accelerate the analysis process, and at the same time, to automatically extract, analyze, and preserve the digital evidence being collected in electronic equipment, namely disks, smartphones, and other devices with storage capacity. These tools embody techniques and procedures to produce a sustained reconstruction of events, to help digital forensic investigators to build a list of evidences that may dictate about suspect's innocence or guilt. The manual analysis by the criminal investigation team is still needed, but oriented towards specific artifacts previously selected by the digital forensics tools and not necessarily in repetitive and tiresome tasks.

The protection of digital forensics information, and preservation of digital evidence, is achieved by establishing strict guidelines and procedures, namely detailed instructions about authorized rights to retrieve digital evidence, how to properly prepare systems for evidence retrieval, where to store any recovered evidence, and how to document these activities to guarantee data authenticity and integrity [62].


Figure 3.1: Overall procedure to extract and analyze electronic devices.

Figure 3.1 depicts the overall procedure to extract and analyze electronic devices, namely those with storage capabilities. The device is plugged into a write blocker like the one presented in Figure 3.3, to prevent any write operation that may be done inadvertently. Then, by using a software to extract a raw image of the storage device, such as Forensic ToolKit (Forensic Toolkit (FTK), https://accessdata.com/, accessed on 8 October 202), a `E01` format image file is produced as pictured in Figure 3.2.

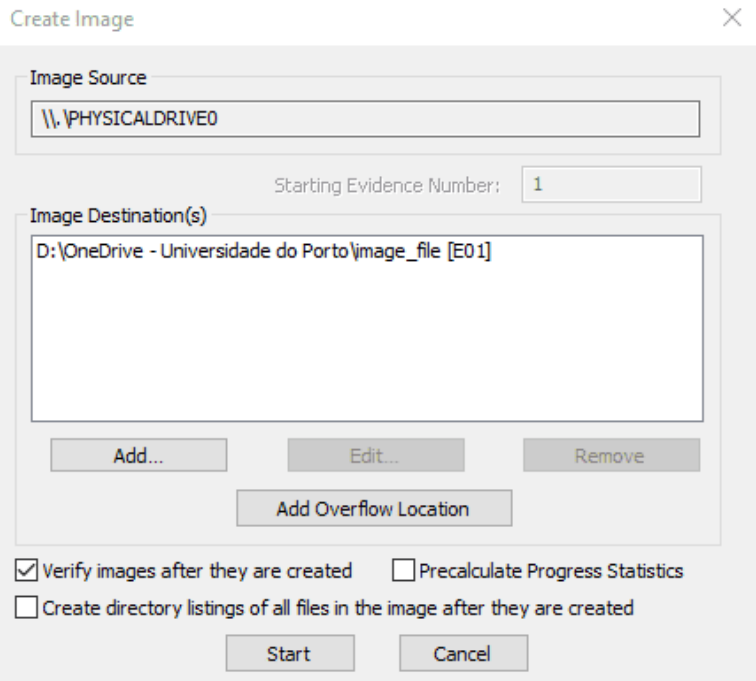



Figure 3.2: Creating E01 image using FTK.


Figure 3.3: Write blocker

Taken as input the `E01` file, as depicted in Figure 3.4, the digital forensic analysis starts, by using appropriate tools, such as those explained in Section 2.2 in this case, Autopsy. The output

produced is a list of artifacts that are worth to the investigation and which digital evidence has to be preserved to be accepted in court.

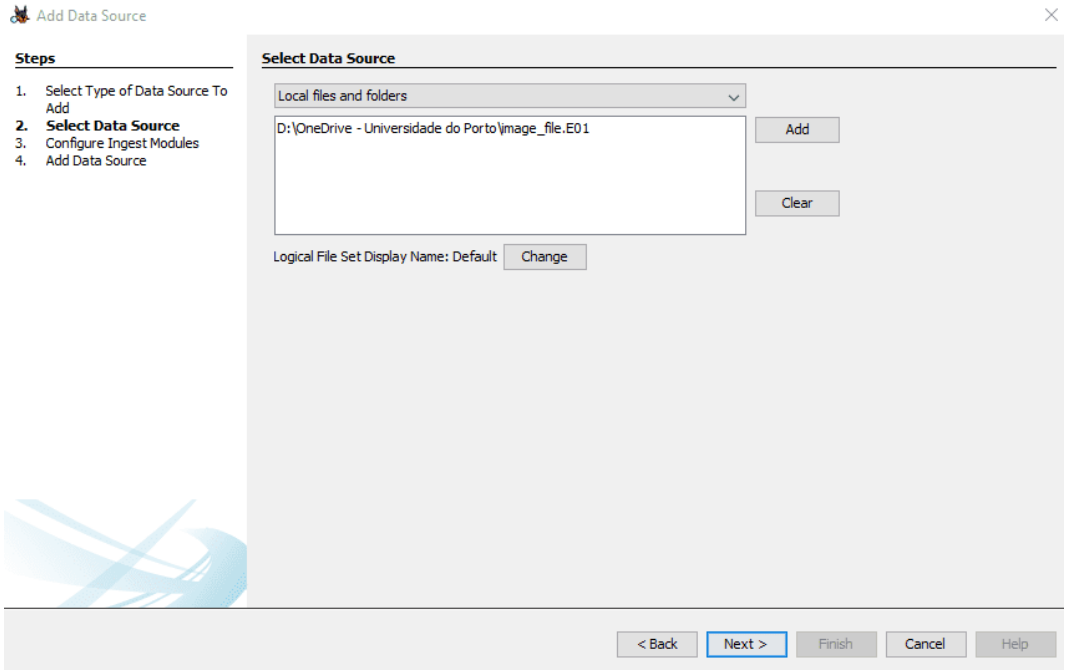



Figure 3.4: Analyzing through Autopsy E01 image created with FTK

The *post-mortem* analysis determines what is needed to be done to produce a full binary image of all the content of the disk that is subject to analysis. Hence, all analysis made to that disk must be done to the binary copy, keeping the digital evidence unchanged. In every device with storage capacity, subject to digital forensic analysis, normally more than one binary copy is made. These copies are know as forensic copies and are meant to be used in data analysis related with the investigation of a case for subsequent probatory recovery. Regardless to the method used to make a forensic copy, the original image or the result copies must be validated by means of a unique and calculated hash [11].

All digital evidences must be validated before and after the recovery. The hash of the original evidence and of all the copies must be the same to maintain integrity and authenticity. The integrity of a forensic copy created for analysis is assured by the hash function that uses an algorithm to calculate the binary sum of the input data, where the output is one sequence of bits of fixed length in hexadecimal. The values produced from the hash function are know by hash sums, checksums or simply hash. The most used algorithms to calculate the hashes are MD5 with 16 bytes key and SHA-1 with 20 bytes key [11]. It is important to note that the same hash algorithm must be used in the original image of the evidence and in the forensic copy as well.

Legally, the hash validation process is called digital signature as stated in point 8 of article 16, about the reception of computer data, in 109/2009 law [28]. The evidence in question may be invalidated by the judge of the criminal investigation as soon as it is seized due to non-compliance with the digital signature during the investigation phase or even during the trial if such incident

is raised by the defense. However, non-compliance is not, by itself, a reason to invalidate the evidence. If a digital signature is technically not possible, the impossibility must be justified in a proper report validated by the Public Prosecutor's Office, ensuring the integrity of the chain of evidence in the case under investigation [11].

In digital forensic analysis of flash memory SSD-based devices, which implement wear levelling algorithms, the digital signature of the total physical storage capacity of the device and the digital signature of its forensic copy are always different. This is due to the continuously changing storage sectors used on the disk, typical made by the leveling algorithms used by this type of device. Thus, there is no ability to validate the integrity of the evidence collected against the original evidence through the digital signature.

## 3.2 Algorithms to detect manipulated multimedia content

Over the years several algorithms have already been proposed for the detection of manipulated content. Some of these proposals will be analyzed below.

### 3.2.1 Digital image forensics technique for copy-move forgery detection using Difference of gaussian (DOG) and Oriented Fast and Rotated Brief (ORB)

This technique is used for detecting manipulated content (with copy-move), and was suggested by Patrick Niyishaka and Chakravarthy Bhagvati of the University of Hyderabad in India [59]. It has three main steps: Border detection with Sobel, feature extraction with DOG and ORB and finally, feature matching. This method combines block-based and keypoint-based detection techniques into a single model.

Block-based techniques have some limitations, namely selecting the block size is difficult, the matching process becomes computationally intensive with small blocks, larger blocks cannot be used to detect small manipulated areas, and uniform regions in the original image will be shown as duplicates. To overcome the above mentioned limitations, it is used blobs instead of photo blocks. A Blob is a group of pixel values that forms a somewhat colony or a large object that is distinguishable from its background, they are basically regions in a photo that differ in properties, such as brightness or color, compared to surrounding regions. Using image processing, it is possible to detect such blobs in photos [23].

For border detection, the Sobel edge detector is used. The resulting photo (Sobel image) like the example in Figure 3.5b) is a 2D gradient map at each point, in which the high gradient areas are visible as white lines. In this proposed method, the resulting Sobel image was the input of the blob detector in order to improve blob localization.

(a) Original image. (b) Sobel image.

Figure 3.5: Sobel edge detector.

Then, the feature extraction process is performed, passing through blob detection and ORB feature detection.

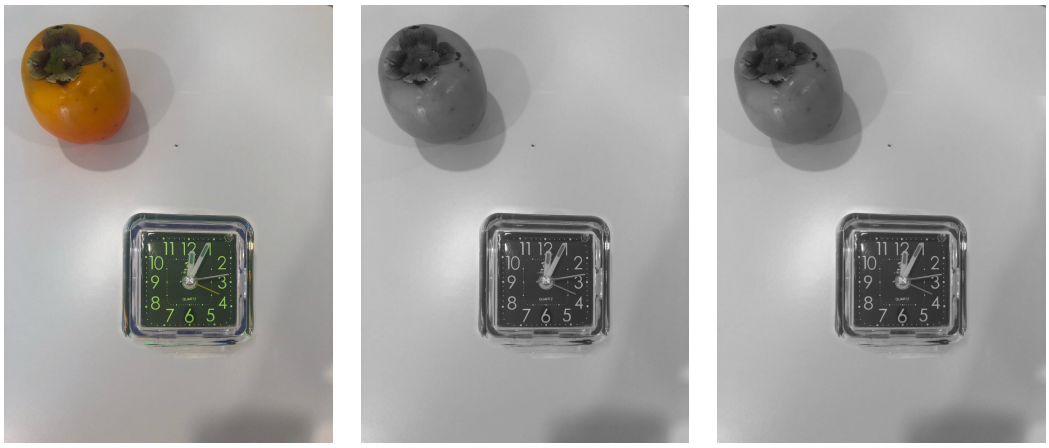
To detect blobs, the DOG method is used [59]. DOG is the difference between two convolved images with two Gaussian filters $g(x, y, \sigma k)$ and $g(x, y, \sigma)$:

$$g(x, y, \sigma) = (1/2\pi\sigma^2) * e^( - x^2 + y^2/2\sigma^2) \tag{3.1}$$

$$DoG = g(x, y, \sigma k) * I(x, y) - -g(x, y, \sigma) * I(x, y) \tag{3.2}$$

where $*$ is a convolution operator, $k$ the scalable variable, $\sigma$ is the standard deviation and $I(x, y)$ represents the photo.

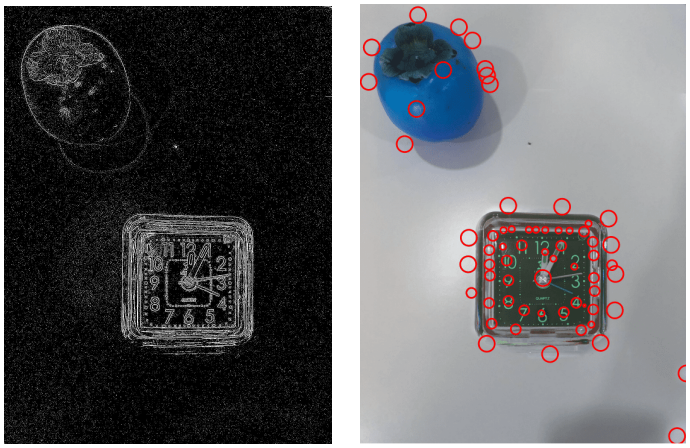
In Figure 3.6 it is depicted the application of DOG.



(a) Original image.

(b) 3x3 Gaussian filter applied to original image.

(c) 5x5 Gaussian filter applied to original image.



(d) Difference between two gaussian filters.

(e) Blob detection.

Figure 3.6: DOG example.

ORB [70] is a fusion of the Features from Accelerated Segment Test (FAST) [82] keypoint detector and Binary Robust Independent Elementary Features (BRIEF) [20] descriptor developed at OpenCV labs by Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski in 2011, as an efficient and viable alternative to Scale Invariant Feature Transform (SIFT) [53] and Speeded-Up Robust Features (SURF) [13]. Initially, to determine the keypoints, ORB uses FAST, and a rotation matrix is computed by using the orientation of patch. BRIEF descriptors are then steered according to the orientation, as depicted in Figure 3.7.

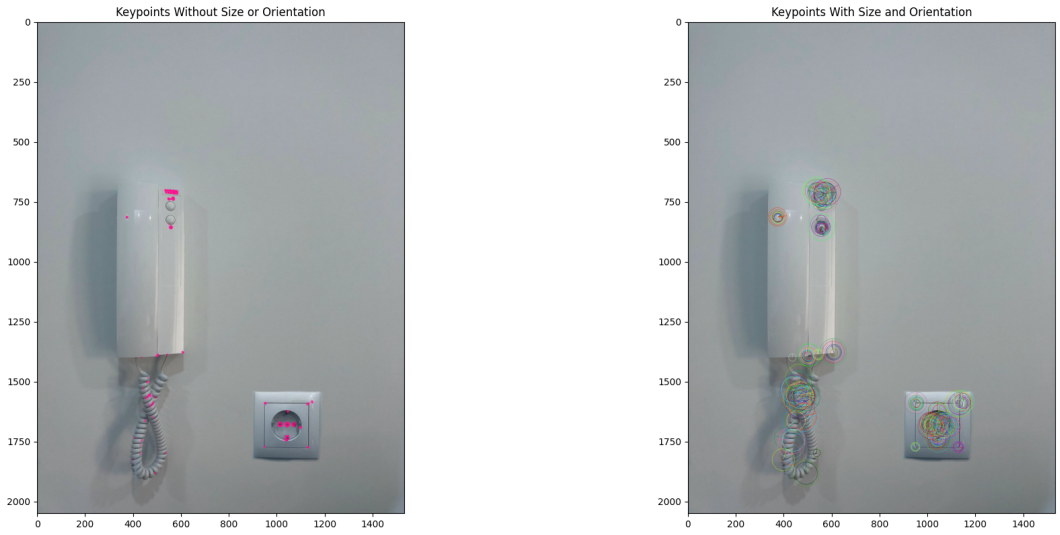
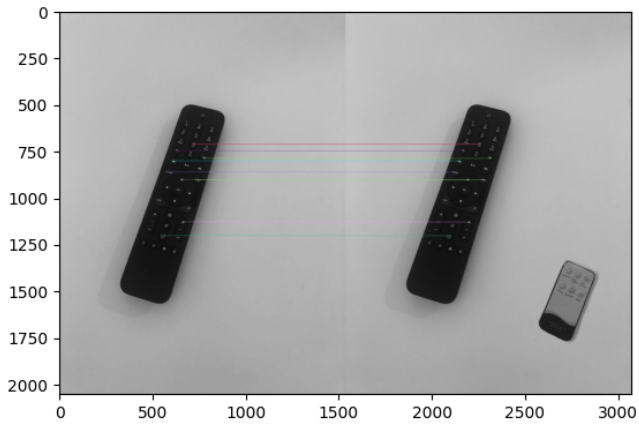



Figure 3.7: ORB feature detector

Feature matching like the one illustrated in Figure 3.8 uses Hamming distance to determine the similarity between two descriptor vectors. The distance between two positions of descriptor vectors of the same size is calculated by the number of positions, where the corresponding symbols are different. Then, the distance is normalized to range between 0 and 1. A match is found between two points of interest if the distance is less than a predetermined threshold.

(a) Original image. (b) Manipulated image.




(c) Feature matching.

Figure 3.8: Matching features between two photos.

In short, using this method, the authors Patrick Niyishaka and Chakravarthy Bhagvati in [59], achieved an average accuracy of 96.47%, a recall of 91.33% and an f1-score of 93.82% detecting copy-move in photos. Their tests were performed on 400 photos (100 original and 300 manipulated photos) chosen randomly from the Comofod dataset (https://www.vcl.fer.hr/comofod/, accessed on 15 November 2020).

### 3.2.2 Unmasking DeepFakes with simple Features

Deepfake is being intensively used in cybercrimes and has became a growing problem with a great impact on today's society. Some algorithms are already documented to deal with this type of manipulation, namely a deepfake detection algorithm proposed by Ricard Durall et al. [32]. This method entitled "Unmasking DeepFakes with simple Features" is based on a classical frequency

domain analysis with Discrete Fourier Transform (DFT), followed by a classification model based on SVM. Figure 3.9 depicts the pipeline of this method.

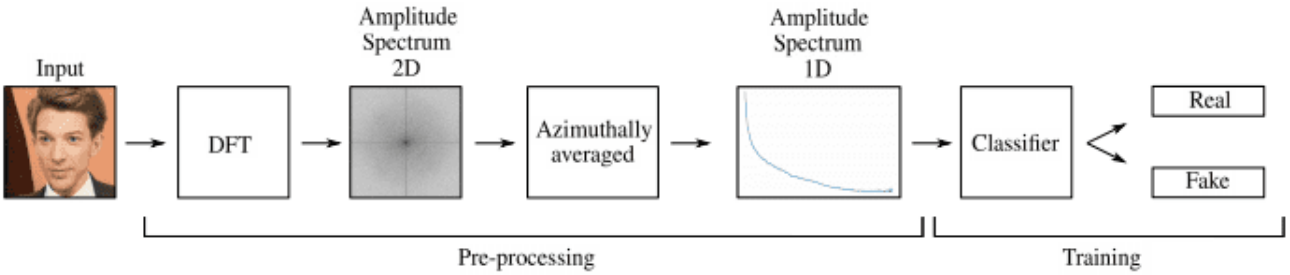


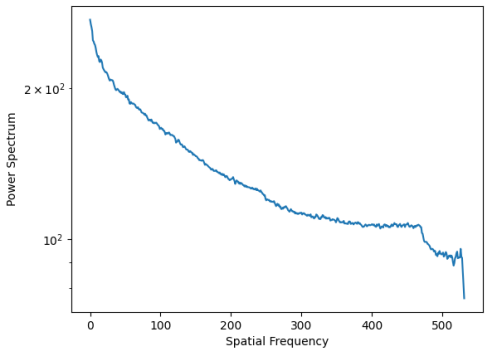

Figure 3.9: Pipeline of the method based on DFT and SVM.

It is possible to analyze the frequency patterns of a photo in a space defined by a Fourier transform, namely a spectral decomposition of the input data, indicating how the signal energy is distributed over a range of frequencies. In this method, a DFT is used, consisting in a mathematical technique to decompose a discrete signal into sinusoidal components of various frequencies ranging from 0 (constant frequency, corresponding to the mean value of the image) to the maximum permissible frequency, given by the spatial resolution. The frequency domain representation of a signal carries information about the amplitude and phase of the signal at each frequency, and can be calculated as in equation 3.3:

$$X_{k,l} = \sum_{n=0}^{N-1} \sum_{0}^{M-1} x_{n,m} \cdot e^{(-\frac{i2\pi}{N} k_n)} \cdot e^{(-\frac{-i2\pi}{M} l_m)} \tag{3.3}$$

After applying a DFT to a photo, the returned values are represented in a new domain but within the same dimensionality. Therefore, since this method is dealing with photos, the output still contains 2D information.

(a) Photo being analyzed.




(b) DFT power spectrum.

```
[1.         0.84224635 0.78570679 0.75473273 0.71715572 0.69231963
 0.67016791 0.65023157 0.63120031 0.61437809 0.59388545 0.58517916
 0.56403071 0.54422546 0.53295873 0.51479493 0.50410594 0.49511568
 0.4845871  0.47045922 0.46812872 0.46119489 0.45625843 0.43764557
 0.43271859 0.41981565 0.41594143 0.41101351 0.41115967 0.40293237
 0.4003339  0.40234623 0.39866122 0.39430526 0.39588078 0.39216254
 0.38928573 0.39037718 0.38334375 0.3876251  0.3903337  0.38917909
 0.38776943 0.38989556 0.36038236 0.34553551 0.34024391 0.33761494
 0.34297509 0.27801506]
```

(c) Features vector

Figure 3.10: Photo features extraction by using DFT.

Azimuthal averaging is then applied to compute a robust 1D representation of the DFT power spectrum (Figure 3.10b). At this point, each frequency component is the radial average of the 2D spectrum (Figure 3.10c). The SVM is subsequently used to create a model based on a training dataset with manipulated and genuine photos. This model will be applied to a test dataset in order to identify an optimal separating hyperplane that maximizes the margin between both classes (manipulated and genuine).

The authors of this method have also developed a new high resolution dataset called Faces-HQ, corresponding to the compilation of 40,000 photos from four different datasets, namely `CelebA-HQ` [45], `Flickr-Faces-HQ` [46] for real photos, and `100K Faces`(https://generated.photos/) and `thispersondoesnotexist`(https://thispersondoesnotexist.com/) for the manipulated ones.

For the sake of testing, the Faces-HQ dataset was divided into two parts: 20% for testing and 80% for training. An accuracy of 100% was obtained on these parameters when using 722 features with regression, and 94% with a K-means classifier.

The architecture developed in this dissertation, which is described in section 4.3, applies the DFT method previously described, keeping in mind the good results obtained.

### 3.2.3 Other methods reviewed

The method described in [44] was also reviewed. This method suggests a deepfake detection using mouth features (DFT-MF) using deep learning and a (CNN) to isolate, analyze, and verify mouth movements. Parts of deepfake videos are extracted using the `moviePy` tool (https://pypi.org/project/moviepy/), taking into account certain word occurrences. From this extraction, using the identified landmarks, the frames in which a person's mouth is closed are removed. In this method, two words per second and five words are a sentence indicator. If the video has more than fifty false frames, it is considered false. Using this method, it was achieved an 71.25% accuracy rate, against the Celeb-df dataset [52].

Castillo and Yang [22] present a comprehensive review of the state-of-the-art deep learning-based models for image forensics, both photos and videos. An exhaustive set of methods are introduced for a set problems, namely median filtering, double JPEG, contrast enhancement, and general-purpose image processing operations. The performance obtained with the methods described, using distinct input features and datasets, is far above the 90% of accuracy.

Yang et al. [86] presents an exhaustive survey of deep learning-based Image Forensics. A wide set of sources are presented, namely source camera identification, recaptured image forensic, computer graphics image forensic, Generative Adversarial Networks (GAN)-generated image detection, and source social network identification. A vast number of detection methods were surveyed, each one using different network architectures and depth. The results are expressive and reveal the good performance obtained by deep learning-based approaches to tackle with image forensics.

Recurrent Neural Networks (RNN) is a type of Neural Network that can have an internal memory to process sequences of inputs along the way in the net. CNN and RNN methods have been fully efficient to deal with the recognition of tampered images and videos [86]. Several authors have applied mixed CNN-RNN based architectures to detect anomalies in videos and recognize facial expression [38, 55, 87].

Despite the significant results attained with deep learning-based methods, simpler feature extraction and classifier methods, like DFT and SVM respectively, can be well integrated into off-the-shelf forensic tools like Autopsy.

## 3.3 SVM

SVM is a supervised learning classifier based on Vapnik's Statistical Learning Theory and Structural Minimization Principle [80]. It is included in a set of kernel-based learning methods, in which the problem is addressed by mapping the data to a larger dimension space. This mapping may not be linear, and the function that allows this mapping is called a kernel [39].

The SVM is a very popular Machine Learning (ML) algorithm, which was firstly proposed as

a binary classification method in 1995 and has been widely used in classification and detection scenarios [81]. This algorithm is based on the principle of statistical learning and aims to minimize the actual error of the problem and not just the error that can be inferred directly from the training examples.

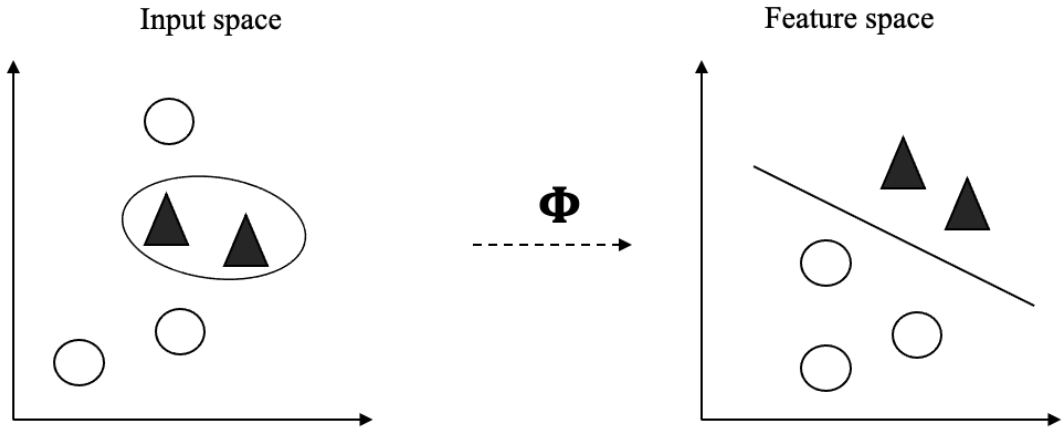



Figure 3.11: Mapping the training data non linearly into a higher-dimensional feature space, and construct a linear separating hyperplane in the feature space.

SVM introduced the concept of kernel method for pattern analysis into ML scenarios [17], in which data is mapped into a high dimensional feature space where each point represents a feature of the input data. This mapping is carried out by using a function $\phi$ described in Equation 3.4, which is denominated by kernel function and where data is mapped into some feature space $F$ via a nonlinear mapping, as depicted in Figure 3.11. Although it does not involve any computations in high-dimensional space, with the use of kernels all computations needed are performed directly in input space [40].

$$\Phi : \mathbf{R}^N \rightarrow F \tag{3.4}$$

Considering that, what it is wanted in this case is a binary classification, that is, with two classes, a positive one (the photo or video being manipulated) and a negative one (the photo or video not being manipulated), the objective of SVM is to find an optimal separation plan between these two classes.

The linear hyperplane (in the feature space) that separates both classes is then selected. It only requires the evaluation of a kernel function and involves only the processing of dot products using the equation 3.5:

$$k(\mathbf{x}, \mathbf{y}) := (\Phi(\mathbf{x}), \Phi(\mathbf{y})). \tag{3.5}$$

Ultimately, the goal of SVM is to find a hyperplane in an N-dimensional space (N being the number of features) that distinctly classifies the points. To separate the two classes of points,

there are several hyperplanes that can be chosen as depicted in Figure 3.12.

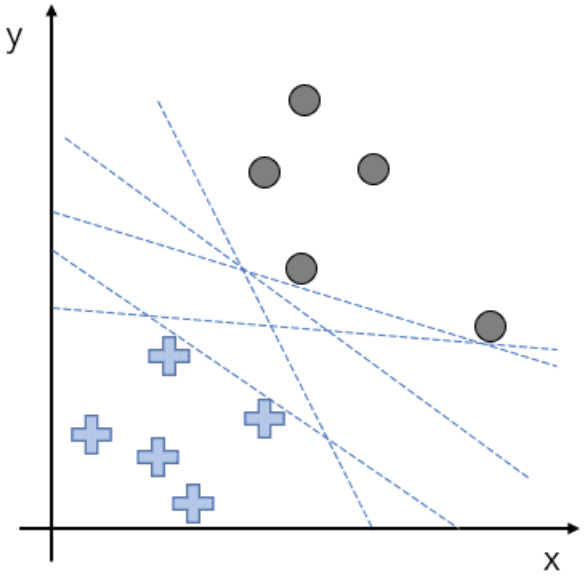



Figure 3.12: Possible hyperplanes

The goal is to find the plane that has the maximum margin i.e. the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence. Intuitively, the margin can be defined as the width of separation between two classes, defined by a hyperplane. Geometrically, the margin corresponds to the smallest distance between the closest data points to any point on the hyperplane.

SVM belongs to a set of methods called kernel-based learning. In this case, the data is mapped into a space with higher-dimensional features. This data mapping may be non-linear and the function that enables this mapping is called a kernel. The Regularization Parameter tells the SVM optimization how much it wants to avoid misclassification of each training example. If $C$ is high, the optimization will choose a smaller margin hyperplane, so the misclassification rate of the training data will be low. In the opposite side, if $C$ is low, then the margin will be large, even if there are classification errors of the training data. The cost is 0 if the predicted value and the current value are of the same sign. If they are not, the loss value is calculated. A smoothing parameter is also added to the cost function, to balance margin maximization and loss. After adding the smoothing parameter, the cost function looks like equation 3.6:

$$min_w \lambda ||w||^2 + \sum_{i=1}^{n} (1 - y_i < x_i, w >) \tag{3.6}$$

SVM finds the most similar examples between classes. These will be the support vectors.

Choosing the right kernel is crucial, because if the transformation is incorrect, then the model

can have very poor results. As a general rule, it is important to always check if the data is linear, and if so, it is important to use linear SVM, as know as *linear kernel*. Linear SVM is a parametric model, but an SVM Radial Basis Function (RBF) kernel, represented by equation 3.7, is not. RBF kernel is more complex comparing with linear kernel, and the complexity level increases with the size of the training set.

$$k(x, y) = exp(-\frac{||x - y||}{\sigma}) \tag{3.7}$$

The next important parameter to consider is $\gamma$, which defines how far the influence of a single training example reaches. This means that the higher $\gamma$ will consider only points close to the plausible hyperplane and the lower $\gamma$ will consider points at a greater distance. Decreasing $\gamma$ will result in more points to be used, when finding the correct hyperplane points at greater distances will be considered.

Since we know *a priori* that all the data that will be used can be linearly separated, it is possible to use a strict-margin SVM to learn to classify. The optimal hyperplane is defined in this case as:

$$< w, x > +b = 0 \tag{3.8}$$

where `w` and `b` are the weights and bias vector respectively.

SVM models have been applied in this dissertation as ML-based classifiers to distinguish between original and tampered photos and videos. In [33] a SVM classifier is used to detect re-sampled images. This method is based on examining normalized energy density present within varying size windows in the second derivative of the frequency domain and exploits this characteristic mentioned to derive a 19-dimensional feature vector that is used to train the SVM classifiers. SVM is also used in the methods described previously in this section.

## 3.4 CNN

CNN, also known as ConvNet, is a deep learning algorithm comprised of neurons that self-optimise through learning. Each neuron receives an input and performs an operation (such as scalar product followed by a non-linear function) [63]. Technically, in CNN each input photo will pass through a series of layers, in order to train and test the model. There are three types of layers: convolutional layers, pooling layers and fully-connected layers.

CNN photo classification takes an input photo, process it and classifies it under certain categories pre-defined like in our case, fake or real. The input photo is seen as an array of pixels and it depends on the photo resolution. Based on the photo resolution, it is observed for each photo `height x width x dimension`.

Convolution layer is the first layer to extract features from an input photo. Convolution preserves the relationship between pixels by learning features using small squares of input data. It is a mathematical operation that takes two inputs such as photo matrix (`h x w x d`) and a filter or kernel. Depending on chosen filter, it is possible to perform operations like edge detection and blur. The output of this layer is called convolved feature or feature map [65].

An additional operation called Relu (rectified linear unit) is usually used after every convolution operation. Relu aims to introduce non-linearity in the CNN. This activation function allows faster and more effective training by mapping negative values to zero and maintaining positive values [73].

The pooling layer simply performs downsampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation. There are two types of pooling: max pooling and average pooling. Using max pooling the maximum value for each patch of the feature map is calculated. In average pooling the average value for each path of the feature map is calculated.

Finally, the feature map matrix is flatten into a vector and that is provided into a fully connected layer, like a neural network. In this layers, features are combined together in order to create a model. An activation function is used to classify the outputs (as fake or real in this case). It is also suggested that Relu may be used between these layers, to improve performance.

## 3.5 Summary

In this chapter a description of the functioning of digital forensics investigation was made, emphasizing the main tasks. It was also depicted the most relevant methods studied to detect multimedia manipulations. DFT-SVM-based method was described, as it is used in the application developed in this dissertation. An explanation about SVM and CNN was also made in this chapter. The Autopsy tool, the methodology used to develop new modules for this tool and the architecture followed in the development of this dissertation will be described in Chapter 4. The architecture of the CNN-based method developed for benchmarking is described in Section 5.4.

# Chapter 4

# Development

This Chapter describes the overall procedure to do a forensics analysis through Autopsy, namely by describing the types of modules that can be used to execute different analysis, and how the new modules can be developed by the community. The overall architecture of the modules and the outcome produced are also detailed in this chapter.

## 4.1 Digital forensics analysis using Autopsy

The investigation process in the Autopsy is organized by *cases*, as explained in Section 2.2, which can be executed by several investigators simultaneously. Each investigator is configured to have its own time zone setting and clock bias, so that the timestamp shown is the same as the original user would have seen. Each case can have one or more file system images to analyze. The processing of a case in Autopsy follows the initial steps depicted in Figure 4.1.

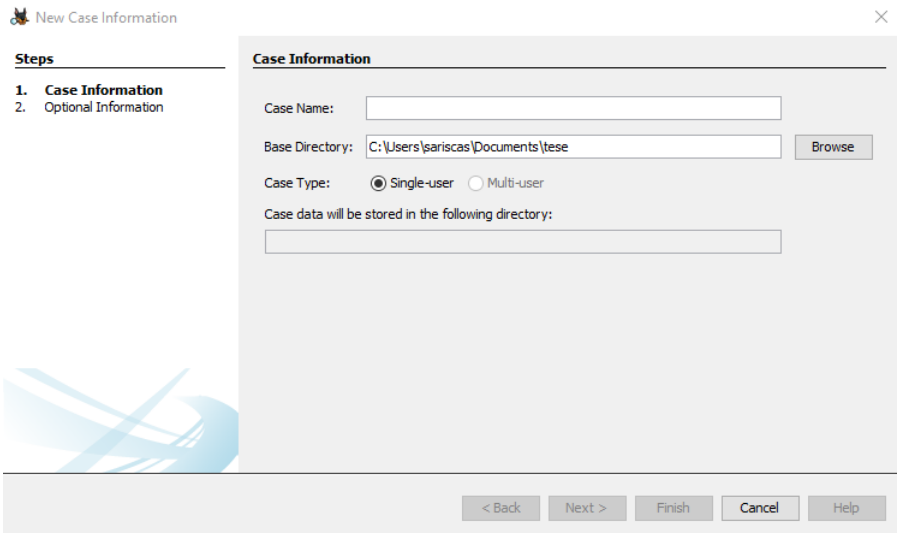



Figure 4.1: Creating new case

Once a new case is created, the next step is to add a data source to be the target of analysis as depicted in Figure 4.2. It is possible to add several data sources types, such as an image of a disk, a virtual machine, folders or even directly from the XRY tool, which, as explained in Section 2.2, extracts images from a storage device (for example, a hard disk).

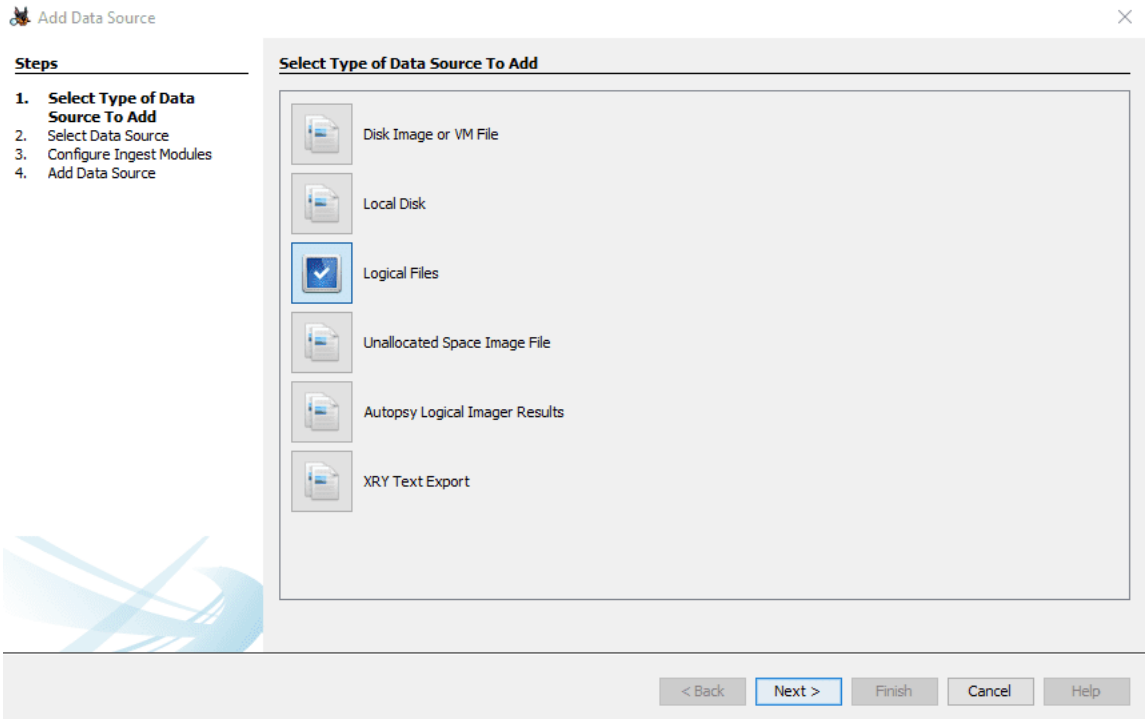
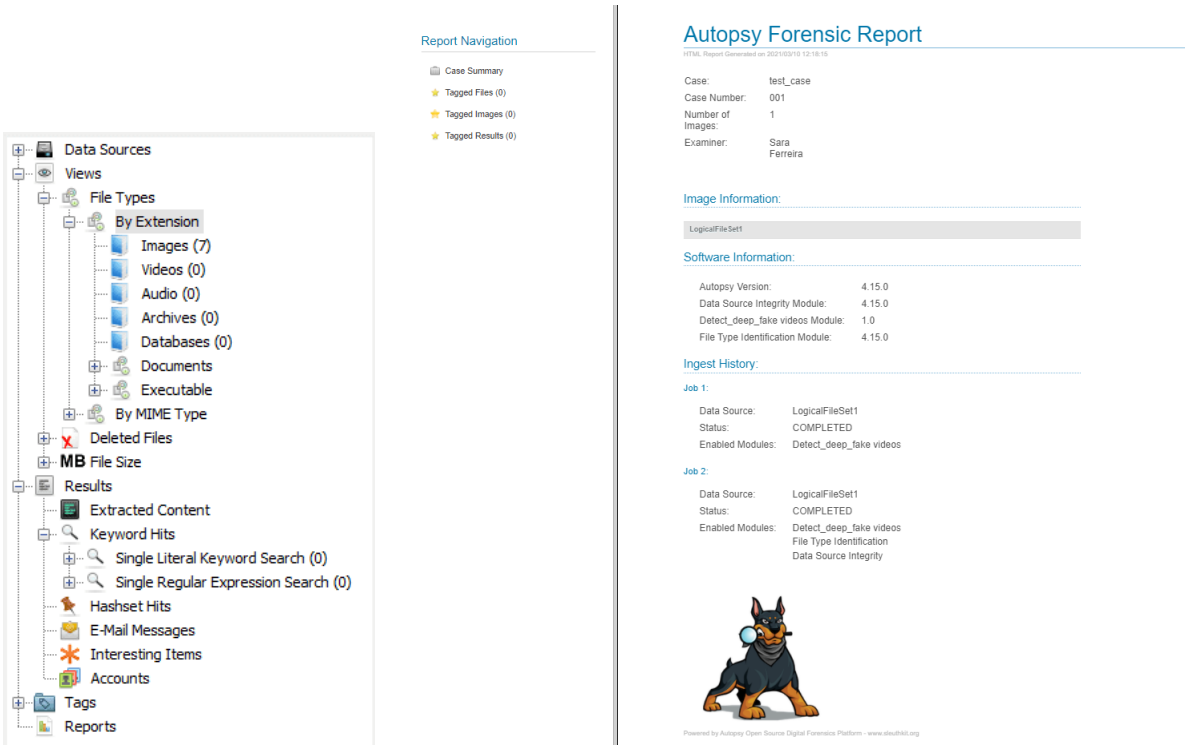



Figure 4.2: Adding data source to case

Finally, when the data source is added correctly, it is necessary to choose one or more modules to proceed to the analysis. The analysis processing obtained with the Autopsy produce two distinct outcomes, as depicted on Figure 4.3. On the left side (Figure 4.3(a)) the resulting interesting artifacts are shown (Figure 4.3(a)), while on the right side Figure 4.3(b)) reports the most relevant information about the case. The report module was used according to the explanation presented in Section 2.2.




(a) Found Artifacts (b) HTML report generated

Figure 4.3: Analysis results

By default, Autopsy comes with an initial set of modules, to be used as represented in Figure 2.3. Some of the most commonly and widely used Autopsy modules are `File Type Identification`, `Data Source Integrity`, and `Android Analyzer`.

**File Type Identification** [3] module identifies the input files based on their internal signatures and not depending on their extensions. Autopsy uses the Tika library (http://tika.apache.org/) to do its primary file type detection and that can be customized with user-defined rules. It is not necessary to configure anything before using this module, unless it is necessary to define custom types for a search of a specific file type. To define a specific type it is necessary to go to the "Tools" menu and, from there, rules can be defined, based on the offset of the signature and whether the signature is a sequence of bytes of an ASCII string.

**Data Source Integrity** [2] module has two main goals. If the data source has any associated hashes with it (either entered by the user or contained in an E01 file), it will check those hashes. If the data source has no hashes associated with it, the module will calculate the hashes and store them in the internal database, in order to check the data source integrity.

**Android Analyzer module** [1] allows to parse SQLite and other files from an Android device. Autopsy processes the physical dumps of the most commonly used Android devices however, will not support older ones, which do not have a volume system. These devices often have a single physical image file for themselves, and there is no information in the image that describes the layout of the file systems. In these cases, Autopsy will not be able to detect the file

system and process it. This module is able to extract a wide variety of data, being some of them described below:

- Text messages / SMS / MMS
- Call logs
- Contacts
- GPS from browser and Google Maps
- GPS from *cache.wifi* and *cache.cell* files

## 4.2 Example module development

The main reason for considering writing a module for Autopsy rather than a standalone tool is that Autopsy will handle with the various data inputs and display the results to the user in its graphical interface. This is an advantage since the large majority of forensic investigators are expert with several forensic tools, but have no prior knowledge on how to automate analysis tasks with programming languages.

Autopsy is written in Java and therefore all the modules described above can be developed in that language. It is also possible to write some of the modules in Python (Ingest Modules and Report Modules) and integrate them directly in the Autopsy.

Setting up the development in Python is easier than in Java so, Python was the programming language chosen for this project. However, Autopsy uses Jython (https://www.jython.org/, accessed on 8 November 2020) to enable scripting in Python. Jython is just like Python, but the scripts are converted to Java byte code and run on the Java virtual machine (JVM). This development setup has however some limitations:

- It is limited to Python version 2.7.
- Python libraries that use native code cannot be used.

When using Jython, we are limited to Python 2.7 but currently many features of libraries in Python, like OpenCV (https://opencv.org/, accessed on 8 November 2020) are deprecated for this version and therefore it is necessary to integrate Python 3.x for module development. The solution to overcome this limitation is to use Python executables in the Autopsy module. Doing so, with Jython (using Python 2.7) it is possible to run scripts developed for Python 3.x.

To create Python executables, it is possible to use the `pyinstaller` package (https://pypi.org/project/pyinstaller/, acessed on 5 December 2020) which bundles a Python application and all its dependencies into a single package. Using this, the user can run the packaged application without installing the Python interpreter or any modules.

All modules that are developed for Autopsy must be placed in a specific folder, which is defined in the Autopsy menu under the option `Tools -> Python Plugins`.

First, an example module has been developed in order to understand how this development process works. The first step is to create a new module folder. For that, `Tools -> Python Plugins` menu item was accessed in Autopsy and a folder with the name of the module was created.

The next step is to write the Python script corresponding to the analysis that the module is meant to perform. Regardless of the type of ingest module that is being developed, two classes are essential:

1. The **Factory class** provides Autopsy module information, such as display name and version. It also creates the instances of ingest modules as needed.
2. The **Ingest module class** incorporates the actual analysis process being developed.

The example presented below corresponds to a file ingest module in order to find text files. The development presented here was focused on four main parts: module details, module analysis, creating new artifacts, and the Python executables usage.

**Module Details** : In order to start developing a new Autopsy module, it is needed to create the factory class. In this class it is possible to define an unique name and description to present the module to Autopsy users. It is also defined if the module is going to be a file ingest or a data source module type. Listing 4.1 shows factory class related to the development of an example file ingest module.

Listing 4.1: Factory class of example autopsy module

```
class ExampleIngestModuleFactory(IngestModuleFactoryAdapter):

    #Unique name that is going to be shown in module list in Autopsy
    moduleName = "Example txt finder module"

    def getModuleDisplayName(self):
        return self.moduleName

    #Description of module
    def getModuleDescription(self):
        return "Example module to find text files"

    def getModuleVersionNumber(self):
        return "1.0"

    # Return true if module wants to get called for each file
    def isFileIngestModuleFactory(self):
        return True

    def createFileIngestModule(self, ingestOptions):
        return ExampleIngestModule()
```

**Module analysis:** The class described in Listing 4.2 has four main methods, namely `log`, `startUp`, `process`, and `shutDown`. These methods enable the possibility to present to the user, in the Autopsy interface, the running status of the module, to apply some pre-processing tasks needed for the module analysis (setup and configuration), to define the analysis methodology, and to free resources. The ingest module used as example, does the following two tasks: 1) find text files in a data source, 2) post the files found in the Autopsy blackboard.

Listing 4.2: Ingest module class of example autopsy module

```
class ExampleIngestModule(FileIngestModule):

    _logger = Logger.getLogger(ExampleIngestModuleFactory.moduleName)

    #Shows the user that the module is running
    def log(self, level, msg):
        self._logger.logp(level, self.__class__.__name__, inspect.stack()[1][3], msg)

    #Where any configuration is done
    def startUp(self, context):
        self.filesFound = 0

    #Where the analysis of the module is done.
    #Since it is a file ingest module, each file will be passed into here.
    def process(self, file):

        blackboard = Case.getCurrentCase().getServices().getBlackboard()

        # Flag files with .txt in the name and make a blackboard artifact.
        if file.getName().lower().endswith(".txt"):

            self.log(Level.INFO, "Found a text file: " + file.getName())
            self.filesFound+=1

            # Make an artifact on the blackboard. TSK_INTERESTING_FILE_HIT is a generic type
            #of artifact.
            art = file.newArtifact(BlackboardArtifact.ARTIFACT_TYPE.TSK_INTERESTING_FILE_HIT)
            att = BlackboardAttribute(BlackboardAttribute.ATTRIBUTE_TYPE.TSK_SET_NAME,
                  SampleJythonFileIngestModuleFactory.moduleName, "Text Files")
            art.addAttribute(att)

            try:
                # index the artifact
                blackboard.indexArtifact(art)
            except Blackboard.BlackboardException as e:
                self.log(Level.SEVERE, "Error indexing artifact " + art.getDisplayName())

            # Fire an event to notify the UI and others that there is a new artifact
            IngestServices.getInstance().fireModuleDataEvent(
                ModuleDataEvent(SampleJythonFileIngestModuleFactory.moduleName,
                    BlackboardArtifact.ARTIFACT_TYPE.TSK_INTERESTING_FILE_HIT, None))

        return IngestModule.ProcessResult.OK


    def shutDown(self):
        # Send a message to the ingest inbox with the number of files found (in this thread)
        message = IngestMessage.createMessage(
            IngestMessage.MessageType.DATA, SampleJythonFileIngestModuleFactory.moduleName,
                str(self.filesFound) + " files found")
        ingestServices = IngestServices.getInstance().postMessage(message)
```

**Creating new artifacts:** Although Autopsy has defined built-in types for artifacts, such as the type used in Listing 4.2 (`ARTIFACT_TYPE.TSK_INTERESTING_FILE_HIT`), it is possible to create new types. Once a new artifact type is created, a new attribute type can be added to associate it with the new artifact. Then simply create an attribute with the new type and add content to it to later add the attribute to the artifact and post it to the blackboard. The process of creating a new artifact can be divided into the following six steps:

1. Create new artifact type;
2. Create new artifact;
3. Create new attribute type;
4. Create new attribute;
5. Add content to attribute (for example, a string or an integer);
6. Associate the attribute with the artifact.

In Listing 4.3 a new artifact type named `TSK_DEEP_FAKE_DETECT`, and a new attribute type `TSK_DEEP_FAKE_SCORE`, are created to hold the probability of a photo or video have been manipulated. This artifact will be used in the modules developed in the scope of this dissertation and detailed in this Chapter.

Listing 4.3: New artifact and attribute types

```
1
    # Use blackboard class to index blackboard artifacts
3   blackboard = Case.getCurrentCase().getServices().getBlackboard()

5   #create artifact
    artId = blackboard.getOrAddArtifactType("TSK_DEEP_FAKE_DETECT", "Deep fake Detections")
7   artifact = dataSource.newArtifact(artId.getTypeID())

9   #create attribute
    attId = blackboard.getOrAddAttributeType("TSK_DEEP_FAKE_SCORE",
11  BlackboardAttribute.TSK_BLACKBOARD_ATTRIBUTE_VALUE_TYPE.STRING, "Probability of
    being manipulated")
13
    count_fake=0
15          for pred in predictions:
                if str(pred)=="0":
17                  count_fake=count_fake+1

19   #If a third of the frames are fake, probably fake
            if count_fake>=(len(predictions)//3):
21              artifact_content= file+" is probably fake"
            else:
23              artifact_content= file+" is probably real"

25    #Add data to atribute
      atribute=BlackboardAttribute(attId,
27    SampleJythonDataSourceIngestModuleFactory.moduleName,
      artifact_content)
29
      try:
```

```
31          #Add atribute to artifact
                artifact.addAttribute(atribute)
33          except:
                    self.log(Level.INFO, "Error adding attribute to artifact")
35
                try:
37                  #Post artifact in the blackboard
                    blackboard.postArtifact(artifact,
39                  SampleJythonDataSourceIngestModuleFactory.moduleName)
                except:
41                  self.log(Level.INFO, "Error indexing artifact")
```

**Using Python executable:** As mentioned earlier in this chapter, it is not possible to use Python 3.$x$ in module development for Autopsy. Hence, to include features that require this version of Python it is necessary to use executables. Listing 4.4 illustrates an example of how to use one of these executables in a module for Autopsy.

Listing 4.4: Use python executables

```
2   #path to python executable
    path_to_exe=str(os.path.dirname(os.path.abspath(__file__))).replace('/',"//")+
4   "//dist//video_to_image//video_to_image.exe"

6   #Pass arguments if needed
    pipe = Popen([path_to_exe, file, temp_dir, temp_dir2, str(seed)],stdout=PIPE)
8
    #Output as string
10  out_text = pipe.communicate()[0]
    self.log(Level.INFO, "Output from run is ==> " + out_text)
```

## 4.3 Photo and video manipulations detector

### 4.3.1 Background

As the wide majority of the forensic investigators do not have programming background, and in order to get faster results in the course of a forensic investigation, it is important to have tools that can automate forensics analysis. As mentioned in Section 4.1, Autopsy allows the creation of modules by the community, which become available for all the practitioners. This ability to develop modules is very important as it is possible to create specific modules for certain not built-in task available on Autopsy. These new modules bring new capabilities to enhance the automation, fully or partially, of the criminal investigation and analysis process.

In a digital forensic investigation involving the analysis of photos and videos, it is common to have a large amount of multimedia content to evaluate their authenticity. In some crimes, it is also relevant to evaluate if the multimedia content has been tampered and digitally manipulated. The manual analysis becomes an extremely extensive and time-consuming task for the investigator

and, moreover, in some cases it is not possible to detect some digital manipulation with the naked eye.

Currently, there is no tool that automates the whole process of media manipulation analysis. In the scope of this dissertation two modules for Autopsy were developed, to detect manipulated photos and videos. Autopsy was chosen as it is the most open-source and free forensic tool used worldwide, and has the availability to accommodate third-party developed modules. Initially, it was developed a module to detect manipulated photos, and afterwards this module was the source of inspiration to develop a new one, to evaluate the probability of an input video have been subjected to some kind of manipulation. Using these modules it is possible for a forensic investigator, in just one step, to find out the probability that a photo or video have been manipulated without having to go through the extensive process of analyzing photos or videos manually, thus decreasing the time it takes to solve the investigation.

### 4.3.2 Architecture

There are two types of ingest modules as described in Section 2.2, namely File Ingest Modules and Data Source Ingest Modules. To implement the multimedia content analysis, it was decided to develop a Data Source ingest module, due to the flexibility to analyze the data source all at once. First, a standalone application was developed to evaluate the appropriateness of using Support Vector Machines (SVM)-based method to detect fake photos and videos. By adopting this strategy, it was easier to apply the SVM method without having to deal with the specific formatting inherent to the development of modules for Autopsy.

When the SVM method was up and running in the standalone application it was necessary to evaluate how to use it in a module for Autopsy. It was necessary to create two executables to run the main functions in the module, due to the limitation of using Python 2.7 and this feature need a later version of Python. One executable to pre-process the photos and videos and another to create the SVM model and classify the data. These executables will then be used in the development of the modules for Autopsy as described below, in this section.

Firstly, a new module was developed only to detect manipulated photos. The results were promising and it was decided to apply the same method to frames extracted from videos. Therefore, the standalone application and the two modules developed for Autopsy follow the same overall architecture depicted in Figure 4.4. The Autopsy modules are optimized for Autopsy version 4.15.0 and the executables were developed in Python version 3.9. The experiments were carried on in a computer with Windows 10, 8GB RAM and AMD Ryzen 52600.

Depending on the module chosen for the analysis, autopsy starts by looking for photos in the data source target of analysis looking for files with the extension `.JPG` or looking for videos, in this case, all files with the extension `.MP4`. Only files with the extension `.JPG` or `.MP4` are searched in order to simplify the process, being these the extensions most commonly used for the type of files it is intended to find using the modules developed. Those files that were found, were

stored in a temporary folder relative to the case in Autopsy. From now on, the two modules have three main building blocks, as illustrated in Figure 4.4: pre-processing, processing and results.

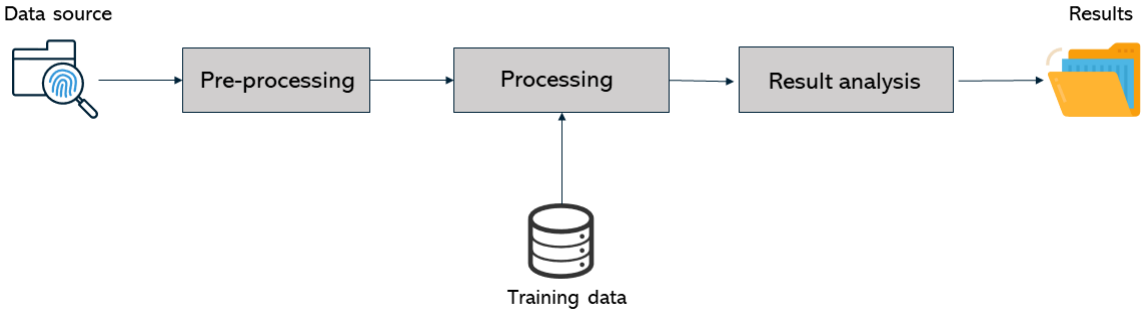



Figure 4.4: Overall architecture of the standalone application and Autopsy modules.

Pre-processing phase corresponds to the data processing of the photos and videos files that were found by Autopsy to create the test dataset. The next step is the processing phase, where the SVM model is created and where the test dataset is going to be classified. Using the results acquired from processing phase, the results analysis phase produces a final report with the classification made, for each file that is analysed.

**Pre-processing** In order to obtain a functional manipulation detection system using the method of Discrete Fourier Transform (DFT) and SVM described in [32], it is necessary in a first step, to obtain the data to provide to the classification model, which in the end, will classify each photo as manipulated or legitimate. To achieve this, it is necessary to do some pre-processing first as depicted in Figure 4.5.

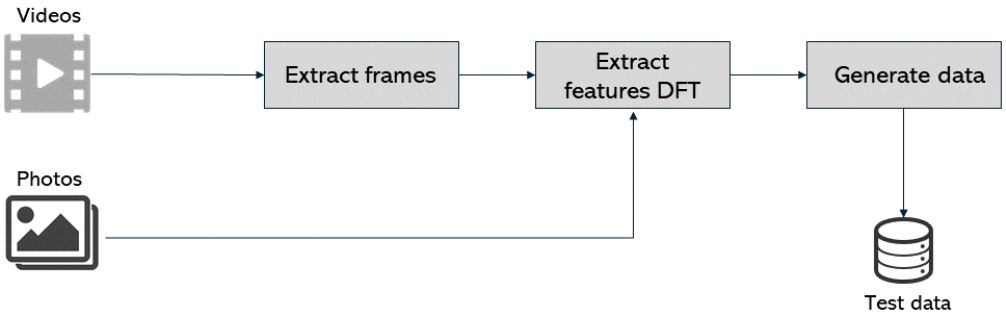



Figure 4.5: Pre-processing phase.

In the case of an analysis using the manipulated photos detection module, this pre-processing phase essentially corresponds to extracting features from the photos found in the data source and creating labeled data for processing, which will be used as the test dataset. In the case of analysing the target data source using the manipulated video detection module, initially the

frames are extracted from the found videos in order to apply the feature extraction method, as with the photos files. To extract frames from videos, a Python script was developed as shown in Listing 4.5, to extract between three to four frames per second from the videos found to be processed by Autopsy.

Listing 4.5: Script to extract frames from videos.

```
import cv2
import time
import os

def video_to_frames(input_loc, output_loc,filename,seed):
    try:
        os.mkdir(output_loc)
    except OSError:
        pass

    time_start = time.time()
    count_total = 0
    count_file=0

    for file in os.listdir(input_loc):
        if file==filename:
            count=0
            sum_fps=0
            print(file)

            cap = cv2.VideoCapture(input_loc+"\\"+file)

            video_length = int(cap.get(cv2.CAP_PROP_FRAME_COUNT)) - 1
            fps= int(cap.get(cv2.CAP_PROP_FPS))
            print ("Frames per second:", fps)
            print("Duration of video:", int(video_length/fps))


            print ("Converting video..\n")

            while cap.isOpened():

                ret, frame = cap.read()

                cv2.imwrite(output_loc + "/"+seed+"%#\05d.jpg" % (count_file+1), frame)
                sum_fps= sum_fps+1
                count = count + fps/3
                count_total = count_total + 1
                count_file = count_file + 1
                cap.set(1, count)

                if (count > (video_length-1)):

                    cap.release()
                    break
            time_end = time.time()
    print ("Done extracting frames.\n%d frames extracted in total" % count_total)
    print ("It took %d seconds for conversion." % (time_end-time_start))

if __name__=="__main__":
    if(len(sys.argv) != 5):
        print("Not enough arguments")
        print("insert filename, input dir, output dir and seed")
        exit()

    input_dir=sys.argv[2]
    output_dir=sys.argv[3]
```

```
58      filename= sys.argv[1]
        seed= sys.argv[4]
60      if os.path.isdir(input_dir) is False:
            print("this input directory does not exist")
62          exit(0)

64      if os.path.isdir(output_dir) is False:
            print("this output directory does not exist")
66          exit(0)

68      video_to_frames(input_dir, output_dir,filename,seed)
```

Therefore, the module extracts features from all photos or video frames found in the data source (already saved in a temporary folder), applying DFT method. The DFT method described in Section 3.2.2 for feature extraction was chosen since it was recently published and have produced promising results using simple features [32].

The extracted features are used to produce the labeled input dataset for classification (in processing phase) using the SVM model. The pre-processing phase reads the photos or video frames previously saved in temporary folder of Autopsy case, through the `OpenCV` library and extracts exactly fifty features, that are loaded into a new file with the corresponding label. Since at the beginning of the analysis it is unknown if a photo or a video frame is manipulated or legitimate, all files will be labeled with 0. Accordingly, it is presumed that all photos or frames are manipulated until proven otherwise.

At the end of pre-processing phase, a fully labeled dataset to feed to the SVM model is created. To serialize all the data obtained from the pre-processing, `Pickle` library is used. All data used to train and test the classification model has the `.pkl` extension. The dataset and all the scripts related to the modules that were developed, are available at the following Github repository: https://github.com/saraferreirascf/Photo-and-video-manipulations-detector.

**Processing** The processing phase corresponds to the SVM processing. SVM is included in a set of kernel-based learning methods as explained in Section 3.3, in which the problem is addressed by mapping the data to a larger dimension space. This phase comprehends three main parts as depicted in Figure 4.6, namely creation of SVM model, fitting training data, and classifying test data.

Firstly, the following parameters were chosen, to create the SVM model: The Radial Basis Function (RBF) kernel and a regularization parameter of 3. This choice took into account what the authors of the method [32] used and some experiments that were done with other parameters, which produce unsatisfactory results. The implementation of SVM processing was made through `scikit-learn` library for Python 3.9.

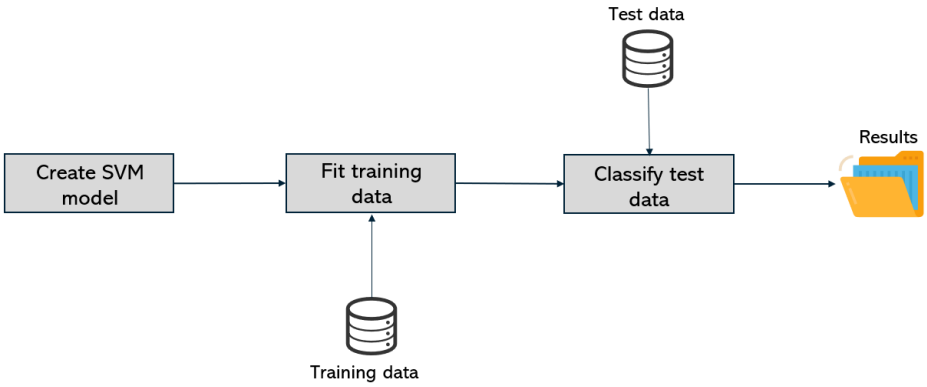



Figure 4.6: Processing phase.

In order to have a correct classification of the test data, the SVM model needs to be trained. A training file was created *a priori* and distributed throughout with the modules. The training file was created from the dataset described in 5.2 using the same pre-processing as test dataset to extract features. All photos and frames presented in the created dataset were labeled with 0 if the examples corresponds to a manipulated photo, and with 1 if it is a legitimate photo.

With the SVM model created, the test dataset generated in pre-processing phase and the training dataset sent to the modules, the module is ready to to classify the test data and then the results phase follows.

**Results** In this phase the module evaluates the results obtained within the processing phase, to give a final classification of the file being analyzed, as depicted in Figure 4.7.

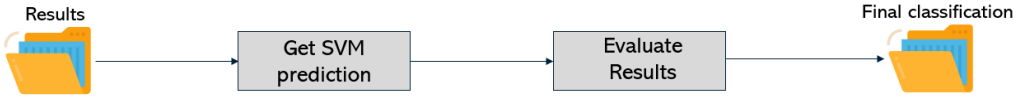



Figure 4.7: Results phase.

The SVM model outputs the predictions obtained in the processing phase. With that predictions it is possible to assess the probability that a photo or video has been manipulated or not. In the case that a video is being evaluated, instead of a photo, the video is considered to be fake if at least one third of their frames are considered fake. One third was the value chosen because it seems reasonable that if one third of the video has fake frames, it is possibly fake. For each photo or video evaluated, an artifact is created with the name of the file and the probability of beeing manipulated. When the analysis is over, all artifacts are posted in the case's blackboard, so the criminal investigator can have access to the results.

**Limitations** Some limitations regarding the modules development can be pointed out. Python method `popen()` (https://docs.python.org/3/library/subprocess.html#subprocess.Popen) used in the executables, opens a pipe to or from a command. Because of that, all arguments are passed in the command line and all the output produced by the executables comes from command line too. This arises to the following limitations: the output produced from the executables needs to be treated as strings, and since the maximum length of the string that it is possible to use at the command prompt is $8,191$, if it is needed to pass an argument with a size bigger than that, it is not possible using `popen` method. This is a problem mainly because when dealing with video files, several frames are extracted and, from these, 50 features are also extracted. That is, instead of analyzing just 50 features like on processing photos, 50 times the number of frames present in the video are analyzed, which generates an output that is too large for the command prompt. In this stage only photos with `.JPG` extension and videos with `.MP4` extension were searched and processed, to simplify the process. Photos and videos files with other extensions, like `.PNG` or `.GIF` go unnoticed.

### 4.3.3 Case study

To start an analysis, demonstrating the functioning of the developed modules, it is necessary to create a new case within Autopsy and to attach a data source to it. An example of a data source is a disk image as explained in Section 2.2. In this case a folder with several files as pictured in Figure 4.8 was added. The intention here is to detect the existence of any manipulated video in this folder.

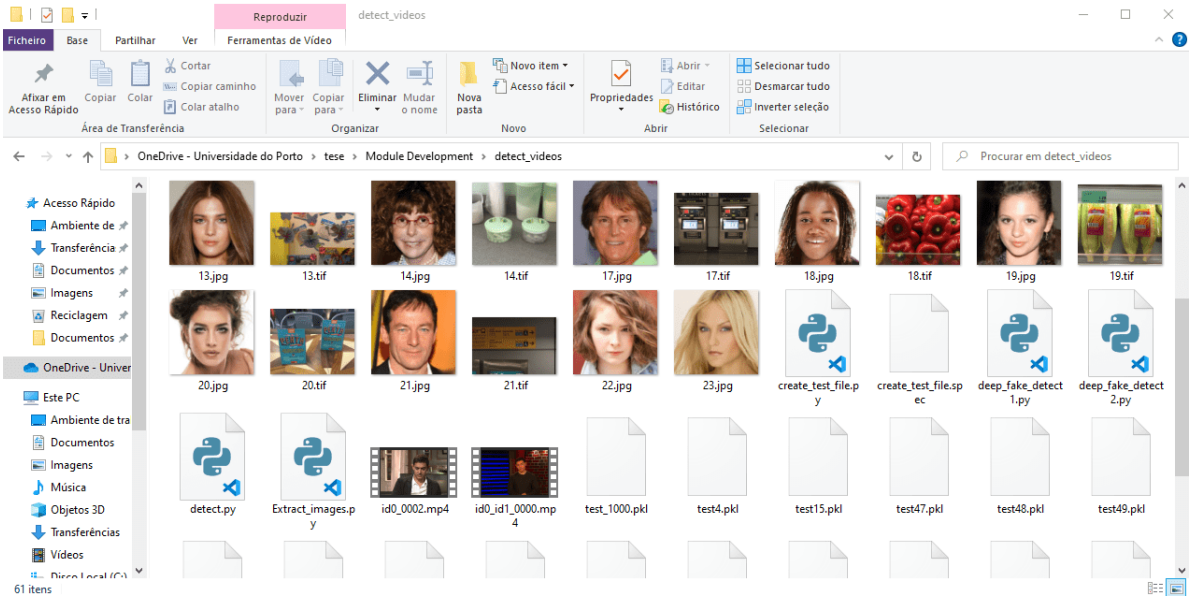



Figure 4.8: Target folder to module analysis.

After the data source has been added to Autopsy, it is necessary to choose which module is going to be used in the analysis of that data source. Since the goal here is to find manipulated

videos, the module for manipulated video detection is chosen and the analysis starts right away with the pre-processing phase.

In this phase, two `.MP4` files corresponding to two videos, were found. An artifact for each file found was created and posted in the user's Autopsy blackboard, as noticed on Figure 4.9

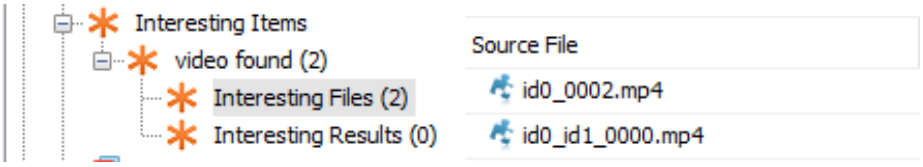



Figure 4.9: Videos found in data source.

For each one of the interesting files that were found, the module begins the processing phase and classifies the extracted features from each file. With the results obtained by the classification of each file, an artifact is created and posted in the blackboard as pictured in Figure 4.10.

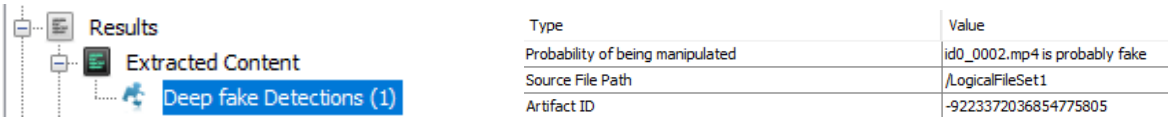



Figure 4.10: Artifacts with final classification.

## 4.4 Summary

As demonstrated in this Chapter it is possible to create modules for Autopsy in order to achieve more specific and customized analyses. It was achieved the automation of the detection of manipulated photos and videos through two Autopsy modules as defined in the goals of this dissertation. Using these modules, a forensics investigator can analyze digital evidence more effectively because he is able to focus only on those photos or videos that have a higher probability of have been manipulated, since analyzing the entire content can be an extensive and time-consuming process and often impossible for the human eye. These modules are available at https://github.com/saraferreirascf/Photo-and-video-manipulations-detector.

# Chapter 5

# Results and analysis

In this Chapter, all metrics used in the evaluation of the Support Vector Machines (SVM)-based method will be described, as well as the results obtained and the corresponding analysis. The effectiveness of the method developed is going to evaluated by a ten-fold cross validation in the dataset described in 5.2 and the method implemented will be compared with a Convolution Neural Networks (CNN) approach.

## 5.1 Evaluation metrics

In order to have a correct evaluation of the classification method, it is necessary to discuss the evaluation metrics. The metrics used to evaluate the results were Precision (P), Recall (R) and F1-score, which can be calculated through the well known and documented confusion matrix [71] represented in Table 5.1. It was also used Receiver Operating Characteristic (ROC) curves to graphically identify the performance obtained by the classifier.

Table 5.1: Confusion matrix

| | Positive | Negative |
|---|---|---|
| Positive | TP | FP |
| Negative | FN | TN |

In confusion matrix, each row represents the instances in a predicted class, while each column represents the instances in an actual class. The positive class refers to the manipulated content, while the negative class is the genuine ones. True Positive (TP) represents the events where the model has correctly predicted the positive class, that is a manipulated photo. True Negative (TN) calculates the events that were correctly predicted as negative, that is genuine photos. False Positive (FP) and False Negative (FN) evaluate the events that were incorrectly predicted by the model, namely those legitimate photos classified as manipulated and those manipulated that

were classified as genuine, respectively.

Precision and Recall correlates the metrics described above. Precision measures the percentage of examples identified as true that are really true. That is, those photos that are really manipulated, from those that were classified as manipulated. Precision is calculated by Equation 5.1:

$$P = \frac{TP}{(TP + FP)} \tag{5.1}$$

Recall is the percentage of manipulated images that could be found, from the total number of manipulated images. Recall corresponds to the following Equation 5.2:

$$R = \frac{TP}{(TP + FN)} \tag{5.2}$$

F1-score is an harmonic mean between Precision and Recall. The range for F1-score is between [0, 1] and measures the preciseness and robustness of the classifier. That is, the amount of instances that were correctly classified and those that were misclassified, respectively. F1 measure is calculate by Equation 5.3:

$$F1 = 2 * \frac{P * R}{(P + R)} \tag{5.3}$$

ROC curve is a versatile and well-adopted technique to graphically show the performance of a binary classifier. It plots the probability of true positive rate versus the probability of a false positive rate may happen, at different classification thresholds. The ROC curve is plotted with TP rate against the FP rate where TP rate is on the y-axis and FP rate is on the x-axis as pictured in Figure 5.1.

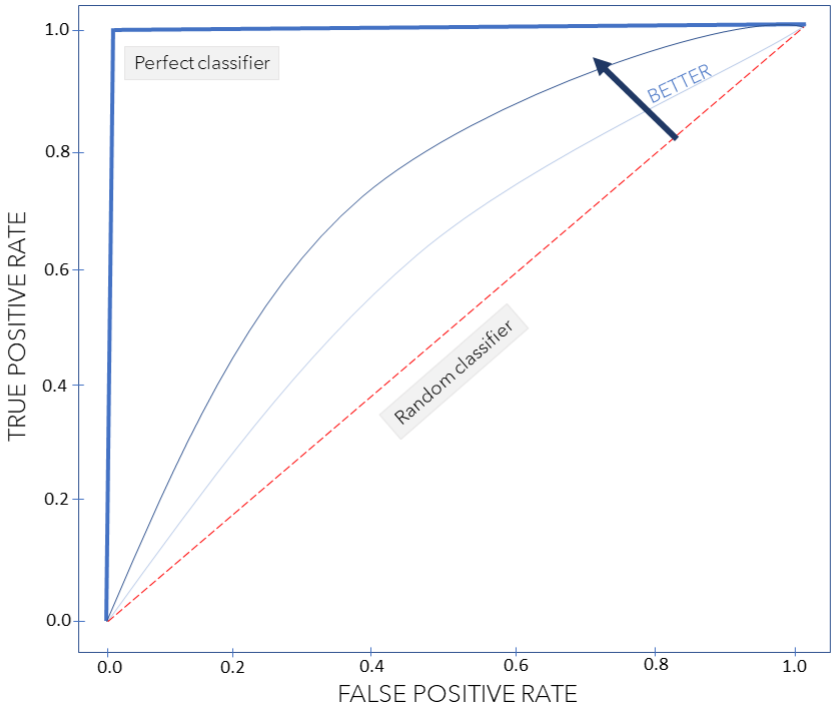



Figure 5.1: ROC curve

The red dashed line represents a random classifier, which assigns the values 1 and 0 randomly to the samples [54]. As it is possible to see in the Figure 5.1, the classifier becomes better as the ROC curve moves away from the red dashed line, to the upper left side of the figure. Lowering the threshold, increases the number of TP and FP, and vice-versa. That is, lowering the threshold causes more number of samples, which are originally labeled positive but with low predicted probability, to get re-classified as Positive (TP increases). In the same sense, samples which are originally labeled negative but with high predicted probability, also gets re-classified as Positive (FP increases). Lowering the threshold, the value of TP Rate and FP Rate approaches the value 1, hence moving towards the right and upwards along the curve. As the threshold is increased the True Negatives and False Negatives increases, and hence we move towards the left and downwards along the curve.

Another very important aspect is the evaluation of the algorithms used in order to be able to choose the best one for each situation. Usually the dataset is divided into two parts, namely one for building the machine learning model, and another for testing. The most commonly used percentages for splitting the whole dataset in training and testing sets are 50/50, 67/33 or 70/30, respectively.

One alternative to that split is to use cross-validation, where the dataset is divided into $K$ parts. The dataset is trained and tested $K$ times, each time using one of the $K$ parts for testing and the remaining for training, as pictured in Figure 5.2. In order to evaluate the method developed it was used a ten-fold cross validation. The value of $K = 10$ was choosen because it is a very common value of $K$ in the field of Machine Learning (ML) [16, 19].

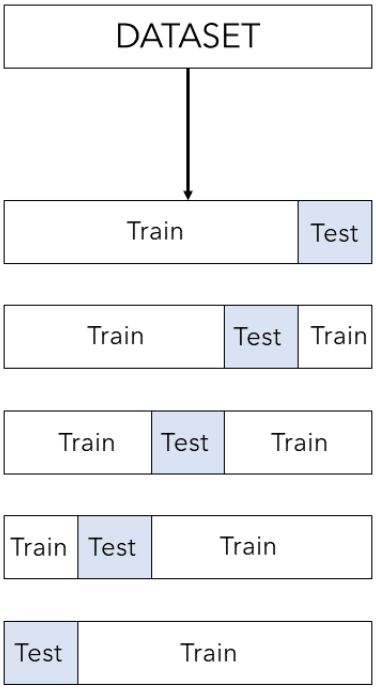



Figure 5.2: Five-fold cross validation

## 5.2 Datasets

A global dataset containing both people's faces and objects was created to train and test the SVM-based classification model. The dataset used in [32], is a compilation of photos available in CelebA-HQ dataset [45], Flickr-Faces-HQ dataset [46], "100K Faces project" (https://generated.photos/) and *"this person does not exist"* project (https://thispersondoesnotexist.com/). In order to avoid the influence of the method being applied to the same datasets as the authors of the original method [32], it was decided to add some complexity to the dataset. This was made by including in the dataset other types of manipulations besides deepfake, such as splicing and copy-move. With this new dataset an attempt is made to assess the possibility of detecting manipulations in photos and videos that contain objects and not only people's faces.

Table 5.2 itemizes the datasets collected and used in the experiments.

Table 5.2: Datasets

| Dataset | Fake | Genuine |
|---|---|---|
| CelebA-HQ dataset | - | 10,000 |
| Flickr-Faces-HQ dataset | - | 10,000 |
| "100K Facesproject" | 10,000 | - |
| "this person does not exist" | 10,000 | - |
| COVERAGE dataset | 97 | 97 |
| Columbia Image Splicing Dataset | 180 | 183 |
| Created by us | 14 | 14 |
| Celeb-DFv1 | 795 | 158 |
| | 21,086 | 20,452 |

COVERAGE dataset [83] is a copy-move forgery database with similar but genuine objects that contains 97 legitimate photos and 97 manipulated ones. Columbia Uncompressed Image Splicing Detection Evaluation Dataset [43] was also added, which consists of high-resolution photos of spaces, 180 authentic and not manipulated, and 180 spliced photos.

In addition, a small dataset of photos of everyday objects was created and these photos were manipulated through splicing and copy-move techniques using PicsArt (https://picsart.com/) and Photoshop (https://www.adobe.com/pt/products/photoshop.html). This dataset contains fourteen real photos and fourteen manipulated photos. Two examples of photos present in the newly created dataset are depicted in Figure 5.3. In Figure 5.3a) the original and not manipulated photo is illustrated, while in Figure 5.3b) the original photo was subjected to a digital manipulation process.

(a) Original image (b) Manipulated image

Figure 5.3: Example photos of the created dataset

Celeb-DF [52] was used to include fake and real videos to the final dataset. This dataset contains 795 fake videos and 158 real ones extracted from Youtube (https://www.youtube.com/). In the interest of combining these videos with the rest of the dataset, three frames per second were extracted from each video, being treated as a photo thenceforth. In total, 6,201 frames were extracted from real videos and 31,551 from fake ones.

In order to use these photos to train and test the SVM-based method, the dataset is balanced, having the same number of fake and genuine examples. To achieve that, if at some point there are more real photos than fake ones or vice-versa, it is only used the minimum between the two classes. To be more specific, as there are 31,551 fake frames extracted from videos and 6,201 real frames extracted from videos as well, it will be used only 6,201 photos from the fake ones, totaling 12,402 extracted from videos. Adding up all datasets containing only photos (all except Celeb-DF dataset, which contains videos), there are 20,291 fake photos and 20,294 real ones. To achieve the target balanced dataset, it was used 20,290 photos from each class, totalling 40,580 photos. If it had been used 20,291 instead of 20,290 (since 20,291 is the minimum between fake and real photos) a problem would arise with the ten-fold cross-validation that is intended to be used to evaluate the method employed. For the same reason, it was used 12,400 frames from videos instead of 12,402 (since 6,201 is the minimum between fake and real frames extracted from all the videos).

Table 5.3: Composition of the final dataset.

| | Training | Testing |
|---|---|---|
| Photos | 40580 | 4058 |
| Frames | 12400 | 1240 |
| Total photos and frames | 52990 | 5229 |

Putting it all together, the new dataset created in the scope of this dissertation is balanced and has 52,990 photos divided into two classes: 26,495 genuine (or real) photos and 26,495 manipulated. In order to get a better idea of how the method works for photos and videos separately, the dataset created was split in two. One part with only videos (frames extracted from videos in this case) and another with only photos. Table 5.3 specifies the composition of this datasets, namely the number of photos and frames extracted from the videos. For each dataset, the number of examples used for training and testing is also indicated. The results presented in Section 5.3 were validated through a 10-fold cross-validation methodology. That is, each dataset was equally divided into ten parts, being each one tested against the model trained with the remaining nine parts. This evaluation method will be described in more detail in Section 5.1.

## 5.3 Results and analysis

For evaluation purposes, the dataset described in Section 5.2 was divided into three parts: one part with only photos, one part with only frames taken from videos, and a third part with the mixture of the other two parts. For each part a ten-fold cross-validation was performed, and the results are shown in Tables 5.4, 5.5, and 5.6.

For each split, corresponding to the evaluation of a $K$ part of the dataset, the values of TP, TN, FP and FN were obtained. With these values obtained, the precision, recall, F1-score and accuracy were calculated using the formulas explained in 5.1.

Table 5.4: Results obtained with ten-fold cross-validation against the dataset containing only photos.

| | **TP** | **TN** | **FP** | **FN** | **Precision** | **Recall** | **F1-score** | **Accuracy** |
|---|---|---|---|---|---|---|---|---|
| Split 1 | 2016 | 2015 | 7 | 20 | 0.9965 | 0.9902 | 0.9933 | 0.9933 |
| Split 2 | 1983 | 2056 | 12 | 7 | 0.9940 | 0.9965 | 0.9952 | 0.9953 |
| Split 3 | 2057 | 1980 | 7 | 14 | 0.9966 | 0.9932 | 0.9949 | 0.9948 |
| Split 4 | 2032 | 2005 | 11 | 10 | 0.9946 | 0.9951 | 0.9949 | 0.9948 |
| Split 5 | 2012 | 2027 | 4 | 15 | 0.9980 | 0.9926 | 0.9953 | 0.9953 |
| Split 6 | 2079 | 1954 | 9 | 16 | 0.9957 | 0.9924 | 0.9940 | 0.9938 |
| Split 7 | 2004 | 2039 | 9 | 6 | 0.9955 | 0.9970 | 0.9963 | 0.9963 |
| Split 8 | 1971 | 2070 | 4 | 13 | 0.9980 | 0.9934 | 0.9957 | 0.9958 |
| Split 9 | 2026 | 2018 | 5 | 9 | 0.9975 | 0.9956 | 0.9966 | 0.9966 |
| Split 10 | 1989 | 2052 | 6 | 11 | 0.9970 | 0.9945 | 0.9957 | 0.9958 |
| **Mean** | **2017** | **2022** | **7** | **12** | **0.9963** | **0.9941** | **0.9952** | **0.9952** |

Table 5.4 describes the results obtained with a ten-fold cross-validation to the dataset containing only photos. The table highlights the partial results obtained in each split, namely the number of FP, FN, TN, and TP, as well as the calculated values for Precision, Recall, F1, and accuracy. The corresponding mean scores obtained with the ten-fold cross-validation are also indicated. The results obtained show a mean F1-score above 99.52%, which outperforms the state-of-the-art documented work [60]. The mean value obtained for accuracy ($A$) is 99.52%, which surpasses the result of 93.52% achieved in [60]. The number of incorrectly classified examples, namely FP and FN, is low, having a mean value of 7 and 12, respectively.

Table 5.5: Results obtained with ten-fold cross-validation against the dataset contaning only videos.

| | **TP** | **TN** | **FP** | **FN** | **Precision** | **Recall** | **F1-score** | **Accuracy** |
|---|---|---|---|---|---|---|---|---|
| Split 1 | 544 | 442 | 174 | 80 | 0.7577 | 0.8718 | 0.8107 | 0.7952 |
| Split 2 | 553 | 447 | 135 | 105 | 0.8038 | 0.8404 | 0.8217 | 0.8065 |
| Split 3 | 548 | 420 | 188 | 84 | 0.7446 | 0.8671 | 0.8012 | 0.7806 |
| Split 4 | 510 | 441 | 198 | 91 | 0.7203 | 0.8486 | 0.7792 | 0.7669 |
| Split 5 | 520 | 443 | 184 | 93 | 0.7386 | 0.8483 | 0.7897 | 0.7766 |
| Split 6 | 554 | 448 | 159 | 79 | 0.7770 | 0.8752 | 0.8232 | 0.8081 |
| Split 7 | 522 | 426 | 202 | 90 | 0.7210 | 0.8529 | 0.7814 | 0.7645 |
| Split 8 | 505 | 464 | 177 | 94 | 0.7405 | 0.8431 | 0.7884 | 0.7815 |
| Split 9 | 524 | 421 | 196 | 99 | 0.7278 | 0.8411 | 0.7803 | 0.7621 |
| Split 10 | 532 | 453 | 182 | 73 | 0.7451 | 0.8793 | 0.8067 | 0.7944 |
| **Mean** | **531** | **441** | **180** | **89** | **0.7476** | **0.8568** | **0.7983** | **0.7836** |

The results attained with a ten-fold cross-validation against the dataset containing only videos are presented in Table 5.5. The mean values for F1-score and accuracy are 79.8% and 78.3%, respectively. When comparing with previously documented experiments [44], it is possible to note that, using the Celeb-DF dataset [52] as part of the input dataset, the results outperform those obtained with DFT-MF approach, which achieved an accuracy of 71.25%. Regarding misclassified examples, the average values for FP and FN are respectively 180 and 89 in a total amount of 1240 examples.

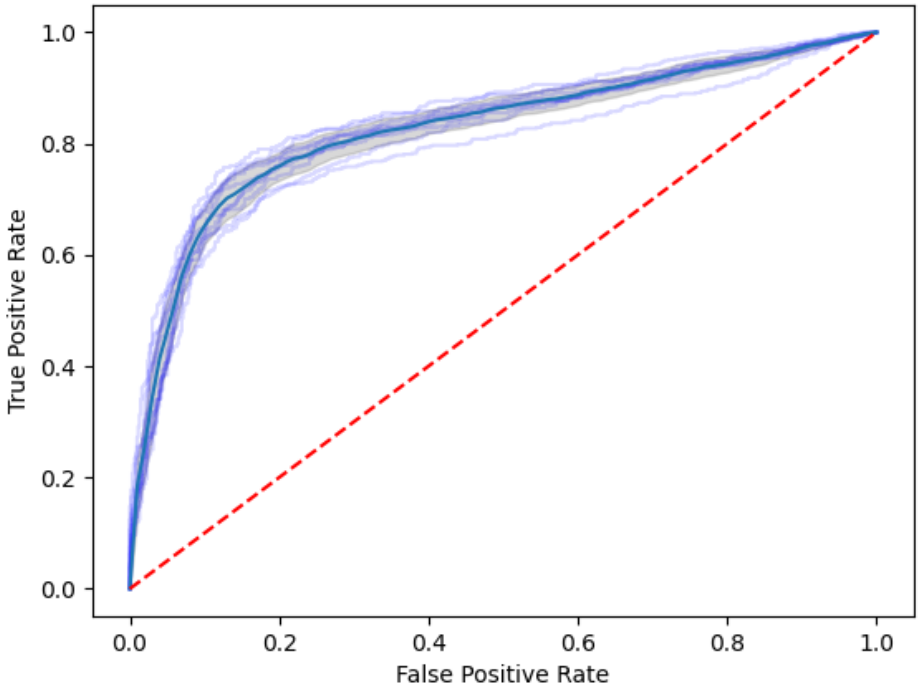
Table 5.6: Results obtained with ten-fold cross-validation against the dataset containing both photos and videos.

| | **TP** | **TN** | **FP** | **FN** | **Precision** | **Recall** | **F1-score** | **Accuracy** |
|---|---|---|---|---|---|---|---|---|
| Split 1 | 2689 | 1962 | 641 | 7 | 0.8075 | 0.9974 | 0.8925 | 0.8777 |
| Split 2 | 2689 | 2005 | 600 | 5 | 0.8176 | 0.9981 | 0.8989 | 0.8858 |
| Split 3 | 2633 | 2040 | 623 | 3 | 0.8087 | 0.9989 | 0.8938 | 0.8819 |
| Split 4 | 2627 | 2021 | 641 | 10 | 0.8039 | 0.9962 | 0.8898 | 0.8771 |
| Split 5 | 2631 | 2012 | 651 | 5 | 0.8016 | 0.9981 | 0.8892 | 0.8762 |
| Split 6 | 2656 | 2000 | 640 | 3 | 0.8058 | 0.9989 | 0.8920 | 0.8787 |
| Split 7 | 2647 | 2015 | 630 | 7 | 0.8077 | 0.9974 | 0.8926 | 0.8798 |
| Split 8 | 2596 | 2083 | 612 | 8 | 0.8092 | 0.9969 | 0.8933 | 0.8830 |
| Split 9 | 2639 | 2023 | 632 | 5 | 0.8068 | 0.9981 | 0.8923 | 0.8798 |
| Split 10 | 2627 | 2043 | 621 | 8 | 0.8088 | 0.9969 | 0.8931 | 0.8813 |
| **Mean** | **2643** | **2020** | **629** | **6** | **0.8078** | **0.9978** | **0.8927** | **0.8801** |

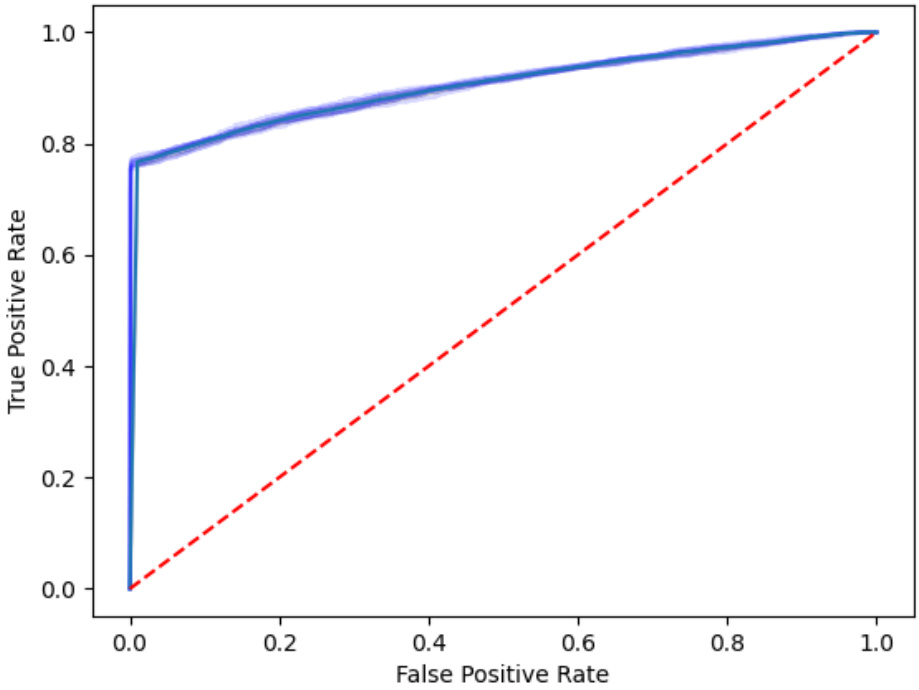
Considering that videos are composed of a set of frames, a third experiment was made to accommodate both multimedia content types. Table 5.6 presents the results obtained with the whole dataset composed of $52,990$ examples, applying a ten-fold cross-validation.

It is possible to observe that the mean values for precision, recall, and F1-score are respectively 80.78%, 99.78%, and 89.23%. The calculated mean accuracy is $88,01\%$, and the overall results outperform those attained and documented in [32].

Figure 5.4 a) depicts the ROC curve for video classification, where it is possible to observe its high performance, as the fake videos classifier gives a curve closer to the top-left corner.




(a) ROC curve for videos classification.




(b) ROCcurve for photos and videos classification.

Figure 5.4: ROC curves calculated for photos and videos processing.

Figure 5.4 b) illustrates the ROC curve related to the processing of the whole dataset, with photos and videos. It is also possible to observe the good performance of the classifier, in which the curve is pushed to the upper left corner of the graphic.

Taking every results in account, it is possible to conclude that very satisfactory results were obtained considering the work already developed and available in the literature, despite using a dataset with more diversity on manipulation types. The results could be even better with an even more diverse training dataset and having photos with high quality.

## 5.4 Benchmark with deep learning approach

The SVM model was compared with a CNN-based method, to benchmark both models in terms of classification performance and processing time.

The architecture of the CNN created to benchmark with the SVM model is depicted in 5.5.

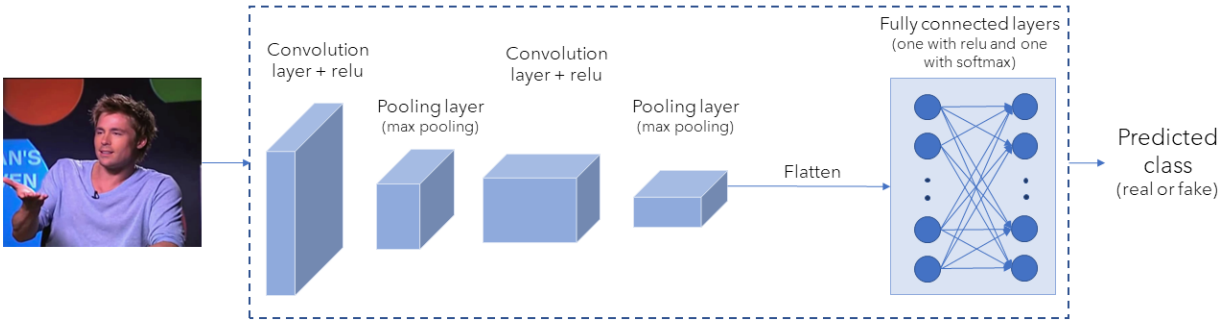



Figure 5.5: CNN architecture used to benchmark SVM-based method.

This was built with Python 3.9 on the top of `Tensorflow` (https://www.tensorflow.org/), being `keras` used to create the training and testing datasets as depicted in Listing 5.1.

Listing 5.1: CNN model

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import confusion_matrix, classification_report
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split

#Photos
src_path_train = "test_cnn_photos/train/"

train_datagen = ImageDataGenerator(
        rescale=1 / 255.0)
```

```
train_generator = train_datagen.flow_from_directory(
    directory=src_path_train,
    target_size=(300, 300),
    color_mode="rgb",
    class_mode="binary",
    subset='training',
    shuffle=False
)

X=np.concatenate([train_generator.next()[0] for i in range(train_generator.__len__())])
y=np.concatenate([train_generator.next()[1] for i in range(train_generator.__len__())])

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10)

cnn = models.Sequential([
    layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu', input_shape=(300, 300, 3)),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(filters=64, kernel_size=(3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Flatten(), #converts 3d feature map to 1d
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])

cnn.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

cnn.fit(train_generator, epochs=10)

predict=cnn.predict(X_test)
y_pred=np.argmax(predict, axis=1)

print("")
print('Confusion Matrix')
print(confusion_matrix(y_test,y_pred))
print("")
print('Classification Report')
target_names = ['Fake', 'Real']
print(classification_report(y_test, y_pred, target_names=target_names,zero_division=1))
```

Table 5.7 depicts the results obtained with the benchmarking of the videos dataset processing, namely the comparison of the results obtained with the Discrete Fourier Transform (DFT)-SVM and CNN-based methods.

Regarding videos processing, when comparing with previously documented experiments, it is possible to note that, using the Celeb-DF dataset [52] as part of the input dataset, the results outperform those obtained by [44], which uses DFT with Mouth Features to extract features and a CNN-based method to classify the videos. The reported accuracy is 71.25% and the accuracy achieved by DFT-SVM-based method is 78.36% and by CNN-based method 83.87% surpassing the reported results.

Table 5.7: Benchmark videos.

| | **Precision** | **Recall** | **F1-score** | **Accuracy** |
|---|---|---|---|---|
| DFT-SVM-based method | 0.7476 | 0.8568 | 0.7983 | 0.7836 |
| CNN-based method | 0.8820 | 0.8045 | 0.8415 | 0.8387 |

Table 5.8 describes the results obtained with the benchmarking of the photos dataset processing, namely the comparison of the results obtained with the DFT-SVM and CNN-based methods.

Table 5.8: Benchmark photos.

| | **Precision** | **Recall** | **F1-score** | **Accuracy** |
|---|---|---|---|---|
| DFT-SVM-based method | 0.9963 | 0.9941 | 0.9952 | 0.9952 |
| CNN-based method | 0.9970 | 0.9966 | 0.9968 | 0.9967 |

It is possible to observe that the results obtained are similar to those reported by the authors in [32], by applying the same DFT-SVM method, but with a more extensive dataset, comprising both faces and objects. In [22] the authors report the results obtained with an exhaustive set of deep learning-based methods applied to the detection of manipulated photos.

Regarding the processing times, CNN and SVM-based methods possess quite distinct realities. Table 5.9 depicts the observed time spent by SVM and CNN-based methods, under the same setup, for processing the aggregated dataset proposed on Section 5.2. It is possible to observe that the processing time used by the CNN-based model is considerably higher than that spent by the SVM-based method, for both photos and videos processing.

Table 5.9: Processing time spent for videos and photos, in the format hh:mm:ss.

| | **Photos** | **Videos** |
|---|---|---|
| DFT-SVM-based method | 00:00:51 | 00:02:00 |
| CNN-based method | 06:36:00 | 02:40:00 |

Deep learning based methods have been widely used, and are considered state-of-the-art cutting edge in what photo and video forensics is about [22, 86]. However, the features extraction methods and the overall functioning of deep learning based models, such as CNN and Recurrent Neural Networks (RNN), are processing time-consuming. Regarding the DFT-SVM method, the results achieved with the dataset proposed in this paper are competitive with CNN model for both photos and videos, with a significantly lower processing time. The tradeoff between the processing time and the evaluation performance obtained by DFT-SVM method [32] should

thus be taken in account in the development of forensic tools to support and help criminal investigator's digital forensics daily routine.

By observing the third-party modules listing (https://github.com/sleuthkit/autopsy_addon_modules), and also the development made by Autopsy's community like annual contest promoted by OSDF conference (https://www.osdfcon.org/), there is not yet a registered Autopsy module designed and developed to detect deepfake and digitally manipulated photos and videos in a forensics context.

## 5.5 Summary

In this chapter a description of the main evaluation metrics was made, namely precision, recall, F1-score and ROC curves. The concept of cross-validation was also described.

The composition of the dataset created in order to test and train the DFT-SVM method implemented was described, comprising in total $40,580$ photos and $12,400$ frames extracted from videos. All results obtained with a ten-fold cross validation were depicted in this chapter too. Finally, a deep learning approach (using a CNN-based method) was produced in order to benchmark both models in terms of classification performance and processing time. In chapter 6 all the conclusions reached with this dissertation will be addressed.

# Chapter 6

# Conclusions

In this chapter the development and results accomplished throughout this dissertation is going to be described. First, a brief overview of the research and development made during the dissertation is made, commenting on the deliverables obtained and the goals achieved. Secondly, a description of the results obtained is done, commenting on the tradeoff between the processing time and the evaluation performance. Finally, some directions on future work are given.

## 6.1 Research and development

This dissertation described the development of an application to detect tampered multimedia content. An Support Vector Machines (SVM)-based method was implemented in a standalone application, to process the previously extracted features obtained by a Discrete Fourier Transform (DFT) calculation in each multimedia file being processed. Two modules for Autopsy digital forensics tool were developed, namely a module to detect tampered photos and another one to identify deepfake videos. The fundamentals behind digital forensics, SVM, and Convolution Neural Networks (CNN) were described and the most relevant and up-to-date literature review related to digital forensics on multimedia content was made, namely the survey on deep learning-based methods applied to photos and videos forensics.

The deliverables obtained with this dissertation, namely the ready-to-use Autopsy modules, give a helping hand to digital forensics investigators and leverage the use of Machine Learning (ML) techniques to fight cybercrime activities that involve multimedia files. The overall architecture and development take advantage of two well-known and documented techniques to deal with feature extraction in multimedia content and to automatically detect from learning classifier models, respectively DFT technique to extract features from photos, and SVM to classify files. Both techniques were incorporated in the developed standalone application, which was further integrated as two separated Autopsy modules. The dataset proposed in [32] was extended with different sources, mainly to accommodate deepfake videos and other types of photo manipulations such as copy-move and splicing. The final dataset has about $50,000$ photos, enriched with

faces and objects, where it is possible to find examples of deepfake, splicing, and copy-move manipulations. Some of the photos are frames extracted from deepfake videos.

Having considered all these points, all the initial objectives were completed successfully and the objective of detecting manipulated videos, that was not in the initial objectives, was also fulfilled.

## 6.2 Results

The results were presented into three distinct dimensions: the classification performance obtained with a ten-fold cross-validation for photos and videos processing; the benchmark between SVM and CNN-based methods using the dataset created; the processing time of SVM and CNN-based methods.

It was possible to achieve a mean F1-score of around 99.52% for manipulated photos detection and 79, 83% for deepfake video detection and 89.27% detecting both manipulated photos and videos using DFT-SVM-based method. Deep learning methods, namely CNN-based method, outperformed those achieved by DFT-SVM, however with a processing time considerably higher. In a strict concern with daily-routine digital forensic interest, despite the better results obtained with CNN-based methods, the tradeoff with the processing time benefits the use of SVM method with the features extracted by DFT since a digital forensics investigation is intended to be completed as quickly as possible.

## 6.3 Future work

By analyzing the misclassified photos and video frames, a possible cause could be related to the low resolution of the photos and different resolutions. A richer dataset with heterogeneous examples regarding the resolution of the photos would improve the overall results obtained. The optimal number of features that should be extracted from the photos, and its impact in computational time, is also worth investigating. An ensemble of learning classifiers, composed of both deep learning and SVM-based methods could benefit both the performance obtained and the processing time. A net model for forensic detection using CNN, eventually using a different architecture, is also worth investigating and implementing.

Besides the well-accepted implementation in Autopsy modules, an emergent subject that may benefit from the developed architecture is the detection of fake news and the spread of hate speech in social networks. The low processing time an the high performance obtained with DFT-SVM method, makes it eligible to be incorporated as a plugin that may be used easily and in real-time, to detect the fakeness level of multimedia contents spread in social networks. The modules will keep on being improved to be submitted at 2021 Autopsy Module Development Contest by OSDF (https://www.osdfcon.org/2021-event/2021-module-development-contest/).

# Bibliography


[1] Autopsy user documentation: Android analyzer module. https://sleuthkit.org/autopsy/docs/user-docs/3.1/android_analyzer_page.html. (Accessed on 06/02/2021).

[2] Autopsy user documentation: Data source integrity module. http://sleuthkit.org/autopsy/docs/user-docs/4.17.0/data_source_integrity_page.html. (Accessed on 06/02/2021).

[3] Autopsy user documentation: File type identification module. https://sleuthkit.org/autopsy/docs/user-docs/3.1/file_type_identification_page.html. (Accessed on 06/02/2021).

[4] ENISA Threat Landscape - 2020. https://www.enisa.europa.eu/topics/threat-risk-management/threats-and-trends/. (Accessed on 03/16/2021).

[5] Module development overview. https://www.sleuthkit.org/autopsy/docs/api-docs/4.2/platform_page.html, April 2017. (Accessed on 09/30/2020).

[6] Accenture/ponemon institute: The cost of cybercrime. *Network Security*, 2019(3):4, 2019.

[7] Relatório anual de segurança interna 2019. https://www.portugal.gov.pt/pt/gc22/comunicacao/documento?i=relatorio-anual-de-seguranca-interna-2019-, June 2020. (Accessed on 03/30/2021).

[8] Relatório anual de segurança interna 2020. https://www.portugal.gov.pt/pt/gc22/comunicacao/documento?i=relatorio-anual-de-seguranca-interna-2021, March 2021. (Accessed on 05/01/2021).

[9] H. Alheneidi, L. AlSumait, D. AlSumait, and A. P. Smith. Loneliness and problematic internet use during covid-19 lock-down. *Behavioral Sciences*, 11(1):5, 2021.

[10] R. Anderson and T. Moore. The economics of information security. *science*, 314(5799):610–613, 2006.

[11] M. Antunes and B. Rodrigues. Introdução à cibersegurança–a internet, os aspetos legais e a análise digital forense, 2018.

[12] M. Bada and J. R. Nurse. The social and psychological impact of cyberattacks. In *Emerging Cyber Threats and Cognitive Vulnerabilities*, pages 73–92. Elsevier, 2020.

[13] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[14] P. Bhatt and P. H. Rughani. Machine learning forensics: A new branch of digital forensics. *International Journal of Advanced Research in Computer Science*, 8(8), 2017.

[15] G. K. Birajdar and V. H. Mankar. Digital image forgery detection using passive techniques: A survey. *Digital investigation*, 10(3):226–245, 2013.

[16] S. Borra and A. Di Ciaccio. Measuring the prediction error. a comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics  Data Analysis*, 54:2976–2989, 12 2010.

[17] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[18] J. Botha and H. Pieterse. Fake news and deepfakes: A dangerous threat for 21st century information security. In *International Conference on Cyber Warfare and Security*, pages 57–XII. Academic Conferences International Limited, 2020.

[19] J. Brownlee. A gentle introduction to k-fold cross-validation. https://machinelearningmastery.com/k-fold-cross-validation/, March 2018. (Accessed on 06/12/2021).

[20] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. volume 6314, pages 778–792, 09 2010.

[21] E. Casey. The chequered past and risky future of digital forensics. *Australian Journal of Forensic Sciences*, 51(6):649–664, 2019.

[22] I. Castillo Camacho and K. Wang. A comprehensive review of deep-learning-based methods for image forensics. *Journal of Imaging*, 7(4):69, 2021.

[23] R. Caubalejo. Image processing — blob detection. https://towardsdatascience.com/image-processing-blob-detection-204dc6428dd, January 2021. (Accessed on 06/02/2021).

[24] P. Chakravorty. What is a signal? [lecture notes]. *IEEE Signal Processing Magazine*, 35(5):175–177, 2018.

[25] M. Chen, J. Fridrich, M. Goljan, and J. Lukás. Determining image origin and integrity using sensor noise. *IEEE Transactions on information forensics and security*, 3(1):74–90, 2008.

[26] J. Christian. Experts fear face swapping tech could start an international showdown, February 2018. (Accessed on 03/11/2021).

[27] I. P. T. Council. What is photo metadata. https://iptc.org/standards/photo-metadata/photo-metadata/. (Accessed on 09/28/2020).

[28] A. da República. Lei 109/2009, 2009-09-15 - dre. https://dre.pt/pesquisa/-/search/489693/details/maximized, September 2009. (Accessed on 06/02/2021).

[29] C. N. de Cibersegurança. Relatório riscos e conflitos 2021. https://www.cncs.gov.pt/content/files/relatorio_riscos.conflitos2021___observatoriociberseguranca_cncs.pdf, May 2021. (Accessed on 05/24/2021).

[30] A. P. de Ciências Forenses. Informática forense. http://apcforenses.org/?page_id=36. (Accessed on 09/25/2020).

[31] L. Deng and D. Yu. Deep learning: methods and applications. *Foundations and trends in signal processing*, 7(3–4):197–387, 2014.

[32] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper. Unmasking deepfakes with simple features. *arXiv preprint arXiv:1911.00686*, 2019.

[33] X. Feng, I. J. Cox, and G. Doërr. An energy-based method for the forensic detection of re-sampled images. In *2011 IEEE International Conference on Multimedia and Expo*, pages 1–6. IEEE, 2011.

[34] A. J. Fridrich, B. D. Soukal, and A. J. Lukáš. Detection of copy-move forgery in digital images. In *in Proceedings of Digital Forensic Research Workshop*. Citeseer, 2003.

[35] S. L. Garfinkel. Digital forensics research: The next 10 years. *digital investigation*, 7:S64–S73, 2010.

[36] A. Hadhazy. Is that iranian missile photo a fake? https://www.scientificamerican.com/article/is-that-iranian-missile/, July 2008. (Accessed on 03/05/2021).

[37] D. Harris. Deepfakes: False pornography is here and the law cannot protect you. *Duke L. & Tech. Rev.*, 17:99, 2018.

[38] J. He, D. Li, B. Yang, S. Cao, B. Sun, and L. Yu. Multi view facial action unit detection based on cnn and blstm-rnn. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 848–853. IEEE, 2017.

[39] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[40] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[41] HiDEF. An introduction to the principles of video 101. https://www.hidefnj.com/video. (Accessed on 03/10/2021).

[42] G. Horsman. Tool testing and reliability issues in the field of digital forensics. *Digital Investigation*, 28:163–175, 2019.

[43] Y.-F. Hsu and S.-F. Chang. Detecting image splicing using geometry invariants and camera characteristics consistency. In *2006 IEEE International Conference on Multimedia and Expo*, pages 549–552. IEEE, 2006.

[44] M. T. Jafar, M. Ababneh, M. Al-Zoube, and A. Elhassan. Forensics and analysis of deepfake videos. In *2020 11th International Conference on Information and Communication Systems (ICICS)*, pages 053–058, 2020.

[45] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[46] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[47] D. Karunakaran. Intro to artificial intelligence | medium. https://medium.com/intro-to-artificial-intelligence/deep-learning-series-1-intro-to-deep-learning-abb1780ee20, April 2018. (Accessed on 06/15/2021).

[48] A. Kaur and J. Rani. Digital image forgery and techniques of forgery detection : A brief review. 2016.

[49] K. Kertysova, E. Frinking, K. van den Dool, A. Maričić, and K. Bhattacharyya. Cybersecurity: Ensuring awareness and resilience of the private sector across europe in face of mounting cyber risks-study. Technical report, Tech. rep. European Economic and Social Committee, 2018., 2018.

[50] J. Kietzmann, L. W. Lee, I. P. McCarthy, and T. C. Kietzmann. Deepfakes: Trick or treat? *Business Horizons*, 63(2):135–146, 2020.

[51] H. S. Lallie, L. A. Shepherd, J. R. Nurse, A. Erola, G. Epiphaniou, C. Maple, and X. Bellekens. Cyber security in the age of covid-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic. *Computers Security*, page 102248, 2021.

[52] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3207–3216, 2020.

[53] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.

[54] S. Manna. Evaluation metrics part 3. https://medium.com/the-owl/evaluation-metrics-part-3-47c315e07222, June 2020. (Accessed on 06/17/2021).

[55] B. Martinez, M. F. Valstar, B. Jiang, and M. Pantic. Automatic analysis of facial actions: A survey. *IEEE transactions on affective computing*, 10(3):325–347, 2017.

[56] R. Metz. These people do not exist. why websites are churning out fake images of people (and cats) - cnn. https://edition.cnn.com/2019/02/28/tech/ai-fake-faces/index.html, February 2019. (Accessed on 05/03/2021).

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

[58] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi. Deep learning for deepfakes creation and detection. *arXiv preprint arXiv:1909.11573*, 1, 2019.

[59] P. Niyishaka and C. Bhagvati. Digital image forensics technique for copy-move forgery detection using dog and orb. *International Conference, ICCVG 2018*, 2018.

[60] P. Niyishaka and C. Bhagvati. Digital image forensics technique for copy-move forgery detection using dog and orb. In *International Conference on Computer Vision and Graphics*, pages 472–483. Springer, 2018.

[61] N. U. Online. 5 steps for conducting computer forensics investigations. September 2017. (Accessed on 10/07/2020).

[62] N. U. Online. 5 steps for conducting computer forensics investigations. https://online.norwich.edu/academic-programs/resources/5-steps-for-conducting-computer-forensics-investigations, September 2017. (Accessed on 04/08/2021).

[63] K. O'Shea and R. Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[64] R. C. Pandey, S. K. Singh, and K. K. Shukla. Passive forensics in image and video using noise features: a review. *Digital Investigation*, 19:1–28, 2016.

[65] Prabhu. Understanding of convolutional neural network (cnn) — deep learning. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148, March 2018. (Accessed on 06/17/2021).

[66] J. Pu, N. Mangaokar, L. Kelly, P. Bhattacharya, K. Sundaram, M. Javed, B. Wang, and B. Viswanath. Deepfake videos in the wild: Analysis and detection. *arXiv preprint arXiv:2103.04263*, 2021.

[67] S. Raghavan. Digital forensic research: current state of the art. *CSI Transactions on ICT*, 1(1):91–114, 2013.

[68] K. Roose. Here come the fake videos, too, March 2018. (Accessed on 03/11/2021).

[69] M. Roškot, I. Wanasika, and Z. K. Kroupova. Cybercrime in europe: surprising results of an expensive lapse. *Journal of Business Strategy*, 2020.

[70] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. pages 2564–2571, 11 2011.

[71] K. P. Shung. Accuracy, precision, recall or f1?, April 2020. (Accessed on 05/03/2021).

[72] C. Silva and B. Ribeiro. *Aprendizagem Computacional em Engenharia*. Imprensa da Universidade de Coimbra/Coimbra University Press, 2018.

[73] M. . Simulink. Convolutional neural network. https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html#how-they-work. (Accessed on 06/17/2021).

[74] V. Sing. Image forgery detection. using the power of cnn's to detect image manipulation. https://towardsdatascience.com/image-forgery-detection-2ee6f1a65442. (Accessed on 06/01/2021).

[75] S. Soltani and S. A. H. Seno. A survey on digital evidence collection and analysis. In *2017 7th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 247–253, 2017.

[76] R. Spivak. Deepfakes: The newest way to commit one of the oldest crimes. *Geo. L. Tech. Rev.*, 3:339–340, 2019.

[77] V. Tabora. Image resizing and resampling — it's not the same thing. https://medium.com/hd-pro/image-resizing-and-resampling-its-not-the-same-thing-b7a607b7fa09, November 2020. (Accessed on 05/03/2021).

[78] A. Tait. How a badly faked photo of vladimir putin took over twitter. https://www.newstatesman.com/science-tech/social-media/2017/07/how-badly-faked-photo-vladimir-putin-took-over-twitter, July 2017. (Accessed on 03/05/2021).

[79] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *Information Fusion*, 64:131–148, 2020.

[80] V. Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[81] V. N. VAPNIK. The nature of statistical learning. *Theory*, 1995.

[82] D. G. Viswanathan. Features from accelerated segment test (fast). In *Proceedings of the 10th workshop on Image Analysis for Multimedia Interactive Services, London, UK*, pages 6–8, 2009.

[83] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler. Coverage—a novel database for copy-move forgery detection. In *2016 IEEE international conference on image processing (ICIP)*, pages 161–165. IEEE, 2016.

[84] M. Westerlund. The emergence of deepfake technology: A review. *Technology Innovation Management Review*, 9(11), 2019.

[85] B. Xu, G. Liu, and Y. Dai. Detecting image splicing using merged features in chroma space. *The Scientific World Journal*, 2014, 2014.

[86] P. Yang, D. Baracchi, R. Ni, Y. Zhao, F. Argenti, and A. Piva. A survey of deep learning-based source image forensics. *Journal of Imaging*, 6(3):9, 2020.

[87] R. Zhi, M. Liu, and D. Zhang. A comprehensive survey on automatic facial action unit analysis. *The Visual Computer*, 36(5):1067–1093, 2020.