Optimized Detector of Manipulated Media Content

David Miguel Santos Maia
Mestrado Segurança Informática
Departamento de Ciências de Computadores
2022

Orientador
Mário João Gonçalves Antunes, Professor Coordenador,
Escola Superior de Tecnologia e Gestão do Politécnico de Leiria

Coorientador
Manuel Eduardo Carvalho Duarte Correia, Professor Associado,
Faculdade de Ciência da Universidade do Porto

U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

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

O Presidente do Júri,

Porto, _____/_____/_____

Abstract

The digitalization of crime created a need for a new forensics field. Computer Forensics is not just an extension of other forensic fields. It aggregates 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. Even if the crime did not occur in the digital dimension, the investigators might be interested in using evidence collected such as files, e-mail addresses, phone contacts or other information, that can only be analysed by using sophisticated digital forensics tools.

Multimedia content (pictures and videos) have particular relevance to digital forensics, giving its relationship with crimes that have a significant public opinion. Fake news, disinformation, sexual abuse, and the dissemination of pornographic content, specially Deepfake pornography, are just a few examples of situations where reported content must be analysed by criminal investigation teams. Manual and time-consuming approaches are typically used in these investigations, which may cause many to not move forward for lack of better resources.

Many advances have been made in creating more accurate and efficient solutions. Yet, there remains of a need for faster and lighter solutions that can be deployed for (almost) real time application scenarios. These tools can not only be effectively used to investigate crimes after they have been committed, but also help to prevent them, by immediately flagging malicious content upon posting or sharing.

This dissertation describes the development of a Support Vector Machines (SVM)-based standalone solution for detecting manipulated media. We also highlight different Feature Reduction and Feature extraction techniques combinations and describe how they perform for detecting manipulated images. We have also created a dataset with both real and manipulated files containing a mixture of images, from existing projects and media from social media platforms, to train and test the developed SVM based model. This is also compared with a Convolutional Neural Network (CNN)-based model that we have also trained with the same dataset. We have also compared and benchmarked both models for efficiency and accuracy on detecting manipulated images.

Resumo

A digitalização do crime criou a necessidade de um novo campo forense. Informática Forense não é apenas uma extensão de outros campos forenses. Agrega técnicas, ferramentas e procedimentos para a recolha, preservação e análise de provas digitais em equipamentos eletrónicos, incluindo discos, smartphones e outros dispositivos com capacidade de armazenamento. Mesmo que o crime não tenha ocorrido na dimensão digital, os investigadores podem ter interesse em utilizar as provas recolhidas como ficheiros, endereços de correio eletrónico, contactos telefónicos ou outras informações, que apenas podem ser analisadas através do referido campo forense.

Os conteúdos multimédia (fotos e vídeos) têm particular relevância para a perícia digital, dando a sua relação com crimes que têm uma opinião pública significativa. Fake news, desinformação, abuso sexual e disseminação de conteúdo pornográfico, especialmente pornografia "Deepfake", são apenas alguns exemplos de situações em que o conteúdo denunciado deve ser analisado por equipas de investigação criminal. Abordagens manuais e demoradas são normalmente usadas nesta investigação, o que pode fazer com que muitas investigações não avancem por falta de recursos.

Muitos avanços foram feitos na criação de soluções cada vez mais precisas. No entanto, permanece a necessidade de uma solução mais rápida e leve que possa ser implementada em aplicativos (quase) em tempo real. O que poderia não apenas ser usado para investigar o crime posteriormente, mas também preveni-lo sinalizando esse conteúdo ao publicar ou compartilhar.

Esta dissertação descreve o desenvolvimento de uma solução autonoma baseada em Support Vector Machines (SVM) para detecção de mulimédia manipulada. Diferentes combinações de técnicas de redução de recursos e extração de recursos e como elas funcionam. Um conjunto de dados com arquivos reais e manipulados contendo uma mistura de imagens de projetos existentes e multimédia de redes sociais também foi criado para treinar e testar o modelo. O método utilizado foi comparado com uma Rede Neural Convolucional (CNN) ao fazer uma comparação de vários aspectos nestes modelos.

Acknowledgements

I would like to thank my supervisors Professor Mário Antunes and Professor Manuel Correia because without them none of this was possible. Your support and interest was encouraging for me and a definitive incentive to go one step further. I also cannot forget my family, who supported me and allowed me to pursue my goals.

Contents

List of Tables

List of Figures

Listings

Acronyms

AT Autoridade Tributária e Aduaneira

ASAE Autoridade de Segurança Alimentar e Económica

CNN Convolutional neural network

DCC Departamento de Ciência de Computadores

DFT Discrete Fourier transformation

ENISA European Union Agency for Cybersecurity

EU European Union

FCUP Faculdade de Ciências da Universidade do Porto

FFT Fast Fourier transformation

FN False Negative

FP False Positive

GNR Guarda Nacional Republicana

ML Machine Learning

OPC criminal police bodies

PCA Principal Component Analysis

PJ Polícia Judiciária

PJM Polícia Judiciária Militar

PM Polícia Marítima

PSP Polícia de Segurança Pública

SEF Serviço de Estrangeiros e Fronteiras

SVM Sector Vector Machine

TN True Negative

TP True Positive

VM Virtual Machine

Chapter 1

Introduction

In Roman mythology, there is a story of a spirit called Dolos (trickery) who became known for his artistic skills when attempted to create a fraudulent copy statue of Aletheia (Veritas, the truth). The copy was so similar that would fool most, but a closer look would show that it had no feet [72].

Centuries after this story was written, a lot of criminals continue trying to trick people with forgeries of paintings, statues and other forms of art. In this problem, the difficulty is to notice some aspect that is different from the original art. The specific colour pallets, the form in which the artist used the brush, any of those can be details used by experts to identify forgeries [26].

Fast forwarding, the world has gone digital, and so did a lot of the content we produce and consume. This includes social media content, streaming platforms and the news. In fact, more people consume news in their devices than in any other medium, and not necessarily in official news website or pages [54]. In fact, as seen in figure 1.1 a lot of people use social media to get their news.

Any unlawful behavior involving a computer, networked device, or network is considered cybercrime. While the majority of this type of crime is carried out in order for the perpetrators to profit, some type of cybercrime are carried out directly against computers or devices in order to harm or disable them. Common forms of cybercrimes are child pornography distribution, cyberstalking, identity theft, cyber laundering, credit card theft, cyber terrorism, drug sale, data leakage, sexually explicit content, phishing, and other forms of cyber hacking. They mostly lead to a privacy breach, security violation, business loss, financial fraud, or damage in public and government properties[1].

Obtaining correct and official statistics on cybercrimes is challenging because of the culture in which the crimes were committed, the severity of the offences, and the unreported incidents due to the lack of knowledge or societal constraints. Nonetheless, Portugal produces an annual report named RASI (Annual Report of Internal Security) in which it's presented the data related to crime in the country [17].

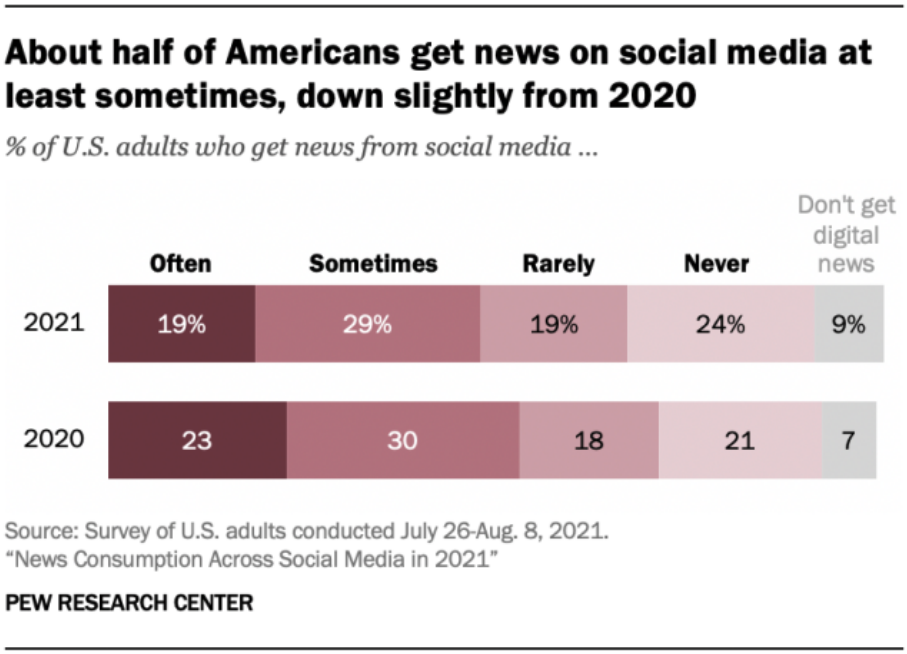

Figure 1.1: Frequency from which Americans get their news from social media [50]

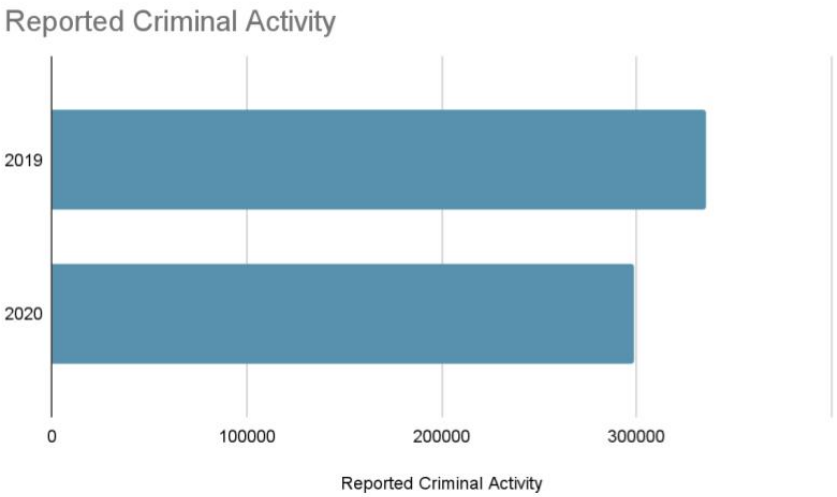

Figure 1.2: Reported Crime in Portugal [17]

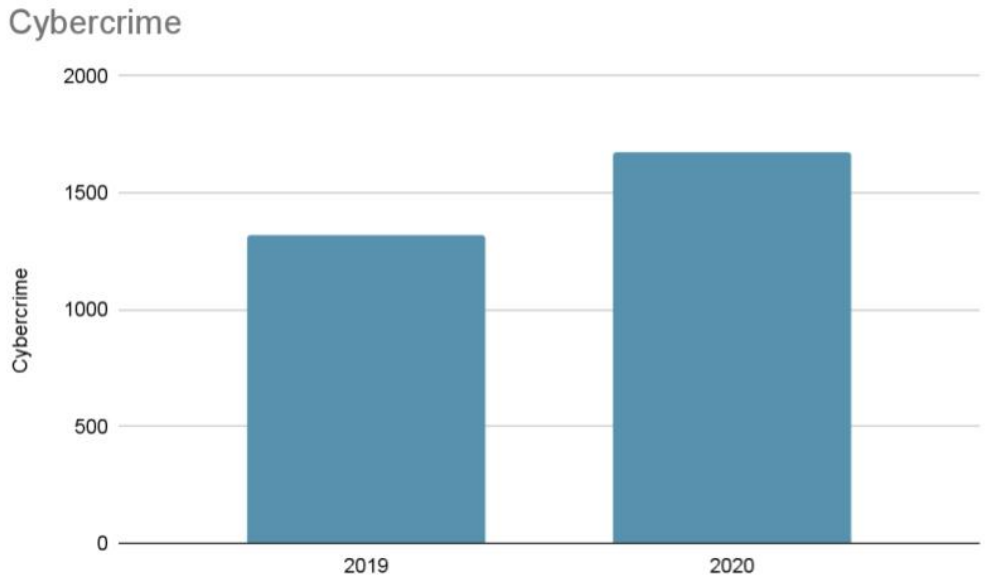

Figure 1.3: Reported Cybercrime in Portugal [17]

The data presented in the RASI is collected by the eight OPC (criminal police bodies), namely GNR (Republican National Guard), PSP (Public Security Police), PJ (Judiciary Police), SEF (Foreign Service and Borders), PM (Maritime Police), ASAE (Food and Economic Safety Authority), AT (Tributary and customs authority) and PJM (Military Judiciary Police). As presented by the graph, in 2020, the total number of registered criminal reports was 298.797, less 36.817 participations that in 2019, meaning a variation of -11% [17].

Clearly depicted in Figure 1.2 and Figure 1.3 is the fact that, although general crime is lowering, cybercrime is on the rise. The trend also applies to "Computer and Communications Fraud", a crime that has a broad scope: phishing, smishing (SMS phishing) and vishing (Voice Call Phishing) are some of the attacks. The explanation is the effectiveness of this attacks and the increased availability of tools that enable attackers. These tools have become very easy to use, very affordable or even free. This creates a problem for investigators of Law enforcement since it might be hard to keep up with such tools or techniques.

The word *forensics* means *"to bring to the court"* and forensics is the process of using scientific knowledge for collecting, analyzing, and presenting evidence to the courts. Forensics deals primarily with the recovery and analysis of latent evidence, which can assume many forms, from fingerprints left on a window to DNA evidence recovered from blood stains to the files on a hard drive. Computer forensics, also know as Digital Forensic, is the field that combines elements of law and computer science to collect and analyze data from computer systems, networks, wireless communications, and storage devices in a way that is admissible as evidence in a court of law[73]. From a technical standpoint, the main goal of computer forensics is to identify, collect, preserve, and analyze data in a way that preserves the integrity of the digital evidence collected

Figure 1.4: Cellebrite UFED a tool used by law enforcement for computer forensics [19]

so it can be used effectively in a legal case.

Video surveillance, videos and images, have been used for a long time in courts, or to assist police investigations. Similar to any piece of evidence, they need to be authenticated and should have a clear and rigorous chain of custody, otherwise this evidence might not be admissible and cannot be use to convict anyone.

Contrary to a court-of-law, the public opinion does not ask for chain of custody or even authentication of information shared. That practice, and the fact that people aren't cross checking their news, caused a big and current problem know as "fake News". Currently social networks are filled with these type of news, manipulated information designed to control public opinion. Unfortunately, it is becoming increasingly harder to detect this information and there is a need for tools that easily identify its veracity.

When the analysis is performed manually, by a human analyst, it can take a long time for a decision to be made. Whether to restrict the material, delete it, block the user, or even determine that the content is original, or manipulated in an accepted way[24]. No matter how long this task may take, there not a slightest chance that companies can keep up with attackers. Specially if these companies, and investigative services, switch to more automated, or even autonomous, mechanisms that can process media in an accelerated manner.

Deepfake is probably the biggest topic nowadays in image manipulation. Even people unfamiliar with the matter know and talk about it. Its notoriety is related with the fact that is becoming an enormous problem in our society. Figure 1.5 presents the growth of deepfake phenomenon. Deepfakes have become popular due to the quality of tampered videos and also the easy-to-use ability of their applications to a wide range of users with various computer skills from professional to novice. These applications are mostly developed based on deep learning

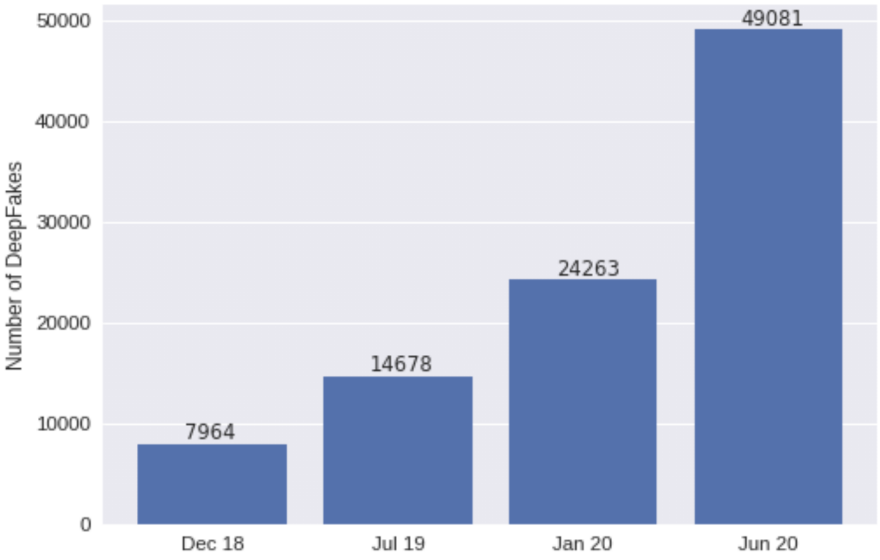

Figure 1.5: Number of deepfakes identified online[61]

techniques, which is well known for its capability of representing complex and high-dimensional data[58].

Similar to most digital technologies, the quality of deepfakes is increasing at an alarming rate, and it is clear that even the most complex deepfake tools will be as easy to use as Instagram [1] filters in the very near future. Not all applications of this technology are meant to be malicious, and we may find beneficial applications for social media and its surrounding technology. For example, PantheonLab [2] is a company dedicated to create software that allows users to manipulate videos to introduce voice cloning, lip synchronization or face shifting. The purpose is marketing content creation, but this and other similar products can be exploited by individuals with malicious intentions.

Deepfake pornography is where computer technology is used to map the faces of celebrities and private citizens on to explicit sexual material [67]. This is a definitely malicious use of deepfake technology. An attack like this may have different motivations. As discussed previously, manipulate media can victimize an individual or the entire society, and the need for a solution that can effective detect this forgery is enormous.

Past work, develop by Sara Ferreira [24], at the Department of Computer Science, from the Faculty of Sciences of University of Porto, achieved promising results on detecting manipulated images by applying machine learning based methods, such as Support Vector Machines (SVM) and Convolutional Neural Networks (CNN). It was possible to achieve a mean F1-score of around 99.52% for manipulated photos detection, 79.83% for deepfake video detection and 89.27% detecting both manipulated photos and videos using DFT-SVM-based method [24]. A CNN-based method was also research but the processing time was proved to be too high for the

[1] A popular social media platform
[2] https://www.pantheonlab.ai/

demands needed.

Despite the developments made so far on the detection of tampered multimedia content, there are however some limitations and open issues. Heavy models tend to only be used in limited contexts, since not everyone will have access to high capacity computing power. Slow solutions will also alienate potential users since practicality was not achieved.

Another current problem is the fact that models and dataset from this field disregard the usage of filters. A capability that a lot of users take advantage since a lot of content is created using smartphones and applications that support such features.

1.1 Goals

The objectives below were defined for the work presented in this dissertation and were fully attained throughout the research:

- To create a system or framework that will enable users to detect manipulated images by presenting a confidence score;
- To bridge the gap between lab models and real-life applications like social media images or smartphone filters;
- To create a new dataset with media shared across social media platforms, that also considers filters, based on an existing dataset;
- To identify new methods for feature extraction and to make the ML pipeline more efficient and faster, namely by applying techniques such as FFT, and using PCA to reduce the data resulting from features extraction.
- To benchmark the final solution using the dataset created, when comparing with other existing datasets;

1.2 Contributions

The major contributions produced during the elaboration of this dissertation can be described as follows:

- A labeled dataset of 20808 multimedia files, which contains several state-of-the-art datasets of non-manipulated images and other subject to manipulation. The dataset includes a subset of of manipulated social media images, as well as images created for this project using social media application for manipulation.

- A published `pypi` module, that can be used by Python developers to create new projects involving the detection of manipulated images. The `pypi` module is available at https://pypi.org/project/Media-Manipulation-Detector/.
- A published GitHub open source project with the scripts developed to detect the manipulations in images. The full set of scripts and other material produced, is available at https://github.com/devmaia/Photo-and-video-manipulations-detector

1.3 Thesis Outline

This dissertation is divided into six chapters, which are described as follows.

Chapter 2 prepares the ground for the rest of this dissertation by presenting the fundamental concepts needed to understand the problem, along with the proposed solutions. The idea behind media manipulation and how is achieved is also described, along with an explanation of the machine learning techniques used to detect such manipulations, namely SVM and CNN. It also presents the state-of-the-art with literature review, analysis of previous projects to sustain the decisions made. The most relevant techniques and scientific research applied to the field of media content manipulation are mentioned, analysed and reviewed.

Chapter 3, lays out the architecture for this project, explaining the several stages of the pipeline deployed to preprocess and process the images dataset. Additionally it goes over the process for the creation of the final dataset, as well as the development of the code until public release.

Chapter 4 presents the results achieved in this dissertation. Starts by describing the benchmark process applied, how it was implemented and the results obtained.

Finally, in Chapter 5, the conclusions that were reached during the development of this project are described, based on the results obtained, and some ideas and proposals on how it can be improved in the future.

Chapter 2

Fundamentals

For a clear understanding of the topics, that will be mentioned in the following chapters, this chapter will present the essential and fundamental concepts behind the research topics explored in this dissertation. This project is about detecting media manipulation, fundamental concepts that need to be understood are digital forensics principles, media files essentials, multimedia manipulations, machine learning methods applied to classification and detection, and Fourier transformations. These subjects are described in the following sections.

2.1 Digital forensics

Digital forensics, or Computer Forensics, is the application of science to the identification, collection, examination, and analysis, of data while preserving the integrity of the information and maintaining a strict chain of custody for the data [42].

Figure 2.1 illustrated the main four stages of computer forensics.

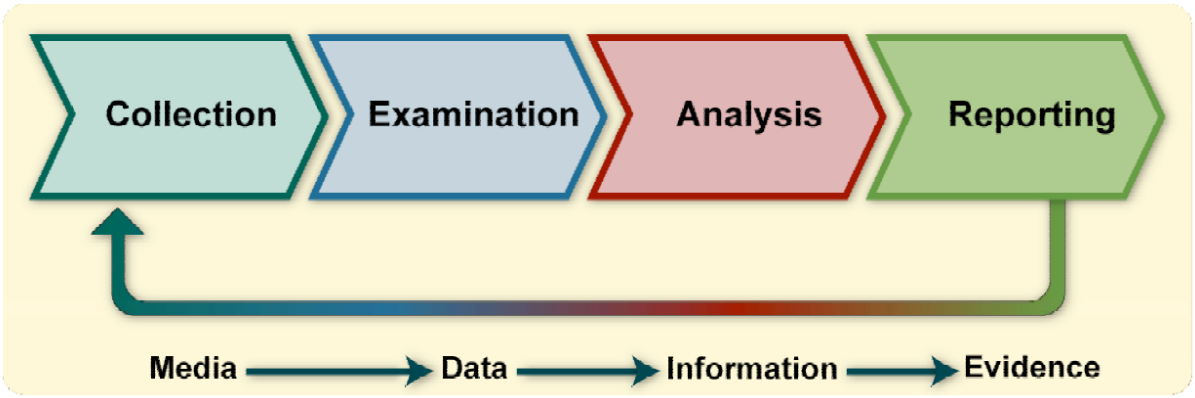

Figure 2.1: Computer Forensics Stages [42]

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. This is achieved by the implementation of a process called chain of custody.

In Portugal, Law number 109/2009[14] is the most popular law and it's known as Cybercrime Law ("Lei do Cibercrime", in Portuguese). Due to some changes in the crime landscape some revisions were introduced in 2021 with Law 79/2021[15]. The revisions did not change the core idea of the law, instead they aligned it with EU directive 2019/713 [20]. The law describes the meaning of computer related crimes (computer sabotage, computer fraud, credit card fraud, illegitimate access,a among others) along with the procedures to produce evidence in the intended form, assuring the legal procedure of evidence validity in court.

Although NIST (National Institute of Standards and Technology) defines four principal stages (Collection, Examination, Analysis and Reporting), represented presented at the Figure 2.1, each organisation can modify the framework to fit their own context. As an example, the authors in [3] propose the four phases to be namely as **Preparation**, **Acquisition**, **Analysis** and **Presentation**. This shows that there is not a set in stone process and it should be adapted for the specific organization.

In order to understand the 4 phases presented on the "Guide to Integrating Forensic Techniques into Incident Response", is important to explain each one of them.

- **Collection:** The first phase in the process is to identify, label, record, and acquire data from all possible sources of relevant data, while following guidelines and procedures that preserve the integrity of the data. Collection is typically performed in a timely manner due to the high probability of losing dynamic data such as network traffic and connections, or data allocated in non-volatile memory.
- **Examination:** Examination involves forensic processing of the large amount of data collected using a combination of automated and manual methods to evaluate and extract data of particular interest while maintaining data integrity.
- **Analysis:** The next step in the process is to analyze the findings using legally justifiable methods and techniques, to derive useful information that addresses the questions that were the impetus for performing the collection and examination.
- **Reporting:** At the final stage, the measures used are explained, the choice of tools and procedures are explained, and the analysis results are reported to determine the additional measures to be taken (for example, forensic research of additional data sources, identified). Provides recommendations for improving vulnerabilities protection, existing security enhancement controls) and forensic process policies, guidelines, procedures, tools, and other aspects. The format of the reporting step varies greatly depending on the situation.

2.1.1 Related work

Computer forensics is not a new topic, as such several tools and projects have been around for a while. In this subsection some of such products will be introduced.

Autopsy (https://www.autopsy.com/, accessed on 11 June 2022) is an open source, fast, easy-to-use forensic platform capable of analyzing all types of mobile devices and digital storage media. The tool can be extended by the use of modules or add-ons. Some of which are dedicated to law-enforcement usage, some examples are ProjectVic (https://www.projectvic.org/, accessed on 11 June 2022) and C4P. Both examples presented are related to child abuse and sexual traffic, areas where image forensics is of extreme importance since allows for more quickly and effectively identifies victims and perpetrators.

Cellebrite (https://cellebrite.com/, accessed on 11 June 2022) is a Israeli based company dedicated to threat intelligence and the creation and commerce of software and tools that are used in digital forensics investigations from enterprises to government authorities. Some of those tools are very famous due to some big claims done by the company [6]. The most famous product from the company is UFED a product used to bypass mobile devices security and extract data from said devices.

MobSF (https://github.com/MobSF/Mobile-Security-Framework-MobSF, accessed on 11 June 2022) is automated, all-in-one mobile application (Android/iOS/Windows) pen-testing, malware analysis and security assessment framework capable of performing static and dynamic analysis.

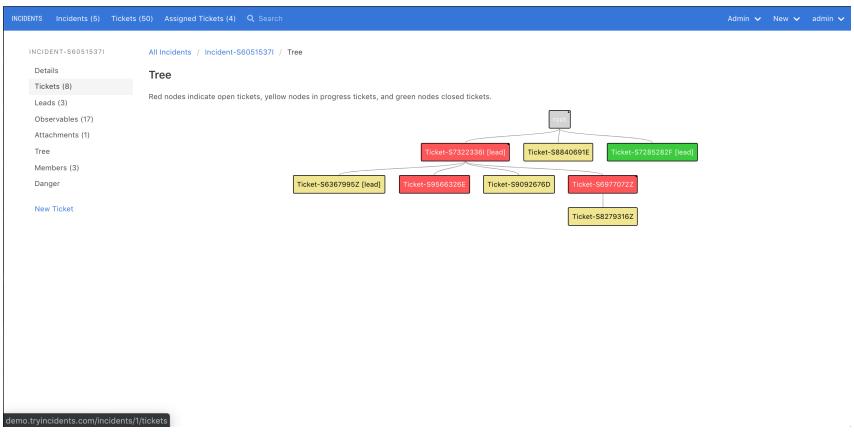

Figure 2.2: Example of a tree in the INCIDENTS software

INCIDENTS (https://github.com/veeral-patel/incidents, accessed on 11 June 2022) INCIDENTS is a web-based tool for incident response, investigation of malware infections, phishing campaign, or any other type of security incident.

2.2 Digital photos and videos

Multimedia files are very familiar to the average computer user: pictures, music, audios, videos, and documents are daily used and consumed in various Internet platforms, such as social networks and institutional websites. For the purposes of the project presented in this dissertation, the focus will be directed towards image and video files. These files can be represented in different formats, being the following the most common:

- Photo file formats: JPEG, GIF, TIFF, BMP
- Video file formats: MP4, AVI, MOV, MPEG-1, MPEG-2, MPEG-4, AVCHD, H.264, H.265

Figure 2.3: Example of a photo [18]

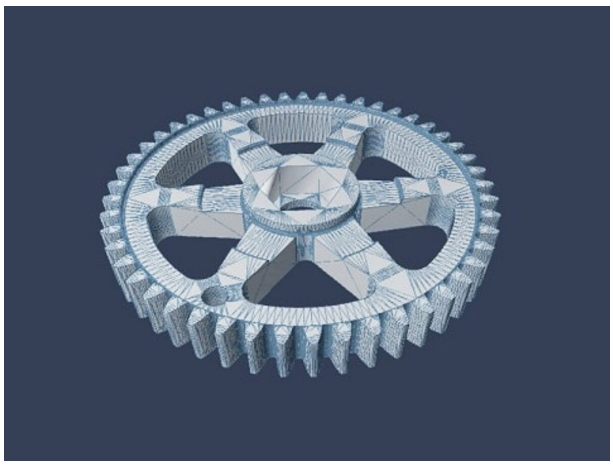

Figure 2.4: Example of a 3D file [66]

An image is a visual representation of something: such as a likeness of an object produced on a photographic material or a picture produced on an electronic display (such as a television or computer screen)[52]. The most common type of image is a static photo, such as the example presented in figure 2.3. Figure 2.4 is also the representation of an object, but with three-dimensional (3D) features instead of only 2D, as represented by Figure 2.3. But not all images

are static, as they can have movement, as represented in videos (will be discussed later in this document). In order to avoid confusion, throughout this dissertation the term "photo" will be used in favor of "images", since the latter can have a broader spectrum of meanings causing confusion.

Figure 2.4 is usually only used to represent 3D objects. This files are heavier since they need to retain all features in case of printing. The specific file type is STereoLithography (STL) [66] which is a 3D printing file format used to store the details about an object, more specifically its shape. The shape can be defined by a mesh composed of triangular faces which are composed of 3 vertices and a normal face to define its orientation.

A digital photo is subdivided into small units called pixels. The word 'pixel' was invented by combining the words 'picture element'. When a photo is visualised on a screen, usually, has hundreds of thousands (and often millions) of pixels — when enough of these colored squares are placed next to one another and displayed at a small enough size, the viewer see continuous images instead of individual pixels. While physical units like inches or centimeters have an exact, real-world size, a pixel is more of a logical unit than a physical one. However, the pixels of digital images are most often displayed at a size so small so as to not be visible, so they usually exist as very small elements [63].

A coloured image can be viewed as three different images (a red scale image, a green scale image and a blue scale image), stacked on top of each other, and when fed into the red, green and blue inputs of a colour monitor, it produces a colour image on the screen[28]. This Red-Green-Blue (RGB) combination allow anyone to visualise a "true colour image" since its colours will be the closest possible representation to real life.

Another important term to learn regarding digital media is *resolution*. It is a measure used to describe the sharpness and clarity of a photo. It is used a lot as a measure of quality of monitors, but is also very important when discussing digital multimedia [69].

Additionally to the information presented by the visual aspect of a photo there are additional data on the files. These data, often called metadata, describe and provide information about rights and administration of a photo. Such details can be stored in the same file as the photo or on a separate one. The example is presented in Figure 2.5, where it is observable information such as file format, photo dimensions and permissions.

There are three main categories of metadata [38]:

- Descriptive – information about the visual content. This may include headline, caption, keywords. Further persons, locations, companies, artwork or products shown in the image. This can be done using free text or codes from a controlled vocabulary or other identifiers.
- Rights – identification of the creator, copyright information, credits and underlying rights in the visual content including model and property rights. Further rights usage terms and other data for licensing the use of the image.

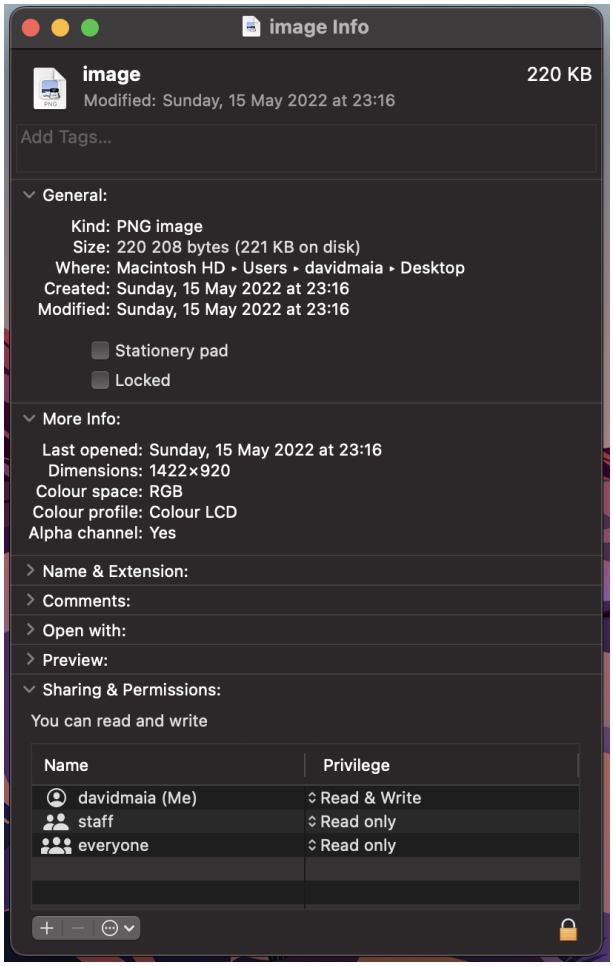

Figure 2.5: Metadata of a photo extracted with MacOS tool

- Administrative – creation date and location, instructions for the users, job identifiers, and other details.

Video is an electronic media for recording, copying, reproducing, broadcasting and displaying moving visual media [40]. 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, similar to what is available today. 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.

An important concept when considering video is frames. The frame is a single image in a sequence of pictures which compose the complete moving video. In general, one second of a video is comprised of 24 or 30 frames per second also known as frames per second (fps) [16] (although 60 fps is becoming very popular for some use cases). 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.

2.2.1 Related work

Most conventional and common methods of image retrieval utilize some method of adding metadata such as tokens, captioning , keywords, or descriptions to the images so that retrieval can be performed over the annotation words. However, the authors in [10] describe a different method that relies on actual content of the image, named Content-Based Image Retrieval (CBIR).

2.3 Multimedia manipulation techniques

Digital media is the most consumed form of media available. Photos and video are a prime way of dispersing misinformation. The ease of trick someone with a manipulated photo or video is so high that it led to the creation of several types of forgery. Since the focus of this project and dissertation is detecting this type of manipulations so a firm grasp of the fundamental concepts is necessary. There are four major types of media manipulation:

- Copy-Move
- Splicing
- Retouching
- Resampling

2.3.1 Copy-Move

Copy-move forgery is the process of copying and pasting from one area of the photo to another part within the same photo. As shown at Figure 2.6 this technique can be simply implemented but can be very effective to confuse the viewer. This photo is an actual representation on how a country used

The simplicity of presenting a repetition of a certain element (or elements) from the same photo is the fundamental reason for the rise in copy-move manipulation. Textured elements, such as grass, leaves, gravel, or cloth with random patterns, are good for this since the duplicated regions will likely fade into the backdrop, making any suspicious artifacts difficult to see. The noise, color palette, dynamic range, and other crucial qualities of the duplicated bits will be consistent with the remainder of the photo because they come from the same digital photo. Methods that seek for statistical measurement incompatibilities in different sections of the photo can't identify this form of modification.

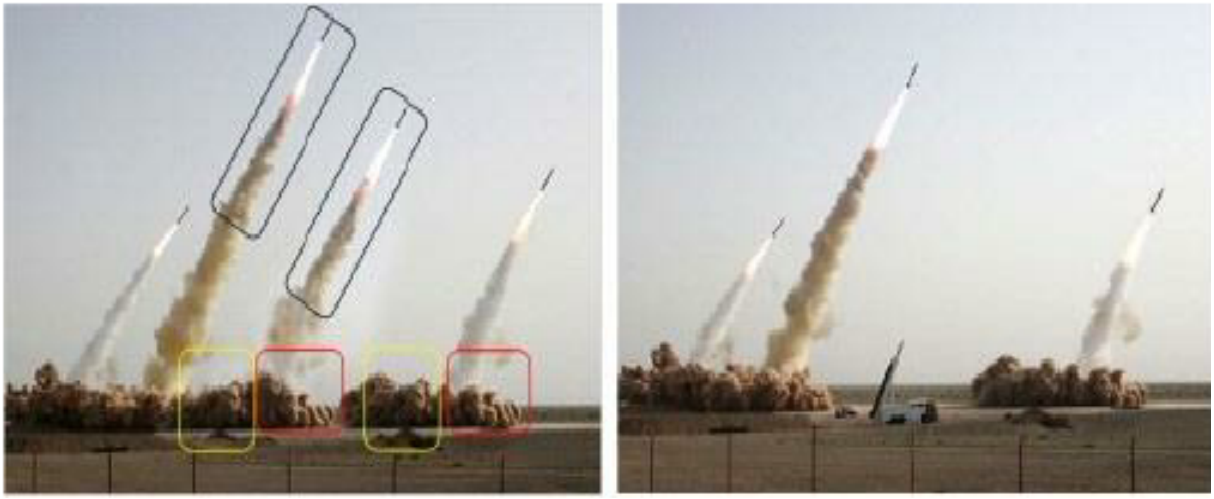

Figure 2.6: Example of Copy-Move forgery: (left) Original photo, (right) Tampered photo [33]

Figure 2.7: Example of Splicing: (a) Original photo, (b) Tampered photo[39]

Powerful digital media editing tools made it possible to produce good quality forgeries for almost anyone [5]. In fact, copy-move forgery can be done very easily by using the tools such as Cloning in Photoshop. Some of these available tools will be discussed later in this document.

2.3.2 Splicing

Splicing is an photo editing method to copy a part of an photo and paste it onto another photo, and it is commonly followed by post-processing such as local/global blurring, compression, and resizing. It is often used as an initial step of photo-montage, which is very popular in digital image content editing. The splicing tampered photo could be used in news reports, photography contest, key proof in the academic papers, and so on, which could bring certain negative influences.

In the splicing process, the manually introduced transitions of edges and corners are different from those in the natural photos. The differences are commonly described by the inconsistency

and abnormality, and they are used for splicing detection [76]. As clearly visible in Figure 2.7, the tiger doesn't belong to the original photo, but is easily perceptible by the viewer since the shadow of the tiger is non-existent, among other characteristics.

Compared to Copy-Move, Splicing has an added degree of difficulty. Since the added element comes from an image with a different "look and feel", meaning texture, lighting, color pallet and other factor. This makes the forgery easier to detect if the post-processing of the forgery isn't done properly in an effort to improve linearity throw out the image.

There is a special and more advanced type of splicing that's called Deepfake, that will be addressed later in another subsection.

2.3.3 Retouching

Retouching is probably the most common type of manipulation done to photos or videos. The amount of tutorials available is enormous, since this manipulation is not used necessarily for tricking, or inducing someone in error, but also for commercial and aesthetic applications. Figure 2.3.3 is an example of retouching, changing the physical appearance of the model face, removing acne or other unwanted marks, making it more appealing.

Figure 2.8: Example of Retouching: (left) Original photo, (right) Tampered photo[57]

Photo retouching is considered a forgery of a minimally harmful form of the digital photo. An original photo does not change substantially but certain aspects of the original image have been reduced. Therefore its detection difficulty might be directly connected with the amount of retouching performed on a photo, or video.

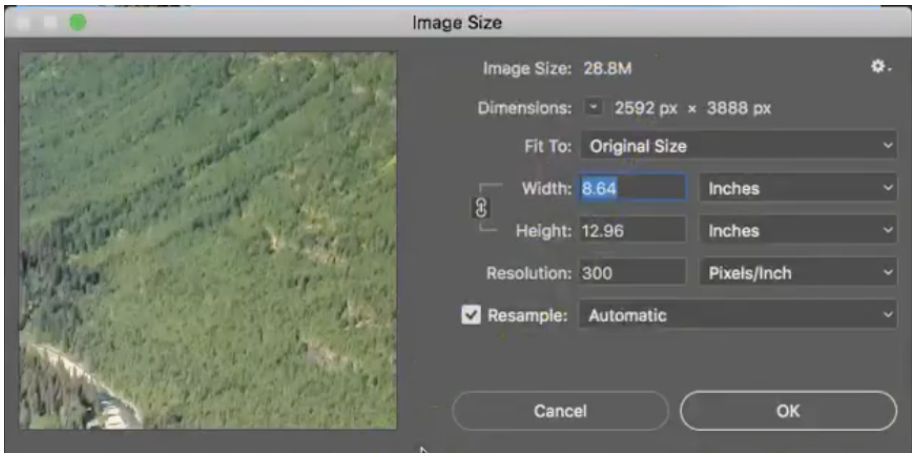

Figure 2.9: Example of Resampling forgery using Photoshop

2.3.4 Resampling

Resampling is the process of altering an images resolution by either adding or removing extra pixels. Increasing the number of pixels in an photo is often referred to as upsampling, reducing the number of pixels is referred to as downsampling. The actual process of resampling a photo requires the application of a mathematical algorithm to the photo file, once the algorithm has been applied the resulting can be rendered for viewing [56].

Figure 2.9 is an example of a configuration performed in Photoshop, for automatic resampling of a forest photo. This exemplifies how simple it is to perform this kind of manipulation with easily available software.

2.3.5 Deepfake

Deepfake: The evolution and development of deep learning technology, along with it's conjugation with forgery techniques, created deepfakes (stemming from "deep learning" and "fake") [75]. These techniques that can superimpose face images of a target person onto a video of a source person to make a video of the target person doing or saying things the source person does. This constitutes a category of deepfakes, namely faceswap. In a broader definition, deepfakes are artificial intelligence-synthesized content that can also fall into two other categories, i.e., lip-sync and puppet-master. Lip-sync deepfakes refer to videos that are modified to make the mouth movements consistent with an audio recording. Puppet-master deepfakes include videos of a target person (puppet) who is animated following the facial expressions, eye and head movements of another person (master) sitting in front of a camera [58].

While some deepfakes can be created by traditional visual effects or computer-graphics approaches, the recent common underlying mechanism for deepfake creation is deep learning models such as autoencoders and generative adversarial networks (GANs), which have been applied widely in the computer vision domain. These models are used to examine facial expressions

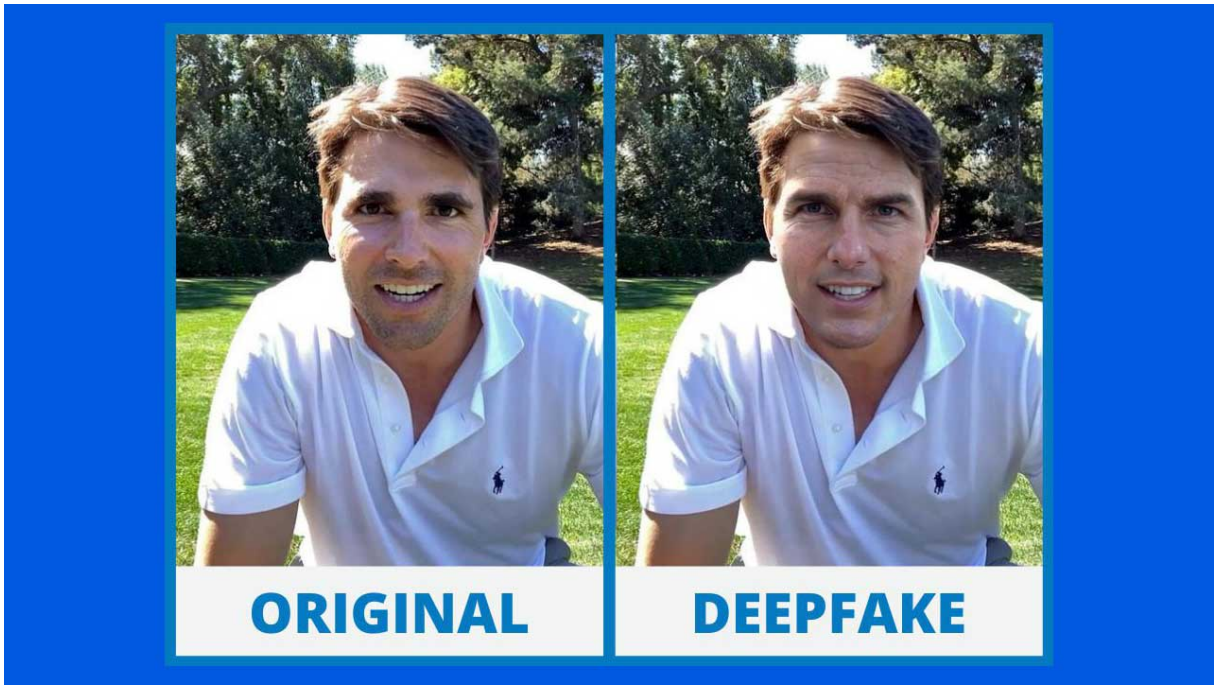

Figure 2.10: Example of Deepfake: (left) Original photo, (right) Tampered photo[60]

and movements of a person and synthesize facial images of another person making analogous expressions and movements [58].

Deepfake methods normally require a large amount of image and video data to train models to create photo-realistic images and videos. As public figures such as celebrities and politicians may have a large number of videos and images available online, they are initial targets of deepfakes. This type of manipulation was already used to swap faces of celebrities or politicians to bodies in porn images and videos. The first deepfake video emerged in 2017 where face of a celebrity was swapped to the face of a porn actress [12]. The potential for this technology to be employed for malicious purposes that can be a threat to world security, specially since they can be employed to create videos of world leaders with fake speeches for falsification purposes [74]. Deepfakes therefore can be abused to cause political or religion tensions between countries, to trick society and affect behaviours or important results in elections, just to mention a few examples.

Deepfakes have become popular due to the quality of tampered videos and also the easy-to-use ability of their applications to a wide range of users with various computer skills from professional to novice. These applications are mostly developed based on deep learning techniques. Deep learning is well known for its capability of representing complex and high-dimensional data. One variant of the deep networks with that capability is deep autoencoders, which have been widely applied for dimensionality reduction and image compression. The first attempt of deepfake creation was FakeApp[1], developed by a Reddit user using autoencoder-decoder pairing structure. In that method, the autoencoder extracts latent features of face images and the decoder is used to reconstruct the face images. To swap faces between source images and target images, there

[1]https://www.malavida.com/en/soft/fakeapp/

is a need of two encoder-decoder pairs where each pair is used to train on an image set, and the encoder's parameters are shared between two network pairs. In other words, two pairs have the same encoder network. This strategy enables the common encoder to find and learn the similarity between two sets of face images, which are relatively difficulty to challenge because faces normally have similar features such as eyes, nose, mouth positions [58].

The availability of multimedia software for editing is increasingly high. The multimedia manipulation can be generally achieved by using software applications, such as Adobe Photoshop (https://www.adobe.com/pt/products/photoshop.html, accessed on 29 May 2022), or GNU Gimp (https://www.gimp.org/, accessed on 29 May 2022), for photo manipulation, and Adobe After Effects (https://www.adobe.com/products/aftereffects.html, accessed on 29 May 2022), or Hitfilm Express (https://fxhome.com/product/hitfilm-express, accessed on 29 May 2022) for video manipulation.

These tools are evolving and AI adds a new layer to traditional photo editing software by taking repetitive, manually intensive tasks, learning what we want and helping us achieve it more easily. Complex work is performed more quickly and automation reduces human errors and thus increasing the difficulty of detecting media manipulations [51].

2.3.6 Manipulation vs Fake media

During the development of this project a common question was raised "What classifies an image as fake?". The reason for this question is essential before proceeding with any further research, planning or even developing. Nowadays, a lot of media is captured by smartphones and using applications that facilitate the usage of filters. These can pose a problem, since the usage of some filters is legitimate and the image shouldn't be classified as fake, since the author actually intended that the photo/video was altered in that manner.

For example, if a user takes a photo of someone (or an object) and adds a filter that changes the photo to black and white, this doesn't mean that the image is fake since the manipulation wasn't performed with a deceiving or fraudulent intention. This is an actual observation to previous work on this field since didn't include considerations of this important premises: a lot of content nowadays is produced in smartphones; photos taken using by smartphones resort to apps with filter capabilities; although these filters are used their intention is non-malicious.

Solving the presented issue is very hard, for this project the decision was made that a photo would be considered fake when altered/manipulated in a way that changes the it's original subject, adds/removes elements not in the original photo, or is changed in a way were the original intention of the photo is deceived. This means that a photo whose only change was a sepia filter would still be considered real, but if text is added then it should be considered fake.

2.3.7 Related Work

This subsection aims at presenting work (projects, literature, or other forms) done by others that is related with media manipulation, ranging from project to forge faces of people that don't exist to previous research into the topic of manipulation detection.

2.3.7.1 Manipulation

Manipulation is a growing field, being for the business opportunities, research purposes, hobby projects or other purposes. Independent of the reason a lot has been achieve on this field. One of the most impressive techniques is Generative Adversarial Networks (GANs) are an approach to generative modeling using deep learning methods, such as convolutional neural networks. The specifics of CNNs and GANs will be adressed later in this chapter, however it's crucial to go over projects that already take advantage of these.

Figure 2.11: Example of Deepfake created at thispersondoesnotexist.com [70]

Created by Philip Wang [59], using research released by chip designer and manufacturer Nvidia, This Person Does Not Exist (http://www.thispersondoesnotexist.com, accessed on 12 May 2022) allows any user to create an endless stream of fake portraits. The algorithm behind it is trained on a huge dataset of real images, then GANs to fabricate new examples. The premise of usage is simple, each time an user refreshes the website a new portrait is generated and appears

to the user, similar to the one presented by Figure 2.11.

The algorithm used is StyleGAN [41] and although this version of the model is trained to generate human faces, the project is open (https://github.com/NVlabs/stylegan, accessed on 21 June 2022) so anyone can create their own version. There are already projects for art, horses and cats.

2.3.7.2 Manipulation detection

In an effort to create a different methodology for detecting manipulated media, in [46] the authors tried a different approach.

2.3.7.3 Existing datasets

Datasets with clear labels of manipulated and not manipulated photos are hard to find. In thta sense, in the previous work developed in [23], the authors created a dataset with some images and videos that can be used to train machine learning models that enable us to benchmarch those models and understand which ones work best.

Deepfake Detection Challenge (DFDC) was a challenge launched in December 2020, and 2,114 participants submitted more than 35,000 models to the competition. The company in charge of the callenge, released the dataset to allow AI researchers develop "new generation and detection methods to advance the state of the art in this field. Moreover, this dataset will be opened for use for other research work in AI domains as well as work on deepfakes." [53]

2.4 Machine Learning and Artificial Intelligence

Machine Learning (ML) is defined as the study of computer programs that leverage algorithms and statistical models to learn through inference and patterns without being explicitly programmed. Machine Learning field has undergone significant developments in the last decade[2]. Before going in-depth about the different types of machine learning methods, it is important to understand some key concepts about classification problems.

The two major prediction problems in the ML and data mining fields are Classification and Regression. These are approached in a very different manner so it's important to distinguish them to select the best algorithm.

Classification problem is the process of finding or discovering a model or function which helps in separating the data into multiple categorical classes, that is discrete values. In classification, data is categorized under different labels according to some parameters given in input and then the labels are predicted for the data. The derived mapping function could be demonstrated in the form of "IF-THEN" rules. The classification process deals with the

problems where the data can be divided into binary or multiple discrete labels. On the other hand, a **Regression problem** is the process of finding a model or function for distinguishing the data into continuous real values instead of using classes or discrete values. It can also identify the distribution movement depending on the historical data. Because a regression predictive model predicts a quantity, therefore, the skill of the model must be reported as an error in those predictions [29].

After analysis, it's possible to identify that the problem presented in this dissertation is a classification problem. The main goal is two classify an image, or video, into one of two categories: manipulated or non-manipulated media (abbreviated to labels fake or real). Being the classification between only two labels, we may term it, without loss of clarity, as a binary classification problem.

Another important concept is the **Imbalanced problem**, which is an example of a classification problem where the distribution of examples across the known classes is biased or skewed. The distribution can vary from a slight bias to a severe imbalance, where there is one example in the minority class for hundreds, thousands, or millions of examples in the majority class or classes [8].

Imbalanced classifications pose a challenge for predictive modeling as most of the machine learning algorithms used for classification were designed around the assumption of an equal number of examples for each class. This results in models that have poor predictive performance, specifically for the minority class. This is a problem because, typically, the minority class is more important and therefore the problem is more sensitive to classification errors for the minority class than the majority class [8]. This concept is very important since fraud problems are generally very imbalanced problems.

There are three main types of machine learning strategies:

- Supervised Learning - Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data.
- Unsupervised Learning - Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.
- Reinforcement Learning - It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged or 'reinforced', and non-favorable outputs are discouraged or 'punished'.

This project is going to focus on supervised learning, as the instances enclosed in the dataset are initially classified in two classes: real and fake. The ML method employed in the research is based on Support Vector Machines and was previously evaluated in [23].

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human

brain, albeit far from matching its ability to "learn" from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy. Deep learning distinguishes itself from classical machine learning approach by the type of data that it works with and the methods in which it learns. Also eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on manual pre-processing [37]. Later in this section we will analyze Convolutional Neural Networks (CNN), a specific algorithm that is very used for image-based problems.

2.4.1 Support Vector Machine

Support Vector Machine (SVM) is a ML kernel-base non-parametric supervised learning method and has been successfully used when applied to solving classification problems, specifically when applied to binary classification (such as the problem of classifying media as real or fake).

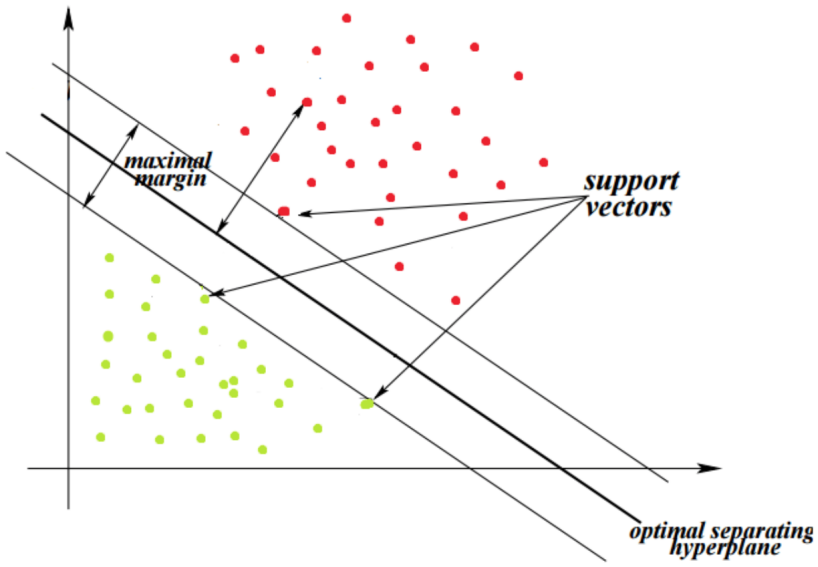

Figure 2.12: SVM graphic [30]

For non-linear classification and regression, they use the kernel trick to map inputs to high-dimensional feature spaces. SVMs construct a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. Figure 2.12 shows the decision function for a linearly separable problem, with three samples on the margin boundaries, called "support vectors"[47].

Figure 2.12 presents a visual representation of Support Vectors, and how they are equidistant from the hyperplane and help in building the SVM. Support vectors are called so because if their position shifts, the hyperplane shifts as well. That means the hyper-plane only depends on the position of support vectors[45].

SVM-based manipulation detection is the approach also taken by this dissertation, the reason being the high correlation with the work developed by the researcher Sara Ferreira, FCUP, [23] developed a solution adapted to Autopsy Modules (https://www.autopsy.com/, accessed on 12 June 2022) with focus on forensics detection of photos/videos manipulation. It showed promising results, with a mean F1-Score (an evaluation metric that will be discussed in chapter 4) of 99.68% on the detection of manipulated images and 84.15% on the detection of manipulated videos.

2.4.2 CNN

Convolutional Neural Network (CNN/ConvNet), represented in Figure 2.13, is a well-known deep learning architecture inspired by the natural visual perception mechanism of the living creatures. In 1959, Hubel & Wiesel [36] found that cells in animal visual cortex are responsible for detecting light in receptive fields. Inspired by this discovery, Kunihiko Fukushima proposed the neocognitron in 1980 [27], which could be regarded as the predecessor of CNN. In 1990, LeCun et al. [13] published the seminal paper establishing the modern framework of CNN, and later improved it in [48]. They developed a multi-layer artificial neural network called LeNet-5 which could classify handwritten digits. Like other neural networks, LeNet-5 has multiple layers and can be trained with the backpropagation algorithm [34]. It can obtain effective representations of the original image, which makes it possible to recognize visual patterns directly from raw pixels with little-to-none preprocessing. A parallel study of Zhang et al. [77] used a shift-invariant artificial neural network (SIANN) to recognize characters from an image. However, due to the lack of large training data and computing power at that time, their networks can not perform well on more complex problems, e.g., large-scale image and video classification[31].

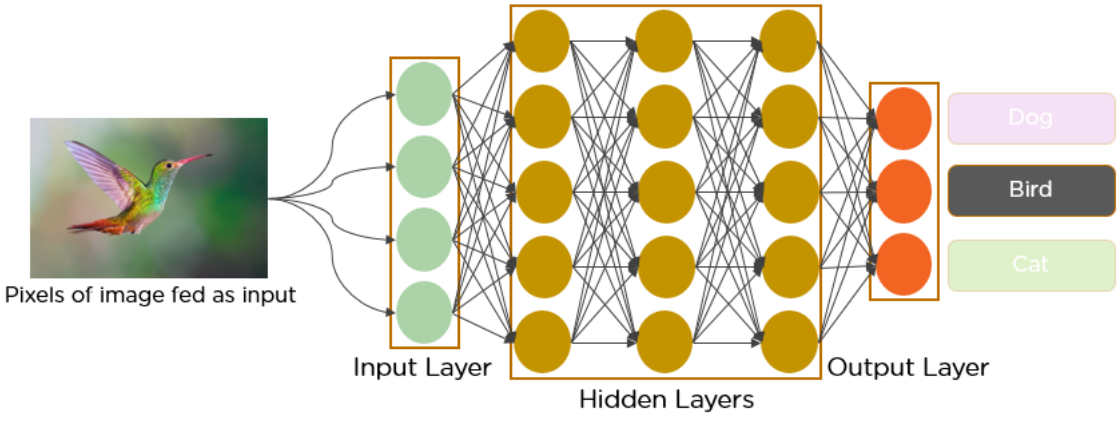

Figure 2.13: CNN representation [49]

Since 2006, many methods have been developed to overcome the difficulties encountered

in training deep CNNs. Most notably, Krizhevsky et al.proposed a classic CNN architecture and showed significant improvements upon previous methods on the image classification task. The overall architecture of their method, i.e., AlexNet[65], is similar to LeNet-5 but with a deeper structure. With the success of AlexNet, many works have been proposed to improve its performance[31].

The most relatable work with CNNs for media manipulation is StyleGAN [41] a generator architecture makes it possible to control the image synthesis via scale-specific modifications to the styles. We can view the mapping network and affine transformations as a way to draw samples for each style from a learned distribution, and the synthesis network as a way to generate a novel image based on a collection of styles. The effects of each style are localized in the network, i.e., modifying a specific subset of the styles can be expected to affect only certain aspects of the image.

Also using CCNs, the researchers of the paper *Image Manipulation Detection using Convolutional Neural Network* [43] proposed an image manipulation detection algorithm using deep learning technology. The model based on a convolutional neural network (CNN) is designed. Especially, a high pass filter is used to acquire hidden features in the image rather than semantic information in the image. The convolutional layer is composed of 2 layers having maximum pooling, ReLU activation, and local response normalization. The fully connected layer is composed of 2 layers. According to their conclusions "at least 95% accuracy was achieved".

2.4.3 PCA

Large datasets are increasingly common and are often difficult to interpret. Principal Component Analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance. Finding such new variables, the principal components reduce to solving an eigenvalue/eigenvector problem, and the new variables are defined by the dataset at hand, not *a priori*, hence making PCA an adaptive data analysis technique. PCA is adaptive in another sense too, since variants of the technique have been developed that are tailored to various different data types and structures[68].

After analysing a visual representation of PCA, using Figure 2.14, it is possible to see it as a technique that finds the directions of maximal variance. This way it is possible to represent a multivariate data table as smaller set of variables (summary indices) in order to observe trends, jumps, clusters and outliers.

PCA will be used in this project to experiment reducing the number of features extracted thus reducing the running times without decreasing the evaluation metric scores (will be discussed in chapter 4).

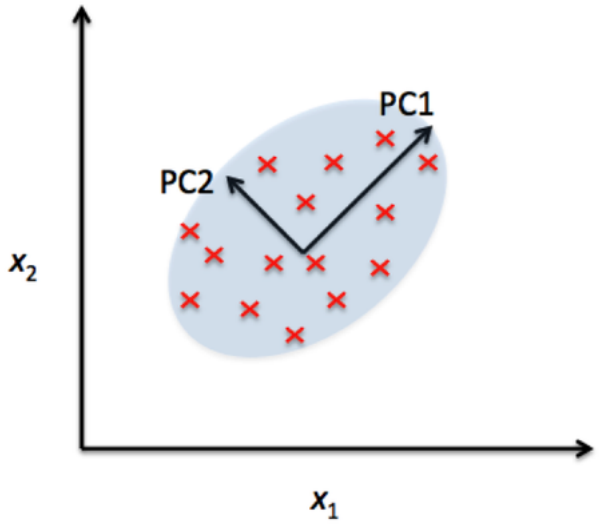

Figure 2.14: PCA [64]

2.5 Fourier Transformation

The Fourier Transform takes a time-based pattern, measures every possible cycle, and returns the overall "cycle recipe" (the amplitude, offset and rotation speed for every cycle that was found)[21]. Although initially created and applied to sinusoidal signals or wave-forms, the process relies on filters to determine the function. For the purposes of this document, the focus will be on two particular types of Fourier Transformations, namely Discrete Fourier Transformation (DFT) and Fast Fourier Transformation (FFT).

2.5.1 DFT

Discrete Fourier transform (DFT) is a frequency domain representation of finite-length discrete-time signals. It is also used to represent FIR discrete-time systems in the frequency domain. As the name implies, DFT is a discrete set of frequency samples uniformly distributed around the unit circle in the complex frequency plane that characterizes a discrete-time sequence of finite duration. Because DFT is a finite set of frequency samples, it is a computational tool to perform filtering and related operations [71].

As explained previously, Fourier transformations are mostly applied to sinusoidal signals (or wave-forms), but in this project DFT will be applied to images feature extraction. The process will be explained in the next chapters.

2.5.2 FFT

There is an efficient algorithm known as the Fast Fourier Transform (FFT) to perform filtering of long sequences, power spectrum estimation, and related tasks. FFT is a discrete Fourier transform algorithm which reduces the number of computations needed for N points from $2N^2$ to $2NlgN$, where lg is the base-2 logarithm.

FFTs were first discussed by Cooley and Tukey (1965)although Gauss had actually described the critical factorization step as early as 1805 (Bergland 1969, Strang 1993). A discrete Fourier transform can be computed using an FFT by means of the Danielson-Lanczos lemma if the number of points N is a power of two, otherwise a transform can be performed on sets of points corresponding to the prime factors of N which is slightly degraded in speed. An efficient real Fourier transform algorithm or a fast Hartley transform (Bracewell 1999) gives a further increase in speed by approximately a factor of two. Base-4 and base-8 fast Fourier transforms use optimized code, and can be 20-30% faster than base-2 fast Fourier transforms. prime factorization is slow when the factors are large, but discrete Fourier transforms can be made fast for $N = 2, 3, 4, 5, 7, 8, 11, 13, and 16$ using the Winograd transform algorithm (Press et al. 1992, pp. 412-413, Arndt).

Fast Fourier transform algorithms generally fall into two classes: decimation in time, and decimation in frequency. The Cooley-Tukey FFT algorithm first rearranges the input elements in bit-reversed order, then builds the output transform (decimation in time). The basic idea is to break up a transform of length N into two transforms of length N/2 using the identity

Chapter 3

Architecture and Development

This chapter will focus not only on the conceptual planning but also on the practical approach of the project. The chapter is divided, on a macro level, into two main sections: architecture and development.

3.1 Architecture

This section will focus on the architecture of the entire project. Figure 3.1 presents a visual representation of the architecture for the framework that will be used to test the various pipeline combinations. This can be described as a pipeline, a set of steps followed one after another that take the initial set of media files and end up with a trained model to be tested at the end.

This process is applicable to the initial creation of the model to be used. As such does not apply to each individual file that will be using the module to obtain a classification regarding the manipulation.

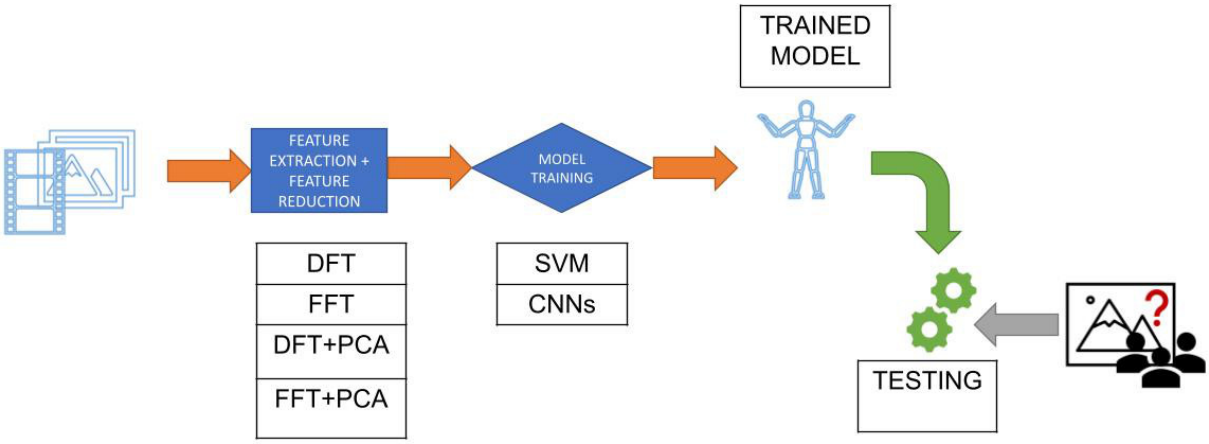

Figure 3.1: Diagram of pipeline architecture

The steps presented in Figure 3.1 are feature extraction, feature reduction, model training

and testing.

3.1.1 Feature Extraction

Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. These variables require a lot of computing resources to process. So Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus, effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with accuracy and originality.

The technique of extracting the features is useful when you have a large data set and need to reduce the number of resources without losing any important or relevant information. Feature extraction helps to reduce the amount of redundant data from the data set.

Image processing is one of the best and most interesting domain. In this domain basically you will start playing with your images in order to understand them. So here we use many many techniques which includes feature extraction as well and algorithms to detect features such as shaped, edges, or motion in a digital image or video to process them.

In the end, the reduction of the data helps to build the model with less machine effort and also increases the speed of learning and generalization steps in the machine learning process [11].

As discussed in previous chapters, this project data is mainly photos. Photos require different methods for feature extraction, so that data can be comparable and without losing essential properties that define the photo.

3.1.2 Feature Reduction

Feature reduction, also known as dimensionality reduction, is the process of reducing the number of features in a resource heavy computation without losing important information. Reducing the number of features means the number of variables is reduced making the computer's work easier and faster. Feature reduction can be divided into two processes: feature selection and feature extraction. There are many techniques by which feature reduction is accomplished. Some of the most popular are generalized discriminant analysis, AutoEncoders, non-negative matrix factorization, and principal component analysis.

3.1.3 Model Training

Model training is the phase in the data science development life cycle where practitioners try to fit the best combination of weights and bias to a machine learning algorithm to minimize

a loss function over the prediction range. The purpose of model training is to build the best mathematical representation of the relationship between data features and a target label (in supervised learning) [9].

3.1.4 Testing

The final stage of is testing, which can be divided in two ways: classification of a specific photo/video or benchmark a pipeline combination. For the purposes of classifying a specific media file, the user must provide the image/video to be classified and will obtain the probability of that being manipulated. For benchmark, the dataset is automatically divided, and the tests are performed to obtain metric scores that allow us to determine how accurate is the entire pipeline.

3.2 Development

This section addresses the development of the solution from the dataset creation to the development and launch of the python package.

3.2.1 Datasets

Having the right dataset is extremely important to further develop the solution. So using the already available dataset like COVERAGE and combining them was an evident solution. However, the datasets didn't contain every type of media needed so another measures and processes had to be taken.

3.2.1.1 Manual Classification

The abundance of datasets already classified as "manipulated" or "not manipulated" (also named in this thesis as "real" of "fake") is indisputable. However there are not many datasets that have content from social media already classified into the intended binary categories. Presented in previous sections, where the problem was described, there is a clear need to train our model with Social Media Content, so it's clear the need for a well classified base for this and similar projects. Preemptive search showed that there aren't any available datasets that filled out our search parameters. In an approach to solve this issue and contribute to the enrichment of this field, we decided to classify an existent dataset into our intended binary categories. The chosen dataset was InstaCities1M [32], which curates a total of 1 million pictures from the popular social network Instagram (Property and trademark of Meta),

3.2.1.2 thispersondoesnotexist

For the reasons presented previously in this dissertation, "This person does not exist" dataset has characteristics identified as a well done manipulation. Since this dataset cannot be downloaded all together and manual download would take a enormous amount of time, the bash script below was created to perform automatic download and save 10000 generated photos.

```
#! /bin/bash

for i in {1..10000}
do
    wget "https://thispersondoesnotexist.com/image" -nv -O fake/tpdne_$i.jpg
    sleep 1
    echo $i
done
```

3.2.1.3 Adding filters to images

Filters are very used nowadays, social media apps ,such as Instagram, use them directly into the camera. The usage of these filters does not automatic translate to an attempt to manipulate the photo (or video) with malicious intent. To ensure that filters are an elements to have in consideration when training models, datasets (or subsets at least) with filters needed to be included in our final dataset.

The difficulty of getting datasets, or even taking enough photos to create a new big one, caused the need for another solution. The resulting idea was to edit photos from available datasets in a format that emulates what would happen if taken by a smartphone filters application.

```
from PIL import Image,ImageFilter, ImageEnhance

import os
i = 0

dirA = 'images/real'
dirO = 'mixed/real/'

for filename in os.listdir(dirA):
   i = i + 1
   print("Filename: ", filename)

   if i < 3427:img = Image.open(dirA+'/'+filename)

   if i > 3426 and i < 6855:
      img = Image.open(dirA+'/'+filename).convert('L')
```

```
    else:
        im = Image.open(dirA+'/'+filename)
        img = ImageEnhance.Contrast(im).enhance(1.8)

    img.save(dirO + filename)

    print("Image number: ",i)
```

Listing 3.1: Python code to mixture the photos to add filters

3.2.1.4 Final Dataset

The final dataset composition, subsets used to create this final dataset, can be verified in Table 4.6.

Table 3.1: Final dataset composition

Dataset	Manipulated	Real	Type of Content
thispersondoesnotexist	10000	0	Photos
COVERAGE dataset	100	100	Photos
Flickr-Faces-the-dataset	0	10000	Photos
Columbia Image Splicing	180	183	Photos
Insta 1M Cities (Subset 100k NYC)	138	0	Photos
Created for this project	2	9	Photos
Facebook's DFDC	42	54	Videos

Table 3.2: Final dataset distribution

Type of Media	Manipulated	Real
Photos	10420	10292
Video	42	54
Total	10462	10346

Table 3.3: Final dataset filters

Filter Type	Manipulated	Real
Original	3427	3427
Black and White	3428	3428
Enhancement	3607	3491

3.2.2 Problems faced during development

As with any project, the development of the solution was not immune to problems and challenges along the way. One of the biggest issues was using video files, the size of the files caused a lot of problems with the limited computing power available so the decision was made to focus the development using only photos and assume this as something to improve in future developments.

3.2.3 Coding the python package

Since the project is a continuation of the work developed by the researcher Sara Ferreira[24], much of the code was based on what's available, but adapted to match the additional features needed plus the restructure so it can be used in a package.

3.2.3.1 DFT and FFT implementation

Implementing DFT and FFT in python can be achieved by several libraries, for this project it was used the Numpy Library (https://numpy.org/, accessed on 19 June 2022). This library that provides a lot of resources for scientific computing in Python. An additional point in favor is the fact that was used already in one of the projects analyzed[24].

```
if pre_processing == "fft":
    f = np.fft.rfft2(img)
else:
    f = np.fft.fft2(img)

fshift = np.fft.fftshift(f)
```

Listing 3.2: Python code for DFT and FFT

Listing 3.2 presents an excerpt of the python code, where it's possible to see the "IF-THEN" were depending on the type of pre-processing (between DFT and FFT) the respective Numpy code is executed.

3.2.3.2 SVM

SVM implementation was possible thanks to libraries made available by sckikit-learn [47], an excerpt of the code is possible to see in Listing 3.3.

```
from sklearn.svm import SVC

SVM = svclassifier_r.score(X_test, y_test)
y_score= clf.predict_proba(X_test)
x_decision = svclassifier_r.decision_function(X_test)
x_pred = svclassifier_r.predict(X_test)
```

Listing 3.3: Python code for SVM

3.2.3.3 CNN

The implementation of the CNN used the libraries Tensorflow and Keras. This development did not differ significantly from the implementation done by researcher Sara Ferreira [23]. Upon

further research using the resources from the libraries developers revealed that no changes were needed, other than the ones necessary to include it in the package.

3.2.3.4 Calculation of evaluation metrics

The python code was developed to include a way to calculate the evaluation metrics that will be used to assess the different approaches to the problem, this takes advantage of the library *scikit-learn* for Python (https://scikit-learn.org/stable/, accessed on 19 June 2022).

```
from sklearn.metrics import confusion_matrix, classification_report

print(classification_report(y_test, x_pred))
print("")
print("confusion matrix")
print(confusion_matrix(y_test, x_pred))
print("")
print("True Positives: ", confusion_matrix(y_test, x_pred)[0][0])
print("False Negatives: ", confusion_matrix(y_test, x_pred)[0][1])
print("False Positives: ", confusion_matrix(y_test, x_pred)[1][0])
print("True Negatives: ", confusion_matrix(y_test, x_pred)[1][1])
```

Listing 3.4: Python code for calculation of evaluation metrics

Listing 3.4 presents the implementation of the library in this project code, with the *y_test* representing the testing set and *x_pred* are the results given by the ML model. This way is possible to obtain the evaluation metrics necessary: precision, recall, F1score and confusion matrix.

3.2.4 Python Package

Contributing for the development of the computer forensics field, specifically for the detection of manipulated media content, is one of the main goals for this dissertation. The release of a free package, available to be used by other developers, researchers or even hobbyists to create tools that further develop tools. Tools that will benefit and enhance the detection of manipulated images and videos.

The code language chosen for this project is python, this is, in no small part, related with the fact that there are widely available, and easy to install packages for implementations such as SVM, PCA, DFT, FFT, among others.

The definition of a Distribution Package, according to the python foundation, is "A versioned archive file that contains Python packages, modules, and other resource files that are used to distribute a Release. The archive file is what an end-user will download from the internet and install."[25]

Figure 3.2: PyPi logo (https://pypi.org/)

A distribution package is more commonly referred to with the single words “package” or “distribution”, but this dissertation may use the expanded term when more clarity is needed to prevent confusion with an Import Package (which is also commonly called a “package”) or another kind of distribution (e.g. a Linux distribution or the Python language distribution), which are often referred to with the single term “distribution”.

The most common platform to obtain those packages is Python package index (https://pypi.org/, accessed on 16 June 2022), commonly known as PyPi, represented by the logo of Figure **??**. Installing modules is as simple as executing the command below on the command line:

```
pip install <package-name>
```

Where <package-name> is the name of the package intended to be installed.

3.2.4.1 Creation of Python package and publish it

One of the important deliverable for this project, as explained previously, is the python package. This section will present the practical steps taken to create the python package and publish it in PyPi in order to make it accessible.Figure 3.3 presents the file structure of the package created during this project, this will serve as a reference for this section, and for explanation on how to create a python package and publish it, not as a fixed structure for future upgrades.

The first step was to **create a file named *___init___.py*** and save it into the same folder where the code is, as presented by Figure 3.3. This file allows Python to identify that the folder contains a Python package. The code inside the *___init___.py* file gets run whenever you import a package inside a Python program. In this case, the *___init___.py* file is importing ManipulationDetector class from the ManipulationDetector module.

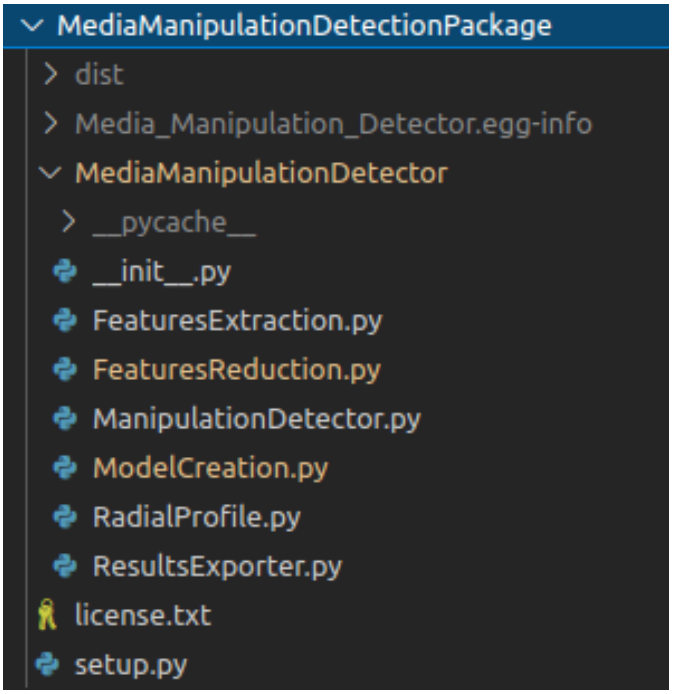

Figure 3.3: Python package file structure

```
from .ManipulationDetector import ManipulationDetector
```

The next file that needs to be created is **the *setup.py* file**. This important file contains information about the package. Every time anyone runs the command pip, it will automatically look for the *setup.py* file.

```
import setuptools
setuptools.setup(
    name="Media Manipulation Detector",
    version="0.0.2",
    author="David Maia",
    description="Module created to detect manipulated media",
    packages=["MediaManipulationDetector"]
)
```

Listing 3.5: Media Manipulation Detector package *setup.py* file

At this point it's possible to **install your package locally** the command below should be executed in the folder that contains the package. The dot indicates to look for a *setup.py* file in the current folder and install the package in there.

```
pip install .
```

From this point on-wards it's possible to run the Python package anywhere on the local system.

Creating the License and README files is optional, but still important. They are created in the directory containing the package, *License.txt* will tell users the terms and conditions under which they can use the package. For simplicity, the MIT license[55] was adapted to this package. *README.md* is a file created to share information with potential users.

Now all needed files are created, so the nest step is to **generate the source distribution**. To do so, the command below installs the latest version of the setuptools package:

```
python -m pip install --user --upgrade setuptools
```

In the same directory where the setup.py file is, this command needs to be executed:

```
python setup.py sdist
```

This created a new folder *dist* containing the tar.gz file that provides metadata and the essential source files needed for installing by pip. It also created the Media_Manipulation_Detector.egg-info folder, it contains files and metadata needed to be moved to a specific location on a target system.

Before **uploading the package to the Test PyPi repository**, *twine* needs to be installed. This utility assists anyone that intends to upload a package to the PyPI repository (or other repositories). Once installed, there is no need need to repeat this step. The command to install *twine* is:

```
pip install twine
```

The command to upload the Python package to the Test PyPi repository is:

```
twine upload --repository-url https://test.pypi.org/legacy/ dist/*
```

Finally, the only step missing is **upload your package to the PyPI repository**, was executing the last command that uploads the package to PyPi, where it is shared with a worldwide community of Python developers. This command needed to be executed on the same directory where *setup.py* is:

```
twine upload dist/*
```

As a result of publishing this python package everyone can now use it to further develop the field of media manipulation detection. This is possible by simply executing the command found in Figure 3.4.

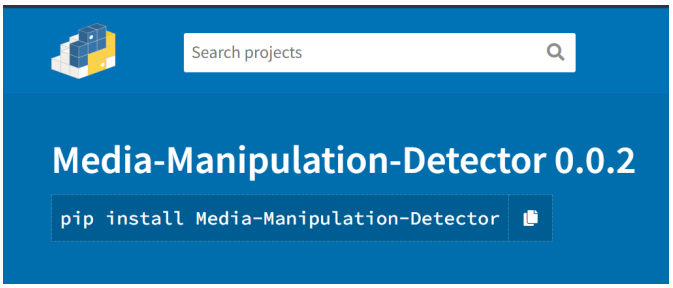

Figure 3.4: PyPi Media Manipulation Detector package page

3.2.4.2 Usage of the python package

The usage of the Media Manipulation python package is very simple. After installation (as explained above) it's possible to use the code below to create a training file or create a model.

```
from MediaManipulationDetector import ManipulationDetector

ManipulationDetector \
    .create_file(mode="train",
                 src_dir="/root/dataset/original/",
                 pre_processing="fft",
                 max_files=1000,
                 number_features=50,
                 pca_reduction=0,
                 out_file="output_fft_50f_1000max_0pca")
```

Listing 3.6: Example usage of the Media Manipulation Detector Package to create a training file

The first step is always the creation of a file that will be used for training or testing purposes, with that intention the method *create_file* needs to receive the following parameters:

- mode - there are two possibilities: "train" and "test". The elected option defines the type of file expected to be created by
- src_dir - the source directory is the location of the media to be used to create the file, the folder should be divided into real and fake subfolders. The subfolder will be used to label.
- pre_processing - the types of pre-processing available are dft and fft, as discussed in previous chapters.
- max_files - the number of files to load per type (fake and real).
- number_features - number of features to be extracted by the Fourier transformation, the default value is 50;

- pca_reduction - if pca is used (when the value is different than 0), then the reduction factor needs to be used. The value is between 0 and 1 and it is multiplied by the number of features to give out the final number of features.
- out_file - the output file that will be created;

```
from MediaManipulationDetector import ManipulationDetector

ManipulationDetector \
    .create_model("svm", 10, "output_fft_50f_1000max_0pca.pkl")
```

Listing 3.7: Example usage of the Media Manipulation Detector Package to create a SVM model

The example presented in Listing 3.7 it's possible to see that creating an SVM model is as simple as two lines of code. There are only 3/4 parameters needed for the method. The first is the algorithm, whose options are currently only "svm" and "cnn". The second is the k-fold/run mode option, the options accept are these below:

- -1: classifies each entry in the *testing file* (4th and optional parameter);
- 0: splits the dataset into two parts (67% for training and 33% for testing);
- 5: splits the dataset to be used in a 5-fold cross validation;
- 10: splits the dataset to be used in a 10-fold cross validation;

The third parameter is the training file that is going to be used to create the model or even benchmark the dataset. When used to test another file, there should be a 4th parameter defined, which is the test file.

Chapter 4

Results and Analysis

This chapter is dedicated to the final results obtained with several experiments. Comparisons between not only the different options available, as shown in the pipeline architecture, but also with other investigators work.

4.1 Final Dataset

On the previous chapter, it was presented the process and difficulties of finding a dataset that matched the requirements of this project. For this reason, a new dataset was created using a mixture of already available media with some photos created for this project.

Table 4.1: Final dataset composition

Dataset	Manipulated	Real	Type of Content
thispersondoesnotexist	10000	0	Photos
COVERAGE dataset	100	100	Photos
Flickr-Faces-the-dataset	0	10000	Photos
Columbia Image Splicing	180	183	Photos
Insta 1M Cities (Subset 100k NYC)	138	0	Photos
Created for this project	2	9	Photos
Facebook's DFDC	42	54	Videos

Table 4.2: Final dataset distribution

Type of Media	Manipulated	Real
Photos	10420	10292
Video	42	54
Total	10462	10346

Table 4.3: Final dataset filters

Filter Type	Manipulated	Real
Original	3427	3427
Black and White	3428	3428
Enhancement	3607	3491

4.2 Test Results

This section is dedicated to the tests performed and their results. From the creation of the files with the extracted features to the several tests performed and the list of the results obtained.

There are going to be considered two datasets for this experiments: the final and the original. The difference being that the original has photos without any kind of filters or obtained by social media (Insta 1M Cities).

For the tests presented below a virtual machine was rented on a cloud provider with the following technical characteristics:

- CPU: 4 vCPUs
- Storage: 80 GB NVMe
- Ram: 16 GB
- Operating System: Debian 11 (64-bit architecture)

Both of these datasets were stripped of videos, since it caused increased load times and memory issues. Something that should be addressed in future developments of this project.

4.2.1 Create Features Files

The first stage is to extract features from the images of the dataset. Those are extracted using DFT and FFT methods, as explained in the previous chapter, and stored in files with the PKL format (pickle).

After obtaining the files of extracted features from the photos using DFT and FFT, with two samples sizes, and registering the running times as registered in Table 4.4. Since some experiments also going to check the impact of PCA, and benchmark pipeline that include it, it was needed to create the corresponding files.

The impact of PCA in processing/running time for a pipeline is minimal as verified by Table 4.5, being insignificant when compared with other stages of the pipeline.

Table 4.4: Running times for feature extraction

Dataset	Number of total items	Features Extraction	Time (in seconds)
original	2000	DFT	506.41
original	2000	FFT	253.40
original	10000	DFT	2589.47
original	10000	FFT	1304.66
final (images-only)	2000	DFT	459.83
final (images-only)	2000	FFT	222.64
final (images-only)	10000	DFT	2329.55
final (images-only)	10000	FFT	1135.63

Table 4.5: Running times with PCA

Dataset	Number of total items	Features Extraction	PCA factor	PCA Running Time (in seconds)
original	2000	DFT	0.5	0.044611
original	2000	FFT	0.5	0.019078
original	10000	DFT	0.5	0.257924
original	10000	FFT	0.5	0.281610
final (images-only)	2000	DFT	0.5	0.285555
final (images-only)	2000	FFT	0.5	0.255066
final (images-only)	10000	DFT	0.5	0.315943
final (images-only)	10000	FFT	0.5	0.270557
original	2000	DFT	0.3	0.050136
original	2000	FFT	0.3	0.015137
original	10000	DFT	0.3	0.390987
original	10000	FFT	0.3	0.230927
final (images-only)	2000	DFT	0.3	0.223123
final (images-only)	2000	FFT	0.3	0.3136561
final (images-only)	10000	DFT	0.3	0.2940781
final (images-only)	10000	FFT	0.3	0.0203497

4.2.2 Evaluation Metrics

Using the 24 files created before, several experiments were performed to obtain benchmarks on the options for the pipeline. Table 4.6 bestows the confusion matrix [35], which inputs the calculations of the evaluation metrics summarized in Table 4.7.

Table 4.6: Confusion Matrix

	Positive	Negative
Positive	TP	FP
Negative	FN	TN

There are four classification metrics that will be used to compare datasets, feature extraction and reduction techniques and algorithms employed. Precision (P) measures the number of photos predicted as manipulated that were, in fact, manipulated. Recall (R) is the percentage of manipulated examples that were predicted from the total number of manipulated examples.

Accuracy (A) measures the rate of correct classifications out of all the examples in the dataset. Finally, the F1-score is weighted average of Precision and Recall. It ranges between [0,1] and measures the robustness and preciseness of the classifier.

Table 4.7: Metrics used to evaluate the dataset

Metric	Equation
Precision	$P = \frac{TP}{(TP + FP)}$ (4.1)
Recall	$R = \frac{TP}{(TP + FN)}$ (4.2)
F1	$F1 = 2 * \frac{P * R}{(P + R)}$ (4.3)
Accuracy	$A = \frac{TP + TN}{TP + TN + FP + FN}$ (4.4)

Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods [7]. The choice of k is usually 5 or 10, but there is no formal rule [44]. For the experiments performed, and the respective results introduced below, it was always considered the k-fold value of 5. Figure 4.1 visually presents how a k-fold 5 experiments is ran.

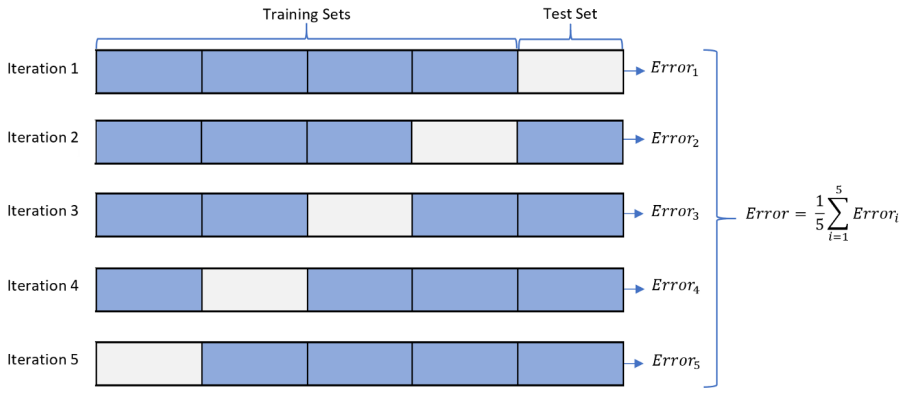

Figure 4.1: Diagram of K-Fold 5 [62]

The number of features to extract from images can be customized, but following the principles and results obtained in the paper "A dataset of photos and videos for digital forensics analysis

using machine learning processing" [22] it was determined that for this experiments the number of features extracted will always be 50.

4.2.3 SVM Results

Table 4.8 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 3.36 seconds, this makes the total pipeline time for this dataset 509.77 seconds (around 9 minutes).

Table 4.8: Results using the original dataset with sample size 2000

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	193	199	6	2	0.99	0.97	0.98	0.98
Split 2	208	182	10	0	1.00	0.95	0.97	0.97
Split 3	196	194	7	3	0.98	0.97	0.97	0.97
Split 4	207	181	10	2	0.99	0.95	0.97	0.97
Split 5	188	203	8	1	1.00	0.96	0.98	0.98
Mean	198	192	8	2	0.98	0.98	0.98	0.97

Table 4.9 shows the results obtained for the final dataset obtained using DFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 6.51 seconds, this makes the total pipeline time for this dataset 466.34 seconds (around 8 minutes).

Table 4.9: Results obtained with 5-fold cross-validation against the final dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	182	189	16	13	0.94	0.93	0.93	0.93
Split 2	191	182	10	17	0.91	0.95	0.93	0.93
Split 3	184	194	7	15	0.93	0.97	0.95	0.94
Split 4	194	180	11	15	0.94	0.94	0.93	0.94
Split 5	175	195	16	14	0.93	0.92	0.93	0.93
Mean	185	188	12	15	0.93	0.93	0.93	0.93

Table 4.10 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 17.07 seconds, this makes the total pipeline time for this dataset 1321.73 seconds (around 22 minutes).

Table 4.11 shows the results obtained for the final dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 107.44 seconds, this makes the total pipeline time for this dataset 2436.99 seconds (around 41 minutes).

Table 4.12 shows the results obtained for the original dataset obtained using FFT and SVM

Table 4.10: Results obtained with 5-fold cross-validation against the final dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1015	960	19	6	0.99	0.98	0.99	0.99
Split 2	980	993	14	13	0.99	0.99	0.99	0.99
Split 3	984	995	14	7	0.99	0.99	0.99	0.99
Split 4	993	983	17	7	0.99	0.98	0.99	0.99
Split 5	991	979	26	4	1.00	0.97	0.98	0.98
Mean	993	982	18	7	0.99	0.98	0.99	0.99

Table 4.11: Results obtained with 5-fold cross-validation against the final dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1002	955	24	19	0.98	0.98	0.98	0.98
Split 2	968	979	28	25	0.98	0.97	0.97	0.97
Split 3	967	969	40	24	0.98	0.96	0.97	0.97
Split 4	988	967	33	12	0.99	0.97	0.98	0.98
Split 5	969	964	41	26	0.97	0.96	0.97	0.97
Mean	979	967	33	21	0.97	0.97	0.97	0.97

combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 0.72 seconds, this makes the total pipeline time for this dataset 254.12 seconds (around 4 minutes).

Table 4.12: Results obtained with 5-fold cross-validation against the original dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	195	205	0	0	1.00	1.00	1.00	1.00
Split 2	208	191	1	0	1.00	0.99	1.00	1.00
Split 3	199	201	0	0	1.00	1.00	1.00	1.00
Split 4	208	189	2	1	0.99	0.99	0.99	0.99
Split 5	189	209	2	0	1.00	0.99	1.00	0.99
Mean	200	199	1	0	1.00	0.99	1.00	1.00

Table 4.13 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 0.34 seconds, this makes the total pipeline time for this dataset 222.098 seconds (around 4 minutes).

Table 4.14 shows the results obtained for the original dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 2.78 seconds, this makes the total pipeline time for this dataset 1307.44 seconds (around 21.79 minutes).

Table 4.15 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM

Table 4.13: Results obtained with 5-fold cross-validation against the final dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	194	205	0	1	1.00	1.00	1.00	1.00
Split 2	204	189	3	4	0.98	0.98	0.98	0.98
Split 3	196	200	1	3	0.99	1.00	0.99	0.99
Split 4	206	191	0	3	0.98	1.00	0.99	0.99
Split 5	188	208	3	1	1.00	0.99	0.99	0.99
Mean	198	199	1	2	0.99	0.99	0.99	0.99

Table 4.14: Results obtained with 5-fold cross-validation against the original dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1019	979	0	2	1.00	1.00	1.00	1.00
Split 2	985	1006	1	8	0.99	1.00	1.00	1.00
Split 3	988	1006	3	3	1.00	1.00	1.00	1.00
Split 4	998	999	1	2	1.00	1.00	1.00	1.00
Split 5	992	1003	2	3	1.00	1.00	1.00	1.00
Mean	996	999	1	4	1.00	1.00	1.00	1.00

model was 4.53 seconds, this makes the total pipeline time for this dataset 1140.16 seconds (around 19 minutes).

Table 4.15: Results obtained with 5-fold cross-validation against the final dataset

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1009	976	3	12	0.99	1.00	0.99	0.99
Split 2	981	1004	3	12	0.99	1.00	0.99	0.99
Split 3	981	1003	10	6	0.99	0.99	0.99	0.99
Split 4	987	994	6	13	0.99	0.99	0.99	0.99
Split 5	984	1002	3	11	0.99	1.00	0.99	0.99
Mean	988	996	5	11	0.99	1.00	0.99	0.99

This concludes the collection of results from experiments without PCA, it's possible to observe the fact that FFT reduces the running time for the feature extraction phase by 49.39% on average, it also reduces the SVM running time by 11.79% on average. Comparing the results we verify that not only DFT feature extraction is slower, it also has worst results than FFT.

4.2.4 PCA Results

Table 4.16 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 8.47 seconds, this makes the total pipeline time for this dataset 514.93 seconds (around 9 minutes).

Table 4.17 shows the results obtained for the original dataset obtained using FFT and SVM

Table 4.16: SVM results obtained for the original dataset with features extracted using DFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	205	174	21	0	0.91	1.00	0.95	0.95
Split 2	189	191	1	19	0.91	0.99	0.95	0.95
Split 3	176	201	0	23	0.90	1.00	0.95	0.94
Split 4	180	191	0	29	0.87	1.00	0.93	0.93
Split 5	167	211	0	22	0.91	1.00	0.95	0.94
Mean	183	194	4	19	0.90	1.00	0.94	0.94

combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 6.56 seconds, this makes the total pipeline time for this dataset 259.98 seconds (around 4 minutes)

Table 4.17: SVM results obtained for the original dataset with features extracted using FFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	190	205	0	5	0.98	1.00	0.99	0.99
Split 2	202	192	0	6	0.97	1.00	0.98	0.98
Split 3	188	201	0	11	0.95	1.00	0.97	0.97
Split 4	203	190	1	6	0.97	0.99	0.98	0.98
Split 5	183	210	1	6	0.97	1.00	0.98	0.98
Mean	193	200	0	7	0.97	1.00	0.98	0.98

Table 4.18 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 54.23 seconds, this makes the total pipeline time for this dataset 2642.96 seconds (around 44 minutes).

Table 4.18: SVM results obtained for the original dataset with features extracted using DFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	970	978	1	51	0.95	1.00	0.97	0.97
Split 2	924	1007	0	69	0.94	1.00	0.97	0.97
Split 3	926	1007	2	65	0.94	1.00	0.97	0.97
Split 4	939	998	2	61	0.94	1.00	0.97	0.97
Split 5	946	1004	1	49	0.95	1.00	0.98	0.97
Mean	941	999	1	59	0.94	1.00	0.97	0.97

Table 4.19 shows the results obtained for the original dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 17.78 seconds, this makes the total pipeline time for this dataset 1322.72 seconds (around 22 minutes)

Table 4.19: SVM results obtained for the original dataset with features extracted using FFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1010	979	0	11	0.99	1.00	0.99	0.99
Split 2	962	1006	1	31	0.97	1.00	0.98	0.98
Split 3	974	1009	0	17	0.98	1.00	0.99	0.99
Split 4	983	999	1	17	0.98	1.00	0.99	0.99
Split 5	982	1004	1	13	0.99	1.00	0.99	0.99
Mean	982	999	1	18	0.98	1.00	0.99	0.99

Table 4.20 shows the results obtained for the final dataset obtained using DFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 2.56 seconds, this makes the total pipeline time for this dataset 462.68 (around 8 minutes).

Table 4.20: SVM results obtained for the final dataset with features extracted using DFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	172	199	6	23	0.90	0.97	0.93	0.93
Split 2	184	185	7	24	0.89	0.96	0.92	0.92
Split 3	158	196	5	41	0.83	0.98	0.89	0.89
Split 4	186	185	6	23	0.89	0.97	0.93	0.93
Split 5	163	207	4	26	0.89	0.98	0.93	0.93
Mean	173	194	6	27	0.88	0.97	0.92	0.92

Table 4.21 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 1.33 seconds, this makes the total pipeline time for this dataset 224.23 seconds (around 4 minutes).

Table 4.21: SVM results obtained for the final dataset with features extracted using FFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	188	204	1	7	0.97	0.96	0.98	0.98
Split 2	200	191	1	8	0.96	0.99	0.98	0.98
Split 3	186	201	0	13	0.94	1.00	0.97	0.97
Split 4	199	190	1	10	0.95	0.99	0.97	0.97
Split 5	182	209	2	7	0.97	0.99	0.98	0.98
Mean	191	199	1	9	0.96	0.99	0.98	0.98

Table 4.22 shows the results obtained for the final dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 52.75 seconds, this makes the total pipeline time for this dataset 2382.62 seconds

(around 40 minutes).

Table 4.22: SVM results obtained for the final dataset with features extracted using DFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	949	966	13	72	0.93	0.99	0.96	0.96
Split 2	934	995	12	59	0.94	0.99	0.97	0.96
Split 3	919	988	21	72	0.93	0.98	0.96	0.95
Split 4	919	987	13	81	0.92	0.99	0.95	0.95
Split 5	917	989	16	78	0.93	0.98	0.95	0.95
Mean	928	985	15	72	0.93	0.99	0.96	0.95

Table 4.23 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 14.75 seconds, this makes the total pipeline time for this dataset 1150.65 seconds (around 19 minutes).

Table 4.23: SVM results obtained for the final dataset with features extracted using FFT and PCA factor 0.5

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	993	974	5	28	0.97	0.99	0.98	0.98
Split 2	972	1000	7	21	0.98	0.99	0.99	0.99
Split 3	973	1004	5	18	0.98	1.00	0.99	0.99
Split 4	969	995	5	31	0.97	0.99	0.98	0.98
Split 5	971	1002	3	24	0.98	1.00	0.99	0.99
Mean	976	995	5	24	0.98	0.99	0.99	0.99

Table 4.24 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 8.26 seconds, this makes the total pipeline time for this dataset 514.72 seconds (around 9 minutes).

Table 4.24: SVM results obtained for the original dataset with features extracted using DFT and PCA factor 0.3

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	175	205	0	20	0.91	1.00	0.95	0.95
Split 2	189	191	1	19	0.91	0.99	0.95	0.95
Split 3	176	201	0	23	0.90	1.00	0.95	0.94
Split 4	184	191	0	25	0.88	1.00	0.94	0.94
Split 5	168	211	0	21	0.91	1.00	0.95	0.95
Mean	178	200	0	22	0.90	1.00	0.95	0.95

Table 4.25 shows the results obtained for the original dataset obtained using FFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM

model was 6.31 seconds, this makes the total pipeline time for this dataset 259.73 seconds (around 4 minutes).

Table 4.25: SVM results obtained for the original dataset with features extracted using FFT and PCA factor 0.3

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	190	205	0	5	0.98	1.00	0.99	0.99
Split 2	202	192	0	6	0.97	1.00	0.98	0.98
Split 3	189	201	0	10	0.95	1.00	0.98	0.97
Split 4	203	190	1	6	0.97	0.99	0.98	0.98
Split 5	183	210	1	6	0.97	1.00	0.98	0.98
Mean	193	200	0	7	0.97	1.00	0.98	0.98

Table 4.26 shows the results obtained for the original dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 49.51 seconds, this makes the total pipeline time for this dataset 2639.37 seconds (around 44 minutes).

Table 4.26: SVM results obtained for the original dataset with features extracted using DFT and PCA factor 0.3

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	974	978	1	47	0.95	1.00	0.98	0.98
Split 2	926	1004	3	67	0.94	1.00	0.97	0.96
Split 3	929	1007	2	62	0.94	1.00	0.97	0.97
Split 4	944	998	2	56	0.95	1.00	0.97	0.97
Split 5	948	1004	1	47	0.96	1.00	0.98	0.98
Mean	944	998	2	56	0.95	1.00	0.97	0.97

Table 4.27 shows the results obtained for the original dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 16.00 seconds, this makes the total pipeline time for this dataset 1320.89 seconds (around 22 minutes).

Table 4.27: SVM results obtained for the original dataset with features extracted using FFT and PCA factor 0.3

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	1010	979	0	11	0.99	1.00	0.99	0.99
Split 2	964	1006	1	29	0.97	1.00	0.99	0.98
Split 3	975	1008	1	16	0.98	1.00	0.99	0.99
Split 4	983	999	1	17	0.98	1.00	0.99	0.99
Split 5	982	1004	1	13	0.99	1.00	0.99	0.99
Mean	983	999	1	17	0.98	1.00	0.99	0.99

Table 4.28 shows the results obtained for the final dataset obtained using DFT and SVM

combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 2.36 seconds, this makes the total pipeline time for this dataset 462.41 seconds (around 8 minutes).

Table 4.28: SVM results obtained for the final dataset with features extracted using DFT and PCA factor 0.3

	TP	**TN**	**FP**	**FN**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Split 1	174	199	6	21	0.90	0.97	0.94	0.93
Split 2	184	185	7	24	0.89	0.96	0.92	0.92
Split 3	159	197	4	40	0.83	0.98	0.90	0.89
Split 4	187	185	6	22	0.89	0.97	0.93	0.93
Split 5	164	207	4	25	0.89	0.98	0.93	0.93
Mean	174	195	5	26	0.88	0.97	0.92	0.92

Table 4.29 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 2000, with a 5-fold cross-validation. The running time for the SVM model was 1.18 seconds, this makes the total pipeline time for this dataset 224.13 seconds (around 4 minutes).

Table 4.29: SVM results obtained for the final dataset with features extracted using FFT and PCA factor 0.3

	TP	**TN**	**FP**	**FN**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Split 1	188	204	1	7	0.97	1.00	0.98	0.98
Split 2	200	191	1	8	0.96	0.99	0.98	0.98
Split 3	186	201	0	13	0.94	1.00	0.97	0.97
Split 4	199	190	1	10	0.95	0.99	0.97	0.97
Split 5	182	209	2	7	0.97	1.00	0.98	0.98
Mean	191	199	1	9	0.96	1.00	0.98	0.98

Table 4.30 shows the results obtained for the final dataset obtained using DFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 46.26 seconds, this makes the total pipeline time for this dataset 2376.1 seconds (around 40 minutes).

Table 4.30: SVM results obtained for the final dataset with features extracted using DFT and PCA factor 0.3

	TP	**TN**	**FP**	**FN**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Split 1	951	965	14	70	0.93	0.99	0.96	0.96
Split 2	935	995	12	58	0.94	0.99	0.97	0.96
Split 3	917	989	20	74	0.93	0.98	0.95	0.95
Split 4	921	986	14	79	0.93	0.99	0.95	0.95
Split 5	920	991	14	75	0.93	0.99	0.96	0.96
Mean	929	985	15	71	0.90	1.00	0.95	0.95

Table 4.31 shows the results obtained for the final dataset obtained using FFT and SVM combination, sample size 10000, with a 5-fold cross-validation. The running time for the SVM model was 13.05 seconds, this makes the total pipeline time for this dataset 1148.7 seconds (around 19 minutes).

Table 4.31: SVM results obtained for the final dataset with features extracted using FFT and PCA factor 0.3

	TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy
Split 1	994	974	5	27	0.97	0.99	0.98	0.98
Split 2	974	1000	7	19	0.98	0.99	0.99	0.99
Split 3	972	1004	5	19	0.98	1.00	0.99	0.99
Split 4	970	995	5	30	0.97	0.99	0.98	0.98
Split 5	971	1001	4	24	0.98	1.00	0.99	0.99
Mean	976	995	5	24	0.97	1.00	0.98	0.98

4.3 CNN Results

Since all the experiments until this point rely on SVM models, an additional experiment was performed to collect results and compare with the results obtained in the previous section. However, since the machine previously used to collect results didn't have a GPU, an hardware component needed to run Deep Learning algorithms, another computer was used. For the two experiments below an Apple Macbook Pro (2020) was selected, with the following characteristics:

- M1 processor (ARM64 architecure)
- 16 GB RAM
- 512 GB SSD

Although this computer doesn't have a dedicated graphics card either, M1 processors have an integrated GPU and a Neural Engine [4] so it's possible to run ML and Deep Learning tasks.

Regarding CNN configuration, there are two important parameters: dataset distribution and the number of epochs. Although, there isn't a rule, the distribution of 80% for training and 20% for testing is the most commonly used. As for the number of epochs, it's the number of complete passes through the training dataset, for both experiments the epoch value was 10.

Table 4.32: CNN results for the complete original dataset

TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy	Running Time
1992	2062	2	57	0.97	1.00	0.99	0.99	2 hrs 5 min 11 sec

Table 4.33: CNN results for the complete final dataset (images-only)

TP	TN	FP	FN	Precision	Recall	F1-Score	Accuracy	Running Time
2032	2072	33	6	1.00	0.98	0.99	0.99	2 hrs 18 min 9 sec

Table 4.32 and Table 4.33 present the results for the two deep learning experiments performed. Both models used the complete dataset available to them, making the size of the original dataset 20563 images and of the final dataset (images-only) 20712 images.

4.4 Summary

In this chapter a description of the main evaluation metrics was made, namely precision, recall, F1-score and accuracy.

The composition of the dataset created in order to test and train the DFT+SVM, FFT+SVM methods implemented (and respective variants with PCA) was described. All results obtained with a 5-fold cross validation were also presented in this chapter. Finally, a deep learning experiment (using a CNN-based method) was conducted to benchmark both models in terms of classification performance and processing time. In chapter 5 all the conclusions reached with this dissertation will be addressed.

Chapter 5

Conclusions

In this chapter the development and results accomplished throughout this project and dissertation are going to be presented. Starting with a brief overview of the research and development perfomed, commenting on the deliverables and the goals achieved. Following, a description of the results obtained is done, commenting on the balance between the processing/running time and the evaluation performance. Finally, some ideas for future work and improvements to be performed.

5.1 Research and Development

This dissertation described the development of an application to detect tampered multimedia content. An entire Python package was created to benchmark the proposed solutions for the pipeline. A Support Vector Machines (SVM)-based method was implemented to process the previously extracted features obtained by a Discrete Fourier Transform (DFT) or Fast Fourier Transform (FFT) calculation in each multimedia file being processed.

A dataset was created, consisting not only on already available photos/videos but also including filters, enhancing the classification for this content, since it can be mistakenly classified as manipulated, or fake, when it was not changed for malicious purposes.

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 and FFT techniques to extract features from photos, and SVM to classify files.

5.2 Results

The results were presented into three distinct aspects: the classification performance obtained with a five-fold cross-validation for photos between datasets with features extracted by DFT and FFT; the impact of using PCA for feature reduction; the benchmark between SVM and

CNN-based methods using the dataset created and another dataset.

Concluding from the results obtained from the experiments conducted, FFT reduces the running time for the feature extraction phase by 49.39% on average, it also reduces the SVM running time by 11.79% on average. Comparing the results we verify that not only DFT feature extraction is slower, it also produced worst results than FFT. A combination of SVM+FFT pipeline was, on average, 48.65% faster than SVM+DFT combination.

Experiments using PCA revealed that their impact is overall negative. Their running time as a independent phase is not very consuming but the scores decreased when compared with other for the same dataset, same features extraction method and algorithm, whose only difference was not using PCA to reduce their dimensionality. Concluding, that for the proposed pipeline PCA has no advantages in terms of reduction of running time or overall efficiency.

Comparing SVM and CCN results is difficult since the same machine was not used. However it's still possible to understand that the most advantageous algorithm to use in this context is SVM. The scores obtained, using the four comparison metrics, are not significantly better (or even better in some cases) for CNNs when compared with SVM. Adding the fact that SVM need significant less computing power and less running power it's possible to determine that a combination of FFT+SVM is a better choice.

5.3 Future Work

Although the results were very successful for photos, it wasn't possible to obtain with videos datasets. A clear path forward would be to improve on that field, re-work the python package, take advantage of other computer architectures, or even use machines with higher computing power.

The python package is already a very good deliverable, but to further develop the field, there is a need to continue the development by making applications that are usable by less technology literate users (the most vulnerable to miss-information).

Bibliography

[1] Wadha Abdullah Al-Khater, Somaya Al-Maadeed, Abdulghani Ali Ahmed, Ali Safaa Sadiq, and Muhammad Khurram Khan. Comprehensive review of cybercrime detection techniques. *IEEE Access*, 8:137293–137311, 2020. doi:10.1109/ACCESS.2020.3011259.

[2] Potentia Analytics. What is machine learning: Definition, types, applications and examples.

[3] Mário Antunes and Baltazar Rodrigues. *Introdução à Cibersegurança A Internet, os Aspetos Legais e a Análise Digital Forense*. FCA, 2018. ISBN: 9789727228614.

[4] Apple. Apple unleashes m1, nov 2020.

[5] Sevinc Bayram, Husrev Taha Sencar, and Nasir Memon. An efficient and robust method for detecting copy-move forgery. pages 1053–1056, Taipei, Taiwan, 2009. IEEE. ISBN: 978-1-4244-2354-5. doi:10.1109/ICASSP.2009.4959768.

[6] Omer Benjakob. Israeli phone-hacking firm claims it can now break into encrypted signal app. Haaretz, December 2020.

[7] Jason Brownlee. A gentle introduction to k-fold cross-validation, may 2018.

[8] Jason Brownlee. A gentle introduction to imbalanced classification, December 2019.

[9] C3.ai. Glossary: Model training.

[10] R. Venkata Ramana Chary, D. Rajya Lakshmi, and K. V. N. Sunitha. Feature extraction methods for color image similarity. April 2012.

[11] Sampriti Chatterjee. What is feature extraction? feature extraction in image processing, October 2021.

[12] Samantha Cole. Ai-assisted fake porn is here and we're all fucked. Online, December 2017.

[13] B. B. Le Cun, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition witha back-propagation network. *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 396–404, 1989.

[14] Diário da República Eletrónico. Lei do cibercrime - lei n.º 109/2009, 2009.

[15] Diário da República Eletrónico. Lei n.º 79/2021, de 24 de novembro, 2021.

[16] darvideo. What is frame?

[17] Gabinete do Secretário-Geral. Relatório anual de segurança interna 2020, 2020.

[18] Philipe Donn. Brown hummingbird selective focus photography.

[19] Denise Dragos and Suzanna Schmeelk. Locating the perpetrator: Industry perspectives of cellebrite education and roles of gis data in cybersecurity and digital forensics, 2021. ISSN: 2367-3370. doi:10.1007/978-3-030-80129-$8_6$8.

[20] Parlamento Europeu e do Conselho. Diretiva (ue) 2019/713, April 2019.

[21] Better Explained. An interactive guide to the fourier transform.

[22] Sara Ferreira, Mário Antunes, and Manuel E Correia. A dataset of photos and videos for digital forensics analysis using machine learning processing. *Data*, 6(8):87, 2021.

[23] Sara Cardoso Ferreira. A machine learning based digital forensics application to detect tampered multimedia files. mathesis, Faculty of Sciences of University of Porto, July 2021.

[24] Sara Cardoso Ferreira. A machine learning based digital forensics application to detect tampered multimedia files. 2021.

[25] The Python Software Foundation. Glossary, May 2022.

[26] Steven J. Frank. This ai can spot an art forgery.

[27] K. Fukushima and S. Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual patternrecognition. *Competition and cooperation in neural nets*, 1982.

[28] GeeksforGeeks. Matlab | rgb image representation, June 2018.

[29] GeeksforGeeks. Ml | classification vs regression, September 2021.

[30] Siong Thye Goh. Maximize the margin formula in support vector machines algorithm, December 2018.

[31] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. December 2015.

[32] Raúl Gómez. The instacities1m dataset, August 2018.

[33] Mohammad Farukh Hashmi, Aaditya Hambarde, Vijay Anand, and Avinash Keskar. Passive detection of copy-move forgery using wavelet transforms and sift features. *Journal of Information Assurance and Security(JIAS) ISSN 1554-1010*, 9:197–204, 08 2014.

[34] Robert Hecht-Nielsen. Theory of the backpropagation neural network. 1:445, 1988. ISSN: 0893-6080. doi:10.1016/0893-6080(88)90469-8.

[35] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

[36] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 1968.

[37] IBM. Deep learning.

[38] IPTC. What is photo metadata?

[39] Hamid Jalab, Thamarai Subramaniam, Rabha Ibrahim, Hasan Kahtan, and Nurul Noor. New texture descriptor based on modified fractional entropy for digital image splicing forgery detection. 21:371. ISSN: 1099-4300. doi:10.3390/e21040371.

[40] HiDEF New Jersey. An introduction to video 101.

[41] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. December 2019.

[42] Karen Kent, Suzanne Chevalier, Tim Grance, and Hung Dang. Guide to integrating forensictechniques into incident response, August 2006. doi:10.6028/nist.sp.800-86.

[43] Dong-Hyun Kim and Hae-Yeoun Lee. Image manipulation detection using convolutional neural network. *International Journal of Applied Engineering Research*, 12(21):11640–11646, 2017.

[44] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.

[45] Nitish Kumar. Introduction to support vector machines (svms), March 2021.

[46] Myung-Joon Kwon, Seung-Hun Nam, In-Jae Yu, Heung-Kyu Lee, and Changick Kim. Learning jpeg compression artifacts for image manipulation detection and localization. August 2021.

[47] Scikit Learn. 1.4.support vector machines, 2022.

[48] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998. ISSN: 1558-2256. doi:10.1109/5.726791.

[49] Manav Mandal. Introduction to convolutional neural networks (cnn), May 2021.

[50] Katerina Eva Matsa Mason Walker. News consumption across social media in 2021, 2021.

[51] Ben Meisner. What is the future of artificial intelligence in photo editing?, January 2022.

[52] Merriam-Webster. image definition.

[53] Meta, June 2020. [link].

[54] Dorene Asare-Marfo Courtney Kennedy Kirsten Worden Michael Barthel, Amy Mitchell. News consumption across social media in 2021, 2021.

[55] MIT. The mit license.

[56] Shutter Muse. Resampling.

[57] Adobe New Africe. Teenage girl before and after acne treatment on beige background, 2021.

[58] Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Thien Huynh-The, Saeid Nahavandi, Thanh Tam Nguyen, Quoc-Viet Pham, and Cuong M. Nguyen. Deep learning for deepfakes creation and detection: A survey. September 2019.

[59] Danny Paez. 'this person does not exist' creator reveals his site's creepy origin story, February 2019.

[60] Shelly Palmer. The deepest deepfake of all, October 2021.

[61] Mohil Patel, Aaryan Gupta, Sudeep Tanwar, and Mohammad Obaidat. Trans-df: A transfer learning-based end-to-end deepfake detector. 11 2020. doi:10.1109/ICCCA49541.2020.9250803.

[62] Rebecca Patro. Cross-validation: K fold vs monte carlo, January 2021.

[63] Pixelmator Pro. Pixels explained.

[64] Sebastian Raschka. Machine learning faq.

[65] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. 115:211–252, 2015. ISSN: 0920-5691. doi:10.1007/s11263-015-0816-y.

[66] sculpteo. What is an stl file?

[67] Tamsin Selbie and Craig Williams. Deepfake pornography could become an 'epidemic', expert warns, May 2021.

[68] Jolliffe Ian T. and Cadima Jorge. Principal component analysis: a review and recent developments. *The Royal Society Publishing*, April 2016.

[69] techopedia. Resolution, August 2020.

[70] ThisPersonDoesNotExist, 2022. [link].

[71] K. Thyagarajan. *Discrete Fourier Transform*, pages 151–188. 01 2019. ISBN: 978-3-319-76028-5. doi:10.1007/978-3-319-76029-2$_5$.

[72] translated by Laura Gibbs. *Aesop's Fables*. 2002.

[73] US-CERT. Computer forensics.

[74] Jane Wakefield. Deepfake presidents used in russia-ukraine war. BBC Technology, March 2022.

[75] Mika Westerlund. The emergence of deepfake technology: A review. 9:39–52. ISSN: 1927-0321. doi:10.22215/timreview/1282.

[76] Bo Xu, Guangjie Liu, and Yuewei Dai. Detecting image splicing using merged features in chroma space. 2014:1–8, 2014. ISSN: 2356-6140. doi:10.1155/2014/262356.

[77] Wei Zhang, Kazuyoshi Itoh, Jun Tanida, and Yoshiki Ichioka. Parallel distributed processing model with local space-invariant interconnections and its optical architecture. 29:4790, 1990. ISSN: 0003-6935. doi:10.1364/ao.29.004790.