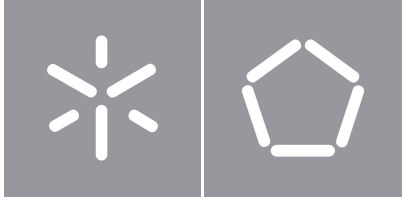


University of Minho
School of Engineering

Tiago Fernandes Ribeiro

Smart Surveillance: violence and vandalism detection using Artificial Intelligence

september 2024



University of Minho
School of Engineering

Tiago Fernandes Ribeiro

Smart Surveillance: violence and vandalism detection using Artificial Intelligence

Master's Dissertation in Informatics Engineering

Dissertation supervised by
Paulo Jorge Freitas de Oliveira Novais
Manuel Fernando Silva Rodrigues

september 2024

Copyright and Terms of Use for Third Party Work

This dissertation reports on academic work that can be used by third parties as long as the internationally accepted standards and good practices are respected concerning copyright and related rights.

This work can thereafter be used under the terms established in the license below.

Readers needing authorization conditions not provided for in the indicated licensing should contact the author through the RepositóriUM of the University of Minho.

License granted to users of this work:



CC BY-NC-SA

https://creativecommons.org/licenses/by-nc-sa/4.0/

Acknowledgements

I would like to express my deep gratitude to everyone who has made a significant contribution to this master's thesis.

First of all, I would like to thank my supervisor, Professor Paulo Novais, for his innovative ideas and constant encouragement. His contribution was essential to the realization of this project.

To my co-supervisor, Professor Manuel Rodrigues, I would like to thank you for your valuable support, tireless guidance, continuous support and shared wisdom throughout this journey. His dedication and patience were fundamental to the development of this work.

I would like to thank Professor Pedro Oliveira for his help and valuable suggestions that have enriched this work. His availability and knowledge were of great importance to me.

I would also like to extend my thanks to all the professors at the University of Minho who accompanied me during this academic journey. All of them were crucial to my training and personal and professional growth.

To the friends and colleagues I've gotten to know over the course of this master's degree, especially Carlos, Francisco, João, Duarte, Izquierdo, Fábio, Pedro, Paulo and António, thank you for your company, mutual support and the unforgettable memories we've shared. You have made this journey lighter, more enjoyable and more memorable.

To my parents and sister, as well as the rest of my family, my deepest thanks for your unconditional love, understanding and unwavering support. You have been my foundation and motivation at all times.

To my girlfriend, Cláudia, my special thanks for your affection, patience and encouragement. Your constant presence has been a pillar of strength and motivation for me.

Finally, I thank God for giving me the opportunity to experience these special years and for having the good fortune to meet all the people I've just mentioned.

To everyone, my heartfelt and sincere thanks.

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, Braga, september 2024

Tiago Fernandes Ribeiro

Abstract

Violence and vandalism in urban environments pose significant risks to public safety, requiring advanced surveillance solutions. This dissertation explores the integration of Artificial Intelligence into video surveillance systems to improve the detection of aggressive and destructive behavior. The study investigates the challenges of urban security, existing crime prevention strategies and the technological landscape of video surveillance, highlighting the ethical considerations involved.

The research examines AI methodologies, particularly Machine Learning, Deep Learning and Computer Vision, and their application in video surveillance. It includes a critical analysis of related research papers, focusing on data processing, the use of contextual data, the avoidance of overfitting and the adjustment of hyperparameters.

The practical part involves preparing datasets, creating models and carrying out tests. The videos were segmented into frames to create a dataset suitable for training AI models. Several models, including Conv3D, ConvLSTM, LRCN, CNN-BiLSTM and a pre-trained MobileNet-BiLSTM model, were developed and evaluated. The CNN-BiLSTM model achieved the highest accuracy among the models created from scratch, with an accuracy rate of 93.750%, while the MobileNet-BiLSTM model achieved an even higher accuracy of 97.768%.

The results highlight the potential of AI applied in the area of public safety through intelligent surveillance systems. Future work should focus on training these models with larger datasets and longer training periods to improve robustness and generalization capacity, ensuring effective detection of violence and vandalism in real scenarios.

Keywords Artificial Intelligence, Violence Detection, Computer Vision, Smart Cities

Resumo

A violência e o vandalismo em ambientes urbanos representam riscos significativos para a segurança pública, necessitando de soluções de vigilância avançadas. Esta dissertação explora a integração da Inteligência Artificial em sistemas de vigilância por vídeo para melhorar a detecção de comportamentos agressivos e destrutivos. O estudo investiga os desafios da segurança urbana, as estratégias de prevenção de crimes existentes e o panorama tecnológico da videovigilância, destacando as considerações éticas envolvidas.

A pesquisa examina metodologias de IA, particularmente Machine Learning, Deep Learning e Visão Computacional, e sua aplicação na videovigilância. Inclui uma análise crítica de artigos de pesquisa relacionados, focando no tratamento de dados, uso de dados contextuais, prevenção de overfitting e ajuste de hiperparâmetros.

A parte prática envolve a preparação de conjuntos de dados, a criação de modelos e a realização de testes. Os vídeos foram segmentados em frames para criar um conjunto de dados adequado ao treino de modelos de IA. Diversos modelos, incluindo Conv3D, ConvLSTM, LRCN, CNN-BiLSTM e um modelo pré-treinado MobileNet-BiLSTM, foram desenvolvidos e avaliados. O modelo CNN-BiLSTM atingiu a maior precisão entre os modelos criados de raiz, com uma taxa de precisão de 93.750%, enquanto o modelo MobileNet-BiLSTM alcançou uma precisão ainda maior de 97.768%.

Os resultados destacam o potencial da IA aplicada na área da segurança pública através de sistemas de vigilância inteligentes. Trabalhos futuros devem focar no treino destes modelos com conjuntos de dados maiores e períodos de treino mais longos para melhorar a robustez e a capacidade de generalização, garantindo a detecção eficaz de violência e vandalismo em cenários reais.

Palavras-chave Inteligência Artificial, Deteção de Violência, Visão por Computador, Cidades Inteligentes

Contents

1 Introduction **1**

1.1 Context and motivation . . . 1

1.1.1 Violence and vandalism . . . 2

1.1.2 Video surveillance . . . 4

1.2 Objectives . . . 6

1.3 Document structure . . . 6

2 State of the art **8**

2.1 Video surveillance . . . 8

2.1.1 Surveillance cameras . . . 9

2.1.2 Data storage systems in video surveillance . . . 11

2.1.3 Cloud computing integration in video surveillance . . . 13

2.1.4 Ethical challenges and considerations in video surveillance . . . 14

2.2 Artificial Intelligence . . . 15

2.2.1 Machine Learning . . . 17

2.2.2 Computer Vision . . . 22

2.2.3 Explainability . . . 24

2.2.4 Video surveillance datasets . . . 25

2.2.5 Machine and Deep Learning models for video surveillance . . . 26

2.2.6 Singular architectures . . . 27

2.2.7 Hybrid architectures . . . 35

2.3 Literature Review . . . 36

2.3.1 Critical analysis of the studies . . . 40

3 Materials and methods **43**

3.1 Data exploration . . . 43

3.2 Data preparation . . . 47
3.3 Technologies . . . 49

4 Experiments **50**
4.1 Training process . . . 50
4.2 Candidate models . . . 52
4.2.1 Conv3D . . . 53
4.2.2 ConvLSTM . . . 54
4.2.3 LRCN . . . 55
4.2.4 CNN-BiLSTM . . . 56
4.2.5 MobileNet-BiLSTM . . . 56
4.3 Evaluation metrics . . . 58
4.4 Pipeline followed . . . 58

5 Results and Discussion **61**
5.1 Model results . . . 61
5.1.1 Conv3D . . . 62
5.1.2 ConvLSTM . . . 64
5.1.3 LRCN . . . 67
5.1.4 CNN-BiLSTM . . . 69
5.1.5 MobileNet-BiLSTM . . . 71
5.2 Comparative analysis . . . 73
5.3 Analysis of Explainability . . . 76

6 Conclusion and Future Work **79**

A Data processing code **97**

B Training process code **99**

C Explainability analysis code **102**

List of Figures

1 Examples of violence and vandalism . . . 3
2 Graph illustrating reported violent and serious crime . . . 3
3 Characteristics of an Smart Video Surveillance System . . . 6

4 Comparison between fixed and **PTZ** cameras . . . 10
5 Example of the use of upscaling algorithms to aid facial recognition . . . 11
6 Example of an architecture of several surveillance systems using cloud computing . . . 14
7 Subfields of **AI** . . . 17
8 Pipeline of the **ML** process . . . 17
9 Example of the architecture of a **ANN** . . . 19
10 Example of the architecture of a **DNN** . . . 20
11 Example of various kinds of visual perception tasks: (a) image classification; (b) object detection; (c) semantic segmentation and (d) instance segmentation . . . 23
12 Example of an architecture of **MLP** . . . 27
13 Example of an architecture **CNN**, adapted to the digit recognition problem . . . 29
14 Example of an architecture **RNN** and how it behaves over time steps . . . 30
15 Architecture of a block of a **LSTM** network . . . 31
16 Architecture of a block of a **GRU** network . . . 32
17 Attention mechanisms used in the Transformer architecture . . . 33
18 Example of a Transformer architecture . . . 34

19 Frames extracted from a violent video in the dataset . . . 45
20 Frames extracted from a non-violent video in the dataset . . . 45
21 Process of choosing the frames to extract . . . 48

22 Visual representation of the training process . . . 51
23 Example of a Conv3D model architecture . . . 54

24 Example of a ConvLSTM model architecture 55
25 Example of a **LRCN** model architecture 55
26 Example of a **CNN-BiLSTM** model architecture 56
27 Example of a MobileNet-BiLSTM model architecture 57
28 Pipeline used throughout the study 59

29 Training and test accuracy of the best Conv3D model 63
30 Training and test loss of the best Conv3D model 64
31 Training and test accuracy of the best ConvLSTM model 66
32 Training and test loss of the best ConvLSTM model 66
33 Training and test accuracy of the best **LRCN** model 68
34 Training and test loss of the best **LRCN** model 68
35 Training and test accuracy of the best **CNN-BiLSTM** model 70
36 Training and test loss of the best **CNN-BiLSTM** model 70
37 Training and test accuracy of the best MobileNet-BiLSTM model 72
38 Training and test loss of the best MobileNet-BiLSTM model 72
39 Comparison between the test accuracy of the four models over the epochs 74
40 Comparison between the test accuracy of the two best models made from scratch with MobileNet-BiLSTM 75
41 Original frames, obtained from a violence video 77
42 Heat maps obtained with Grad-CAM 78

List of Tables

1 Occurrences to highlight in school environments 4

2 Reasons why **CCTV** was or was not useful 12

3 Useful video surveillance datasets 25

4 Comparison between studies 42

5 Violence and Nonviolence Video Statistics 46

6 Results from the Conv3D model 62

7 Results from the ConvLSTM model 65

8 Results from the **LRCN** model 67

9 Results from the **CNN-BiLSTM** model 69

10 Results from the Mobile-BiLSTM model 71

11 Comparison of the results from the different models 73

Acronyms

AI Artificial Intelligence.

AIoT Artificial Intelligence of Things.

ANN Artificial Neural Network.

BiLSTM Bidirectional Long Short-Term Memory.

BoVW Bag of Visual Words.

BVD Big Video Data.

C3D Convolutional 3D.

CCTV Closed-circuit Television.

CNN Convolutional Neural Network.

CNN-BiLSTM Convolutional-Bidirectional Long Short-Term Memory.

CRISP-DM Cross Industry Standard Process for Data Mining.

CV Computer Vision.

DL Deep Learning.

DNN Deep Neural Network.

FPS Frames Per Second.

Grad-CAM Gradient-weighted Class Activation Mapping.

GRU Gated Recurrent Unit.

IoT Internet of Things.

IP Internet Protocol.

LRCN Long-term Recurrent Convolutional Network.

LSTM Long Short-Term Memory.

MFCCs Mel-frequency cepstral coefficients.

ML Machine Learning.

MLP Multilayer Perceptron.

MoBSIFT Motion Boundary Scale-invariant Feature Transform.

PTZ Pan-Tilt-Zoom.

RL Reinforcement Learning.

RNN Recurrent Neural Network.

SIFT Scale-invariant Feature Transform.

SSD Single Shot MultiBox Detector.

SVM Support Vector Machine.

YOLO You Only Look Once.

Chapter 1

Introduction

This work is part of a dissertation in Informatics Engineering at the University of Minho and is entitled "Smart Surveillance: violence and vandalism detection using Artificial Intelligence". The focus of this study is the application of **Artificial Intelligence (AI)** techniques to detect vandalism and violence in monitored environments, with the aim of improving public safety and the efficient management of critical spaces.

This chapter is organized into three distinct parts. The first section, which corresponds to 1.1, consists of a comprehensive contextualization of the central theme of this work, along with an analysis of the reasons that drove the development of this research. The second section, labeled 1.2, outlines the fundamental objectives of this study. Finally, in the third section, identified as 1.3, the overall structure of this document is presented, highlighting the specific topics covered in each of the subsequent chapters.

In addition to the work carried out in this dissertation, it was also possible to expand the research beyond the scope originally proposed, resulting in the production and submission of two scientific articles. The first, entitled "Next-Generation Surveillance: Exploring the Intersection of Artificial Intelligence and Security" [1], has already been published and addresses the intersection between artificial intelligence and security, exploring the implications and applications of these emerging technologies. The second article, entitled "Harnessing Deep Learning for Detection of Violence and Vandalism", is in the process of being published and focuses on the use of deep learning techniques to detect acts of violence and vandalism, demonstrating the practical application and relevance of the concepts discussed throughout this dissertation.

1.1 Context and motivation

Surveillance is an inherent part of modern society, playing a key role in maintaining security and public order [2]. This concept covers a range of practices and technologies aimed at monitoring and observing

people, places and events for various purposes, from security to data analysis. With regard to vandalism and violence, these are worrying challenges that affect communities, property and infrastructure, requiring a way to combat these issues.

1.1.1 Violence and vandalism

Violence is a complex and multifaceted phenomenon that has significant impacts on individuals, communities and societies as a whole [3]. It can take many forms, from physical violence, which involves physical assault and injury, to psychological violence, which can be just as damaging, affecting the mental and emotional health of victims [4]. Understanding and preventing violence is crucial to promoting a safer and more equitable society [5], and research in this field plays a key role in finding effective solutions to reduce violence in all its forms. In this context, this dissertation focuses mainly on the identification of physical violence, but it does not devalue the role that other forms of violence play in the broader picture of security and violence prevention.

Vandalism, on the other hand, refers to deliberate acts of destruction or damage to public or private property. These acts can include graffiti, breaking glass, damaging vehicles and other acts of unauthorized destruction. These acts are often motivated by various factors, such as social discontent, artistic expression or simply an act of rebellion. As a result, vandalism has serious social, economic and legal consequences, and often raises questions about the limits of freedom of expression and the protection of property [6].

In Portugal, the issue of violence and vandalism is a concern that attracts the attention of authorities and researchers. Recent data reveals important information about the prevalence of these types of crimes in the country. In 2022, there was a 14% increase in cases of reported serious violence compared to the previous year.

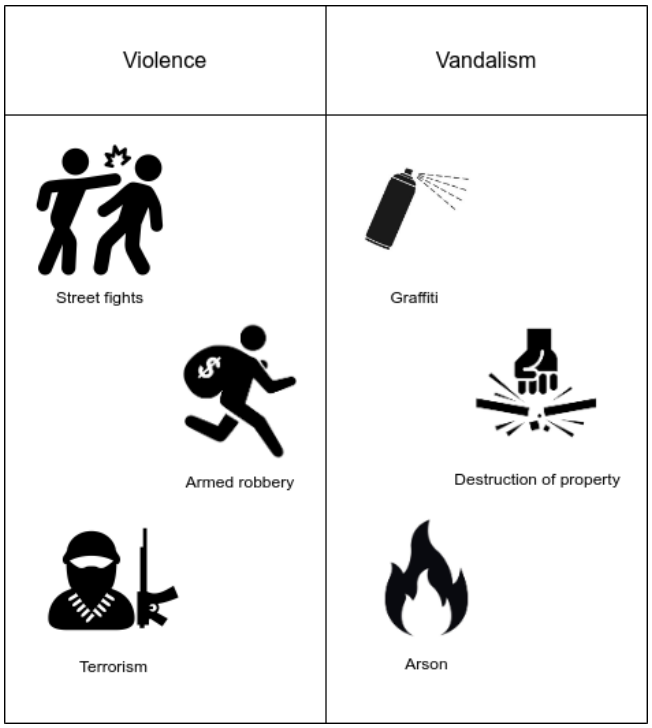


Figure 1: Examples of violence and vandalism

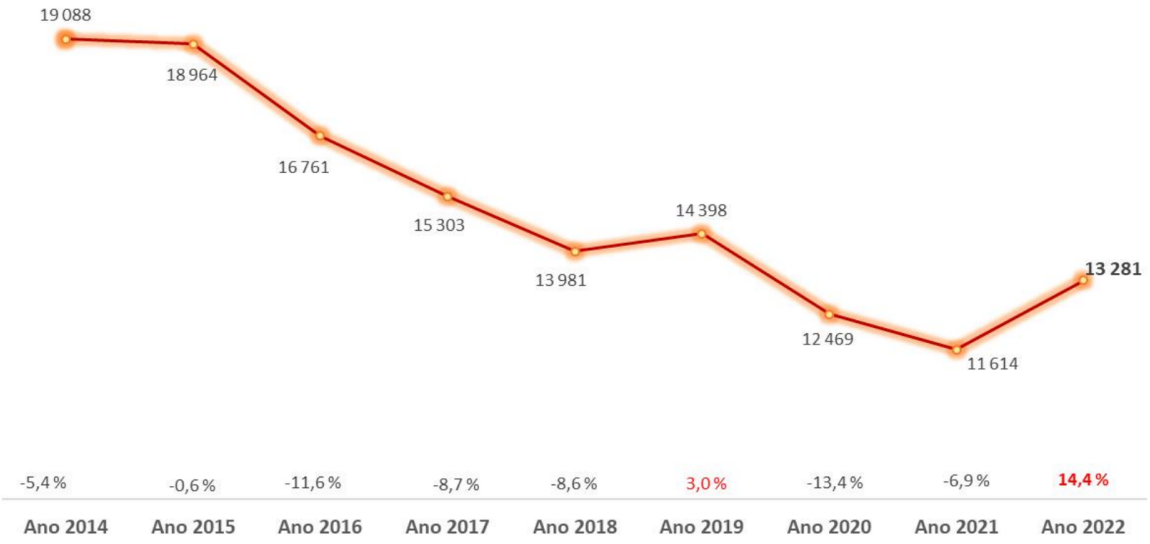


Figure 2: Graph illustrating reported violent and serious crime, taken from [7]

With regard to vandalism crimes, these represent the third largest type of occurrence in school environments, only behind insults/threats and offenses against physical integrity, both of which are related to violence, demonstrating the need for effective prevention and intervention measures against this type of crime [7].

Table 1: Occurrences to highlight in school environments, adapted from [7]

Occurrences	Number
Bomb threats	6
Thefts	711
Robberies	120
Possession/use of weapon	107
Vandalism/damage	298
Injury/threats	1128
Offenses against physical integrity	1860
Possession/consumption of narcotics	127
Sexual offenses	174

1.1.2 Video surveillance

As a result, surveillance has emerged as a vital tool in the attempt to prevent and combat acts of vandalism and violence. In a constantly evolving world, where security and public safety challenges are increasingly complex, surveillance systems play a key role in maintaining order and protecting society. Through the installation of cameras in public spaces, places of transport, commercial buildings and other strategic locations [8], the ability to monitor and deter destructive and violent behavior has become more effective. With the use of cameras for this purpose, it is possible to see a drop in the number of crimes, as well as a reduction in the population's fear of witnessing crimes [9].

Before the advent of **AI** technologies, surveillance relied heavily on human intervention. Operators were needed on the cameras, responsible for continuously monitoring the images captured, detecting suspicious behavior, responding to incidents in real time and coordinating security measures [10]. The ability to discern real-time incidents of both vandalism and violence was intrinsically linked to human attention, judgment and experience [11]. Thus, the need for full-time operators made the use of surveillance cameras expensive, and not all locations could afford to maintain active monitoring teams. When there were no operators on the cameras, they operated for recording purposes only, recording events as they happened, but without the ability to respond immediately. The absence of human operators resulted in a reactive approach to security, where the response to occurrences only happened after the crimes had been committed. Cameras were mainly used to document incidents and provide evidence in retrospect,

being an integral part of police investigations [12].

Smart surveillance

As technology evolves, surveillance systems are also undergoing a significant transformation. This is where the concept of Smart Surveillance comes in, which consists of using video analysis technologies to carry out this surveillance process automatically [13]. With the emergence and advances in **AI** and **Machine Learning (ML)** technologies, automating these processes has become feasible, paving the way for more effective and efficient detection of vandalism and violence.

The transition to automated surveillance systems is motivated by several reasons. Firstly, the automation of these systems offers the ability to process large volumes of video data in real time, a task that would be challenging and exhausting for human operators. With the use of **ML**, **Computer Vision (CV)** and image processing techniques, it is possible to identify patterns and suspicious behavior consistently, without the biases and limitations associated with human observers [14, 15].

In addition, intelligent surveillance allows for a faster response to incidents. When a surveillance system identifies vandalism or violence, it is able to trigger alerts instantly, allowing security authorities to react promptly and help prevent crimes [16]. This is particularly crucial in scenarios where rapid response is essential to prevent further damage.

However, the transition to automated surveillance systems is not without its challenges. Ethical and privacy issues have become crucial concerns as intelligent surveillance systems with **AI** gain more prominence. Clear guidelines and regulations need to be established to ensure that surveillance automation is carried out ethically and respects individual rights [17].

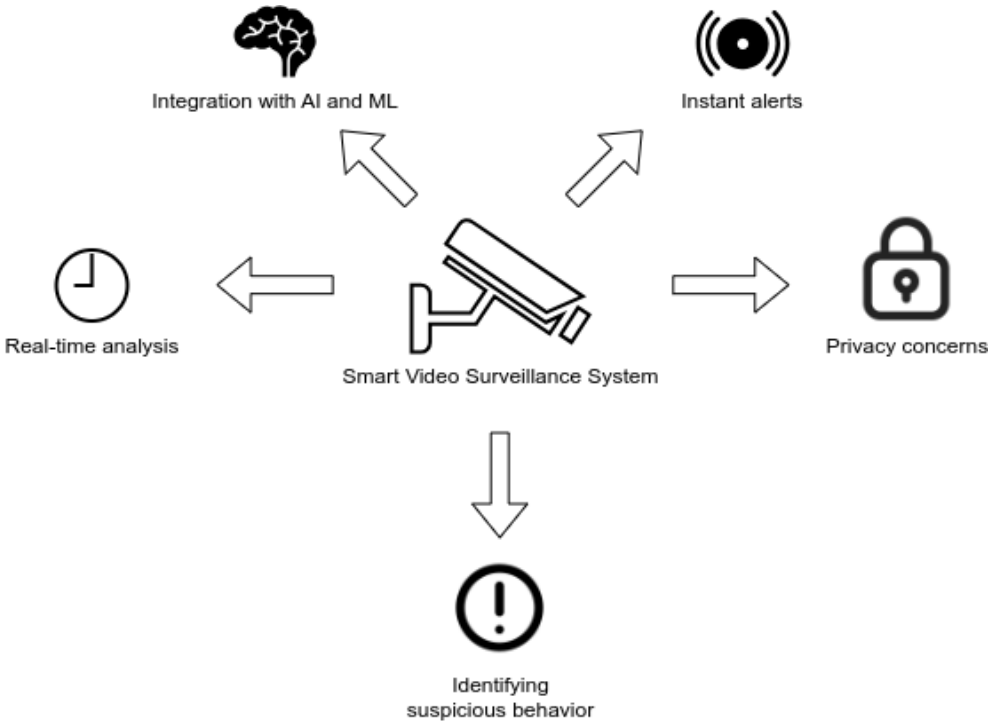


Figure 3: Characteristics of an Smart Video Surveillance System

In addition, the accuracy and reliability of automated systems are areas of continuous research and development. Models must be trained and improved to recognize a variety of scenarios and contexts, as well as minimize false positives and negatives [18].

1.2 Objectives

This dissertation aims to achieve the following objectives:

1. Conduct a review of the existing literature on surveillance systems that use **AI**;
2. Develop various **AI** models for automated detection of vandalism and violence;
3. Carry out relevant tests on the models developed;
4. Analyze and discuss the results obtained.

1.3 Document structure

This document follows a structure organized into several chapters, with the aim of making it easier to understand and read the concepts presented. Starting with the chapter 1, which sets out the framework of

the topic of this work, the delimitation of the problem area and the motivation behind this research. Next, in chapter 2, the fundamental concepts related to the state of the art of this dissertation's topic are explored, which include aspects of smart surveillance, **AI** and its applications. In this chapter, a comprehensive literature review is also conducted to investigate previous work related to the topic, accompanied by a critical analysis.

Subsequently, chapter 3 emerges with the aim of presenting the materials and methods used in this work. The tasks considered in this section involve analyzing the dataset in question, processing it, and getting it ready for the experiments. Following this, chapter 4 appears, where all the experimental scenarios are configured, the training process and design of the **ML** models are presented, as well as the identification of the evaluation metrics and the methodology used to carry out all the experiments. After the configuration and execution of the experiments, the results obtained in chapter 5 are presented, where the analysis and discussion of the results are carried out, as well as the comparison between the best developed models, according to each previously configured experimental approach. Finally, a conclusion is developed in chapter 6, in light of all the work carried out in this dissertation. In addition, relevant measures for application in future work related to the theme of this dissertation are presented.

Chapter 2

State of the art

This chapter provides an in-depth investigation of the fundamental concepts related to the state of the art of this dissertation, focusing in particular on video surveillance and **AI**. Initially, different concepts are covered, starting with the panorama of video surveillance. In the first section, 2.1, the main characteristics of video surveillance are examined, outlining the essential technologies and methods. In the second section, in 2.2, **AI** is introduced and, subsequently, the **ML** and **CV** approaches are explored, in 2.2.1 and 2.2.2. Next, in 2.2.4, datasets used in this area are discussed. Concluding this section, the most relevant **ML** and **Deep Learning (DL)** models applied in the area of video surveillance are presented in 2.2.5, outlining the specific capabilities and applications of these techniques. The chapter culminates with a third section, in 2.3, which comprises a detailed literature review. This review is complemented by a critical analysis, in 2.3.1, of various research articles included in the context of this dissertation, providing a comprehensive overview of existing contributions and the gaps that this research aims to fill.

2.1 Video surveillance

Video surveillance plays a key role in modern security, being an effective tool for preventing, monitoring and investigating criminal activity. At the heart of video surveillance are conventional security camera systems. These systems generally consist of cameras strategically positioned to cover specific areas. Traditional cameras capture and record images, offering a real-time view of activities in the monitored locations [19].

Population growth, together with the terrorist acts of the beginning of the century and the increase in criminal activity, were the main factors driving the large-scale installation of surveillance cameras in urban environments [20]. This phenomenon sparked a great deal of interest in the fields of **CV** and image processing, resulting in millions of cameras covering entire cities and the application of various techniques to ensure the security and efficient management of society [20].

Video surveillance is widely recognized as an extremely useful tool in criminal investigation in general.

When installed in urban areas, with strategically positioned cameras, it becomes a valuable source of evidence for the authorities. The capacity for continuous monitoring allows suspicious events to be identified and provides crucial information for solving crimes. In addition, video surveillance has proven particularly effective in cases of assault, theft and other crimes, contributing to an increase in detection rates. Its importance as a preventive and support tool for the security forces is fundamental to maintaining public safety [21].

2.1.1 Surveillance cameras

As mentioned earlier, cameras are one of the main means of obtaining information in the field of surveillance. These are divided into three categories: analog, digital and **Internet Protocol (IP)** [22]. Analog cameras, due to their age and affordability, have limitations in terms of image quality, recording in traditional media formats and offering only basic functionalities. On the other hand, more advanced digital cameras record in digital format, providing high definition and supporting features such as motion detection and facial recognition, which allows for remote monitoring. On the other hand, **IP** cameras, which are connected to networks, stand out for their high image quality, flexibility in installation via Ethernet or Wi-Fi and advanced features. Considered the most effective, **IP** cameras offer the integration of **AI**-based technologies for real-time detection of suspicious activity, making them a sophisticated and efficient surveillance solution [22].

In terms of how images are captured, there are currently two main types of surveillance cameras: static cameras and **Pan-Tilt-Zoom (PTZ)** cameras. Static cameras are immobile devices positioned to cover a specific area without physical movement. These cameras are ideal for monitoring fixed locations, providing a constant and reliable view. They are usually installed in strategic locations where coverage is essential, but the field of view does not need to be adjusted frequently [23].

On the other hand, **PTZ** mobile cameras offer a dynamic approach to surveillance. These cameras can move horizontally and vertically, as well as adjusting the optical zoom. This design offers great flexibility in covering large areas and enables a rapid response to events in real time [23].

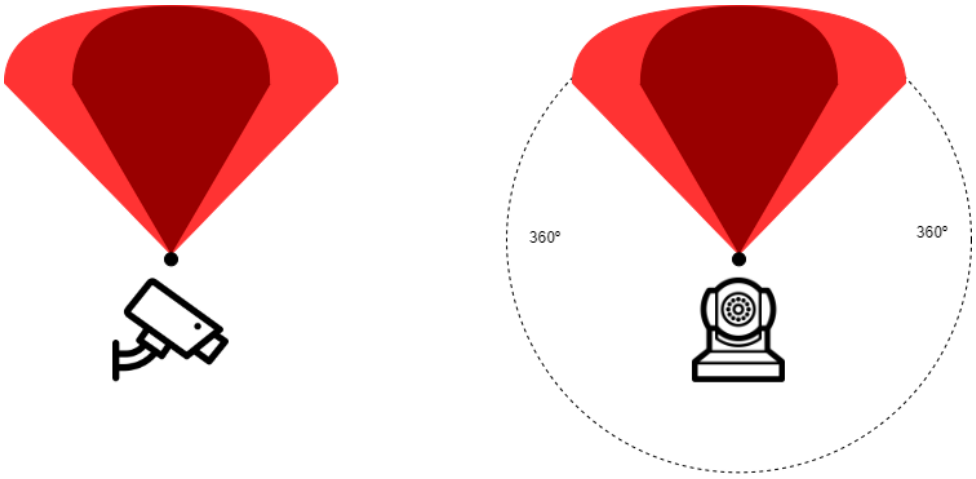


Figure 4: Comparison between fixed and **PTZ** cameras

The two types of cameras have distinct advantages. Static cameras offer stability and are ideal for locations where the focus remains constant. In contrast, mobile cameras are ideal for monitoring large areas, allowing operators to remotely control the direction and zoom of the camera [23].

As well as the choice of camera type being very important, the positioning of the cameras must also be taken into account. The effectiveness of a video surveillance system is intrinsically linked to the strategic distribution of the cameras [24]. Maximizing coverage and minimizing blind spots are key to ensuring comprehensive surveillance of critical areas using security cameras. To do this, it is necessary to adopt specific positioning strategies, taking into account the particularities of each location, such as entrances, exits and high-traffic points, in order to guarantee the effectiveness of video surveillance [24].

Another aspect of high consideration is the camera's resolution, which plays a crucial role in the quality of the image captured. It is essential to select the right resolution to meet the demands of identification and detail required in the context of video surveillance. The higher the image quality, the easier it is to identify illegal behavior, or even facial recognition. To help in situations where the quality of the images is not sufficient, it is possible to use strategies to increase the resolution of the images. This can include the use of technologies such as upscaling, advanced image processing and enhancement techniques to ensure greater clarity and detail in captures [25]. The relationship between camera resolution and storage requirements is also a crucial factor to consider. Accuracy in capturing fine details has a direct impact on the storage capacity required, making it essential to ensure the system's continued effectiveness [26].

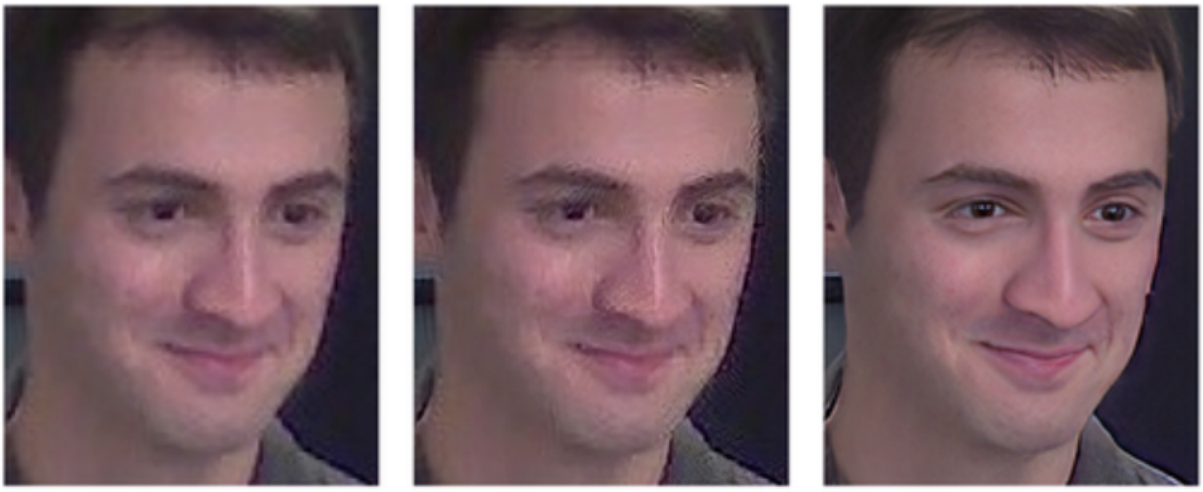


Figure 5: Example of the use of upscaling algorithms to aid facial recognition, taken from [25]

However, contemporary surveillance systems go beyond simply capturing images and into a broader scenario of multidimensional data collection. While cameras remain key elements, these systems have evolved to integrate a variety of sensors and technologies that transcend visual boundaries [27]. In addition to capturing images, surveillance systems now incorporate other types of data, such as acoustic data [28]. This holistic approach allows for a more complete understanding of the monitored environment [27].

Advances in artificial intelligence have also improved the ability to analyze this data, allowing complex patterns and anomalous behavior to be identified. **ML** algorithms are able to correlate data from different sources, offering deeper insights into activities and events in real time [28].

The **Internet of Things (IoT)** plays a crucial role in this scenario, connecting devices and enabling the continuous exchange of data between them. Sensors in buildings, vehicles and personal devices communicate, creating an interconnected network that enhances the effectiveness of surveillance systems [27].

2.1.2 Data storage systems in video surveillance

The security and effectiveness of surveillance systems depend significantly on storage systems, as these are responsible for preserving the integrity and availability of surveillance recordings. This factor becomes even more relevant when considering the authorities' need to access recordings for investigation and law enforcement purposes [21]. The secure and reliable retention of surveillance footage is of paramount importance to ensure that crucial evidence is available when needed. Efficient storage systems ensure that authorities can access historical records even after long periods of time [21].

The practice of replacing old recordings with new ones, also known as overwriting, is widely used in

surveillance systems to optimize the use of storage space. This strategy ensures that the latest recordings are always available, even in long-term surveillance environments. However, it is essential to manage this practice with caution, especially when there is the possibility of important information needing to be accessed at a later date. In certain situations, excessive over-recording can result in the permanent loss of relevant data for future investigations [21]. As indicated by research based on surveillance recordings in the city of London, during the period 2011 to 2015, out of a total of 246,427 reported incidents, in 7,239 (2.9%), footage could not be obtained due to the practice of over-recording, which occurred before the authorities could access the footage [21]

Table 2: Reasons why **Closed-circuit Television (CCTV)** was or was not useful, adapted from [21]

Reason	No. of cases	%
CCTV unavailable	134,819	54,7
Unable to obtain images due to overwriting	7,239	2.9
CCTV available but not useful	39.218	15.9
CCTV useful	72.390	29.4

Ensuring the integrity and accessibility of the information captured is extremely important in the efficient management of video surveillance data. Issues such as the capacity, redundancy and security of storage systems are priorities [26].

Efficient video surveillance data management depends on adequate storage capacity. It is necessary to consider the number of cameras in operation, the resolution of the images and the retention period required when sizing systems. Sizing strategies are implemented to avoid overloads and ensure effective data retention [26].

Redundancy plays a key role in data continuity, even in situations of failure. This is achieved through data replication in distributed systems and the use of redundant storage devices. In addition, fault tolerance is incorporated to guarantee continuous functionality, even in the face of failures in individual components. The integration of cloud storage systems (cloud computing) is also explored, allowing remote access to data and providing flexibility and scalability [29, 30]. This modern approach contributes to operational efficiency and facilitates the centralized management of video surveillance data.

Data security is also a priority and is addressed through practices such as encryption, access control and regular audits. These measures aim to protect data from unauthorized access, guaranteeing the integrity and confidentiality of the information captured by video surveillance systems [30].

2.1.3 Cloud computing integration in video surveillance

The implementation of cloud computing in video surveillance brings with it a significant change in data management and the operational efficiency of video surveillance systems. By enabling remote storage in the cloud, monitoring data becomes instantly accessible anywhere in the world, overcoming geographical limitations [29]. This capability not only facilitates remote management, but also offers flexibility to security professionals, allowing them to monitor events in real time from any location [29].

The dynamic scalability provided by cloud computing is a key element in dealing with fluctuations in storage demand. This flexibility means that systems can continuously and effectively adjust to growing needs, eliminating concerns related to space limitations and ensuring long-term adaptability [29, 30, 31].

Operational efficiency is significantly improved by integrating systems in the cloud. The automation of software updates, simplified maintenance and the ability to quickly incorporate technological innovations are among the main advantages gained by adopting architectures that use cloud computing. These practices not only reduce operating costs in the long term, but also ensure that systems are always up to date and work optimally [31].

However, data security remains a critical priority. When using the cloud for storage, practices such as advanced encryption and robust security protocols are key to protecting the integrity and confidentiality of information [31]. This combination of global accessibility, efficient scalability and rigorous security sets a new standard for the implementation of video surveillance in the digital age [30].

There is currently a growing trend in the area of surveillance towards the adoption of cloud computing technology, although it is important to note that not all systems fully implement this approach [29]. It is essential to recognize that the complete transition to cloud-based architectures is not yet universal. Some surveillance systems remain anchored in traditional client-server architectures, facing scalability and storage challenges. Cloud computing is an efficient and advantageous paradigm, but its widespread implementation is still in progress, reflecting the constant evolution of the security industry to meet increasingly complex and dynamic demands [29].

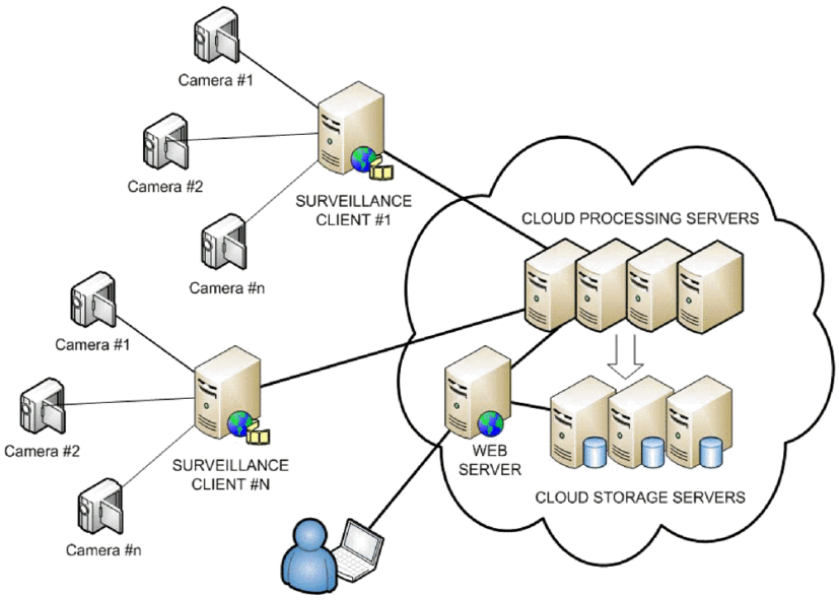


Figure 6: Example of an architecture of several surveillance systems using cloud computing, taken from [29]

2.1.4 Ethical challenges and considerations in video surveillance

The constant evolution of modern video surveillance brings with it a series of ethical challenges that cannot be ignored. One crucial point of concern is privacy, as the increased presence of monitoring systems raises questions about the proper delineation between public safety and individual rights [32]. Another widely raised ethical challenge has to do with data retention. Finding the right temporal balance for storing video surveillance recordings is a crucial consideration [33]. It is of great importance to determine the appropriate duration for keeping information relevant to investigations, while protecting people's privacy [21, 33].

The increasing adoption of facial recognition technologies brings with it profound ethical implications that affect various spheres of society. Privacy concerns are a focal point, since the automatic identification of individuals raises questions about the collection, storage and use of biometric data without explicit consent [34]. This technology also generates discussions about the legal limits needed to regulate its use. The lack of clear regulation could allow questionable practices and raise doubts about the balance between technological innovation and the protection of individual rights [35]. Furthermore, the implications extend to individual freedom, since facial recognition can potentially be used to track and monitor people's daily activities [34]. The debate about the extent to which society is willing to sacrifice privacy in exchange for supposed security is a central point in this context [17]. The potential misuse of biometric information also presents considerable ethical challenges. The mass harvesting of biometric data requires careful thought about who has access to this information, how it is used and what protections exist to prevent its

misuse [35].

Transparency and accountability in the operation of video surveillance systems are seen as ethical pillars. It is important to properly disclose the use of these technologies to the public and the entities responsible need to be accountable for the operation of these systems [34]. All these ethical implications reflect the complexity of a technological advance of this size, requiring in-depth reflection and social deliberation to ensure that the development and implementation of this intelligent video surveillance technology is guided by sound ethical principles and respect for the fundamental rights of individuals [17].

The example of London stands out as a remarkable case in the context of the comprehensive implementation of video surveillance systems. With a robust network of cameras distributed in strategic areas, the city has become an emblematic case of urban surveillance. It is estimated that there are close to one million surveillance cameras in London, resulting in a total number of around 1 camera for every 11 inhabitants of the city [36, 37]. It is estimated that each person is seen around 70 times a day by surveillance cameras [36, 37]. Surveillance cameras in London are installed in public places, transport stations, shopping areas and tourist attractions [36].

These cameras have played a key role in public safety, monitoring daily activities, preventing crimes and assisting in investigations when necessary [8]. In addition, the visible presence of surveillance cameras has a deterrent effect, helping to reduce criminal behavior [8].

However, the extensive use of video surveillance also raises privacy issues, which highlights the need to strike a balance between public safety and the protection of individual rights [38]. The example of London illustrates how modern cities face challenges and at the same time can benefit from implementing comprehensive video surveillance systems [17].

2.2 Artificial Intelligence

Over the last two decades, **AI** has been a crucial factor in the evolution of various sectors, bringing a significant increase in quality, efficiency and satisfaction [39, 40]. Sectors such as service and production systems have experienced remarkable advances thanks to this emerging technology, which allows for the creation of specialized systems capable of performing daily tasks that were previously exclusively carried out by humans [39, 40].

AI represents one of the most remarkable and impactful achievements of computer science and modern technology. The origin of this concept dates back to the early decades of computing, when pioneers such as Alan Turing and John McCarthy explored innovative concepts that would later give rise to this

revolutionary discipline [41]. They defined AI as an area of engineering and science that focuses on creating computer systems that can imitate human behavior, i.e. intelligent systems based on computation [42]. In 1950, Turing presented the proposal for what is now known as the "Turing Test" [43]. It raised the possibility that a machine could be considered intelligent if, during a conversation, a person could not discern whether the answers came from a machine or another human being [43]. This test has had a significant impact on the development of **AI** over the years.

Human intelligence is an extraordinary and multifaceted phenomenon that has been the subject of study and admiration throughout history. It involves the ability to understand, learn, reason, make decisions, solve problems, interpret sensory information and adapt to new situations flexibly [44]. Human intelligence is deeply rooted in the complexity of the human brain, with billions of interconnected neurons that work together to process information and create abstract thoughts.

Intelligence, being one of the most distinctive characteristics of the human being, gives **AI** significant intrinsic value. Over the years, there has always been an incessant quest to understand the mechanisms of thought, perception, reasoning and action in a complex world. However, nowadays, human beings are not only limited to understanding these mechanisms, but also to building intelligent entities [44]. **AI** is grounded in various fields of knowledge, such as philosophy, mathematics, economics, medicine, among many others [45].

The importance of **AI** also lies in its ability to solve complex problems in an agile, effective and economically efficient way, providing solutions that can be reproduced and documented [46]. Despite the benefits, this approach faces the challenge of developing systems capable of performing tasks that are intuitive for humans, but difficult to formalize for implementation in machines [47].

The aim of **AI** is to develop computer systems capable of performing tasks that, if performed by humans, would require intelligence, learning ability and adaptability. **AI** encompasses several subfields, ranging from **ML** to natural language processing, including computer vision and robotics [48]. Research in **AI** is motivated by a deep desire to mimic human intelligence, allowing complex problems to be solved and decisions to be made based on data [42].

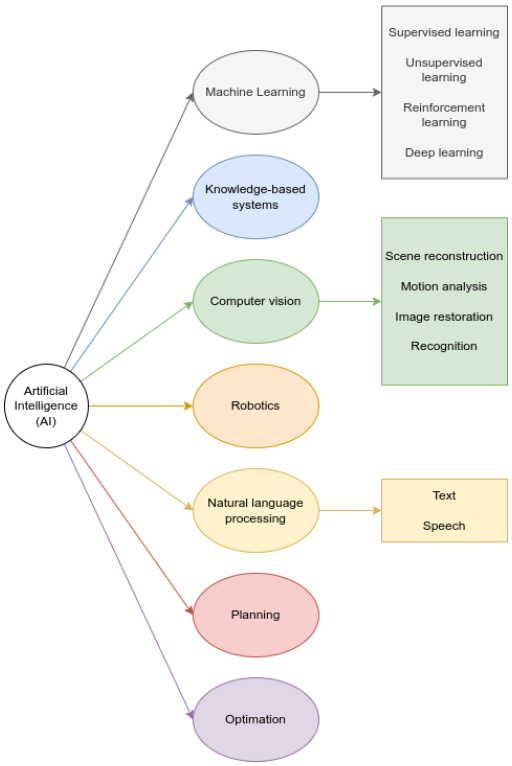


Figure 7: Subfields of **AI**, adapted from [48]

2.2.1 Machine Learning

AI is a constantly evolving field, and **ML** is one of its most prominent subfields. Instead of explicitly programming a computer to perform a specific task, **ML** algorithms allow machines to learn to perform tasks based on data and their experience. In other words, **AI** is a field of study that covers a much wider range than **ML**. **AI** focuses on the ability to make machines intelligent through various approaches, while **ML** focuses primarily on creating machines capable of learning to perform tasks [49]. The essence of this subfield lies in the ability of computers to identify patterns in data and adjust their behavior accordingly, using statistical techniques and algorithms that allow the machine to make predictions or decisions based on past information [50].

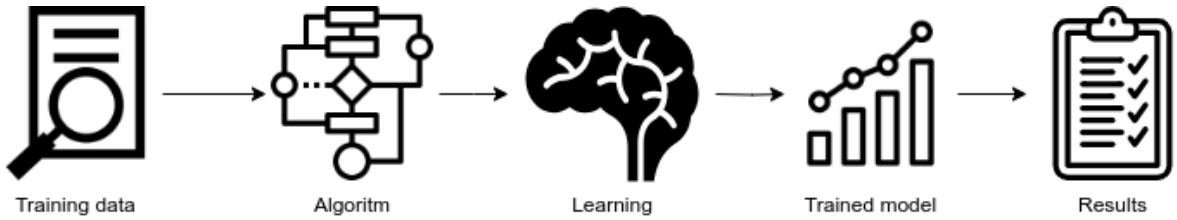


Figure 8: Pipeline of the **ML** process

The complexity of the **ML** field is evidenced when facing challenges in more complex problems, such

as predicting the performance of the stock market in the coming years or recognizing the entity of an object in complex images. To deal with these challenging scenarios, the **ML** technique seeks to create algorithms that learn comprehensively, without relying on fixed rules that detail specific program steps [51].

ML algorithms learn by identifying patterns, acquiring knowledge through experience and observational analysis of the data [50]. Each case in the training datasets has specific characteristics and can include a label that indicates what the algorithm should identify or predict [51]. The main objective is to develop a model capable of determining these labels when presented with new data sets containing previously unseen instances of the same problem [51].

There are several approaches to using **ML**, known as **ML** [52] techniques:

- **Supervised learning:** In this technique, the main objective is to find a function that can map the inputs of a dataset to the desired outputs. This approach requires human intervention, since the training data set includes labels that indicate the correct answers. This technique is widely used in classification problems, where the aim is to separate the data into categories, and regression, to predict continuous results based on the characteristics of the data [53].

- **Unsupervised learning:** Unlike the supervised approach, unsupervised learning does not require human involvement to label the data. This technique is used to identify patterns, relationships and groupings in the data. A practical example would be the use of clustering algorithms to group data without having prior knowledge of the existing categories [54].

- **Reinforcement learning:** In **Reinforcement Learning (RL)**, the main focus is on teaching an agent to make decisions when interacting with an environment. Learning occurs through trial and error, where the agent receives rewards or penalties based on its actions. Over time, the agent learns to perform actions that maximize the rewards. This technique is commonly applied in games, robotics and in situations involving sequential decision-making, such as autonomous cars, where they have to learn to navigate complex environments [55].

- **Semi-supervised learning:** The semi-supervised learning approach is a hybrid technique that works with both labeled and unlabeled data. This technique is especially useful when there is a shortage of labeled data, but a considerable amount of unlabeled data available. In these cases, it is not as beneficial to follow a supervised approach, as this can be complemented by the additional information from the unlabeled data. Practical examples of this technique include fraud detection and machine translation, where the combination of labeled and unlabeled data contributes to more robust results [56].

- **Deep learning:** **DL** is an approach that uses a mechanism called **Artificial Neural Network (ANN)** to obtain complex, hierarchical representations of data that simulate the human brain. These networks are made up of several layers of neurons that learn to identify patterns at different levels of abstraction [57]. The **DL** is widely applied in complex tasks such as image recognition, natural language processing and games. For example, in facial recognition, it allows the model to automatically learn discriminative features such as edges, textures and unique facial patterns [58].

Each approach has its specific applications, depending on the nature of the problem and the availability of labeled data. Choosing the right technique plays a crucial role in developing effective **ML** models.

However, nowadays, the most widely used paradigm is the **DL** technique, due to its ability to adapt to complex problems. **DL**, is notable for its ability to adjust to both supervised and unsupervised problems, bringing remarkable innovations in various areas [57]. The main feature of **DL** is the use of **ANN** architectures [59].

An **ANN** is a complex computational structure that aims to acquire and store knowledge. It is made up of artificial neurons, which are processing units inspired by human biological neurons. These neurons are interconnected using vectors and matrices with artificial weights, which allows the **ANN** to efficiently mimic the human learning process. This implementation is designed to meet specific requirements and guarantee high performance in the acquisition and storage of knowledge [59].

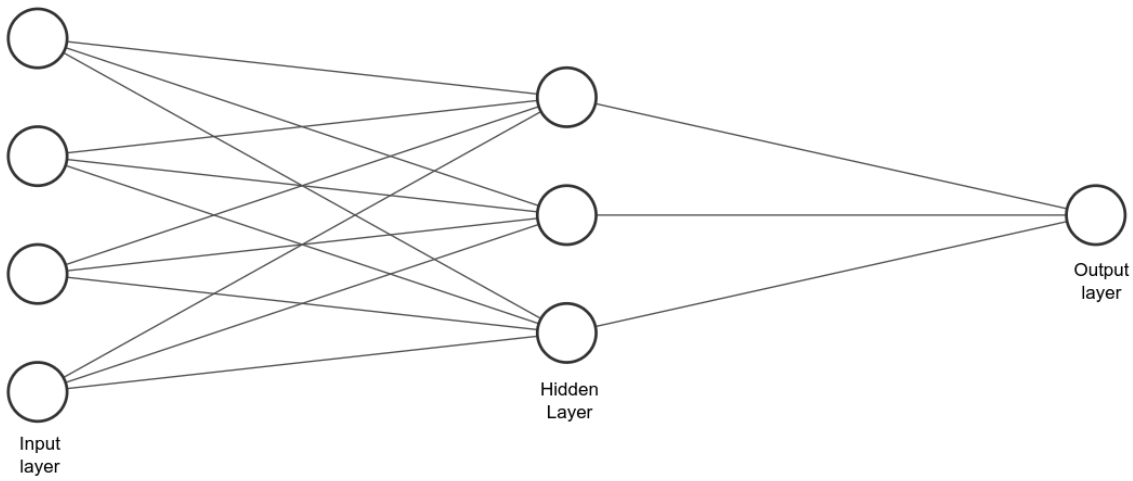


Figure 9: Example of the architecture of a **ANN**, adapted from [60]

A **Deep Neural Network (DNN)**, is an advanced and specific category of **ANN**, which plays a key role in various **AI** applications. These networks are important because of their ability to process information and learn complex representations of data, especially in problems involving large volumes of information. Unlike a conventional neural network, a **DNN** has multiple hidden layers, which perform progressively

more abstract transformations of the input data [60]. Each layer operates with a set of weights and activation functions, adjusting these parameters during training to optimize the network's performance. The hierarchy of concepts in a **DNN** allows each layer to extract specific characteristics from the data, representing increasing levels of abstraction [59]. For example, in an image recognition application, the first layers can learn to detect simple edges and textures, while deeper layers can identify more complex features, such as specific shapes and patterns [61].

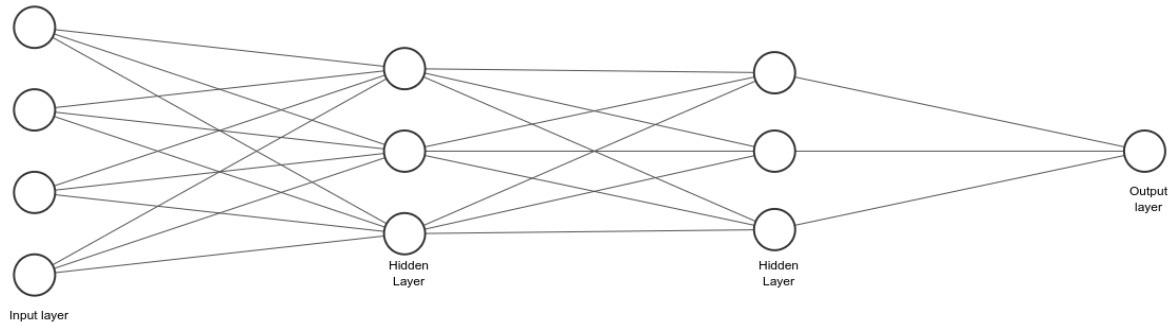


Figure 10: Example of the architecture of a **DNN**, adapted from [60]

With this, it can be seen that **ANN** are considered an extremely powerful tool in the field of **ML**, as they offer solutions to a wide range of complex problems [57]. Their potential is clearly evident in a variety of contexts, from tasks that require explicit learning of the relationships between inputs and outputs, to situations in which **ANN** can intrinsically exploit the structure of the data [59]. This adaptability allows **ANN** to be used in both supervised and unsupervised problems [59]. Therefore, these mechanisms become crucial in the development of advanced solutions in **AI** and **ML** [57].

As explained earlier, **ANN** are complex computational structures that seek to simulate the learning process of the human brain [60]. Composed of artificial neurons, these processing units are interconnected through vectors and matrices with artificial weights [59]. The architecture of **ANN** meets the essential requirements for simulating human learning.

- **Dynamic adaptation with experience:** As the **ANN** is exposed to successive cases in a data set, the artificial weights are dynamically adjusted. This process allows for continuous and adaptive learning, reflecting the ability to adjust based on accumulated experience [59].

- **Learning capacity independent of human intervention:** The **ANN** is able to extract relationships between the attributes of the cases in the dataset without depending on human intervention. This autonomy in learning makes it possible to automatically identify complex patterns and relationships [59].

- **Ability to generalize:** After acquiring knowledge, **ANN** demonstrates the ability to come up with innovative solutions. This means that, unlike some conventional algorithms, **ANN** can offer answers to situations not previously seen, highlighting its ability to generalize [59].

- **Data organization and clustering:** In addition to learning, **ANN** is capable of organizing data into groups using techniques such as clustering. This ability makes it easier to identify patterns and structures in datasets, contributing to a deeper understanding of the relationships present [59].

- **Fault tolerance and robustness:** If parts of the **ANN** are impairing learning, the corresponding neurons become less relevant, demonstrating fault tolerance. The distributed architecture of **ANN** allows for the efficient storage of knowledge, making them robust even in challenging scenarios [59].

The structure of an **ANN** is intrinsically robust, which makes it a highly promising approach for solving complex problems. A basic example of this capability is detecting the type of clothing in a fashion context [62]. To further illustrate this application, one can imagine using an **ANN** to anticipate the style of clothes based on fashion images. In this scenario, the input layer would be made up of a number of neurons equal to the number of pixels in the images in the data set. The output layer would be made up of neurons representing different categories of clothing, such as sweaters, pants, dresses, among others.

Between these layers, several hidden layers could be added, as illustrated in figure 10, each specialized in capturing particular characteristics of the clothes, such as cuts, styles or decorative elements. The information would be transmitted between the layers by means of weighted connections, where the weights associated with each connection would reflect the relative importance of certain characteristics in detecting the type of clothing [59]. After the weighted sum of the inputs and the addition of a bias, this information would go through an activation function, determining whether the neuron would become active [59]. This process would be repeated throughout the layers until a neuron in the output layer was activated, indicating the type of clothing detected by the network.

During training, the **ANN** would continually adjust the weights of the connections and the biases, refining its ability to identify relevant patterns in the images. The emergence of these networks not only improved existing solutions, but also broadened the applications of **DL** [57]. In addition to fashion, the versatility of this approach can be applied in a number of other areas, driving continuous innovation in the field of **AI** [57].

2.2.2 Computer Vision

The discipline of **CV** is essential for the creation of artificial systems capable of acquiring visual information from various sources, such as images and videos [63]. Using methods based on statistics, optimization and geometry, this discipline is a complement to the biological system of vision. In addition, **CV** encounters intersections with various disciplines, such as **AI**, neurobiology, physics, signal processing, imaging, among several other areas [64, 65].

The tasks carried out in the field of **CV** cover crucial areas, such as recognition, movement, reconstruction and restoration [14, 63]. The systems perform a series of steps, from image acquisition to high-level processing, involving the extraction of characteristics and the detection and segmentation of points and regions of interest [66]. **CV** consists of several crucial tasks, among which are image classification, object detection and image segmentation [63].

Image classification refers to the process of assigning one or more categories to an image. This procedure is essential in applications that require accurate identification of visual content. Several models, mostly based on **Convolutional Neural Network (CNN)**, have been widely employed for this task [67]. These models learn patterns and characteristics in images, empowering themselves to recognize and classify objects with remarkable accuracy [68].

Object detection goes beyond classification by focusing on the exact location of objects in an image and assigning them a specific tag. This task is fundamental in scenarios where it is crucial not only to identify, but also to delineate the position of the objects [68]. Currently, the most widely used detection algorithms in this area are **You Only Look Once (YOLO)** [69] and Faster R-CNN (**CNN** based on regions) [70], which can be used in practical applications, such as surveillance systems, autonomous vehicles and image indexing.

Image segmentation focuses on subdividing an image into significant regions, simplifying the understanding and analysis of visual content [68]. There are two main approaches: semantic segmentation, which assigns labels to each pixel to identify objects and regions, and instance segmenting, which distinguishes individual objects in an image [63]. Models such as the U-Net [71] and Mask R-CNN [72] are the most widely used models today for segmentation tasks. Some practical applications include medical diagnostics [71], urban scene interpretation, and augmented reality.

These are just a few examples of tasks and templates in **CV**. The field is constantly evolving, with new architectures and approaches to be developed to deal with specific challenges in various applications [63].

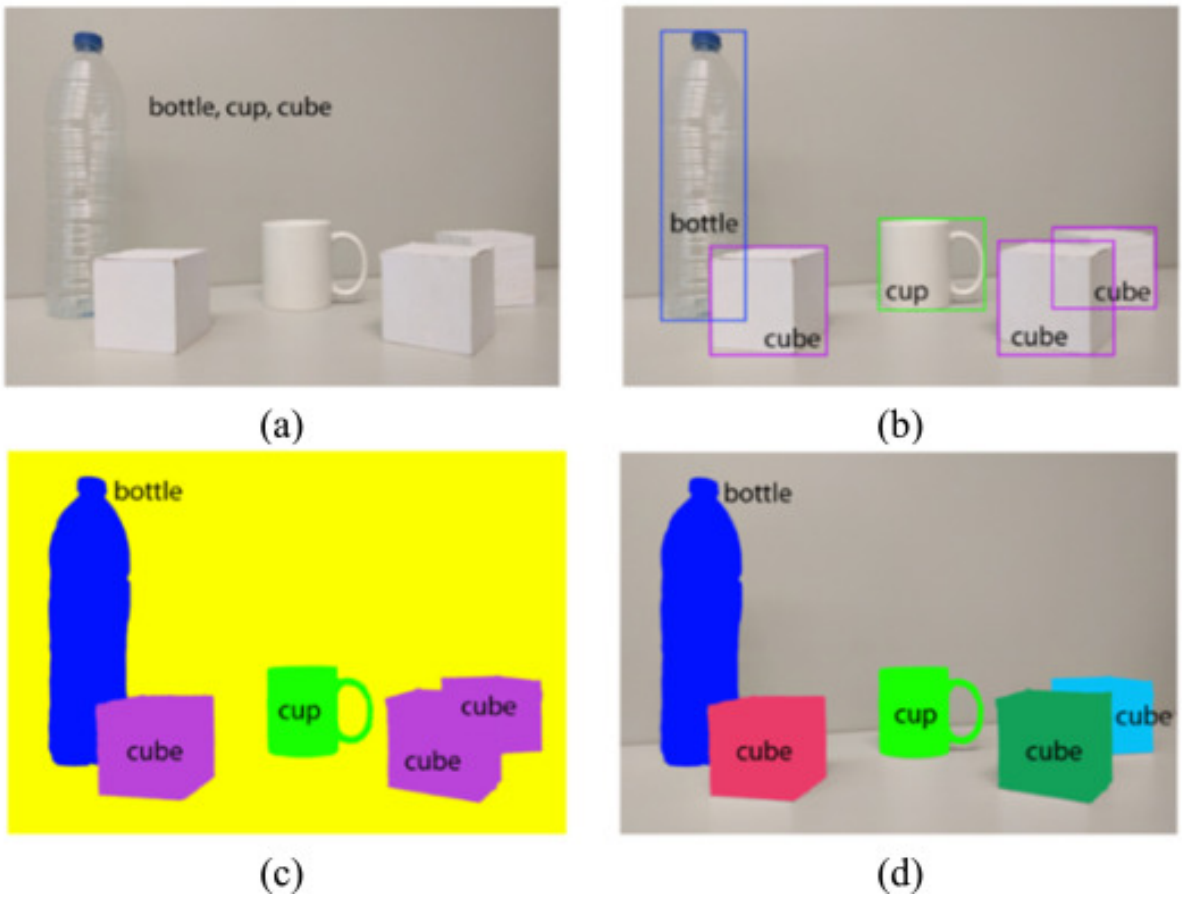


Figure 11: Example of various kinds of visual perception tasks: (a) image classification; (b) object detection; (c) semantic segmentation and (d) instance segmentation, extracted from [63]

CV is not limited only to these tasks, such as image classification, object detection, and segmentation. This discipline covers a wide range of applications, including video understanding, image generation, 3D vision, multimodal tasks, and self-supervised learning [63]. From interpreting dynamic scenes in videos to creating realistic synthetic images, **CV** plays a crucial role in a variety of fields, demonstrating itself as a versatile tool for understanding and processing visual data in various modalities and contexts [63].

The practical applications of **CV** are vast and impactful, from industrial process control to event detection and monitoring in surveillance systems [13]. The discipline plays a crucial role in the organization of information such as the indexing of image databases, and in the modeling, construction, and reconstruction of objects, such as in medical imaging analysis [71, 73]. Object recognition and tracking, pattern recognition, and image analysis are just some of the areas in which **CV** plays a vital role [14].

2.2.3 Explainability

Explainability in **AI** refers to the ability to describe the inner workings and decisions made by a model in a way that is understandable to human beings [74]. In intelligent surveillance applications, such as the detection of violence and vandalism, explainability is crucial for several reasons.

Firstly, explainability is vital for effective interaction between humans and automated systems. In surveillance contexts, operators and analysts often need to interpret and act on decisions generated by **ML** models. Explainable models allow these professionals to understand and validate the systems' recommendations, adjusting their actions based on a clear understanding of the criteria that led to a particular decision. This is especially relevant in critical situations where quick decisions are required and confidence in the system must be high [74].

Secondly, the trust and adoption of surveillance systems depends on the transparency of the models. Users and administrators need to understand how and why a model reached a certain conclusion in order to trust it [74]. Furthermore, in the event of failures or incorrect decisions, it is essential to identify where and why the error occurred in order to assign responsibility and improve the systems [74].

Explainability is important for legal and ethical compliance. Many jurisdictions have legal requirements that demand transparency and explainability in AI systems, especially those used in security and surveillance [75]. From an ethical point of view, automated detection of suspicious behavior involves risks of bias and discrimination. Explainability helps mitigate these risks by allowing critical evaluation of the model's decisions [74, 75].

Another relevant aspect is the continuous improvement of models. Understanding the model's decisions makes it easier to identify areas for improvement, adjusting the algorithms for better performance and lower false positive/negative rates [76].

In intelligent surveillance for detecting violence and vandalism, explainability can be applied in various ways. In anomaly detection, techniques such as autoencoders can be used to identify abnormal behavior and explain it by comparing it to normal behavior [77]. In image and video analysis, **CNN** are explainable through heatmaps and methods such as **Gradient-weighted Class Activation Mapping (Grad-CAM)** can identify which regions of the video or image contributed to the detection of violent behavior or vandalism [78]. For real-time intervention, systems that provide immediate explanations to security forces or human operators help in making quick and informed decisions [74].

2.2.4 Video surveillance datasets

Datasets play a crucial role in the development and evaluation of **AI** models. As such, the video surveillance area takes advantage of these datasets to train algorithms efficiently and ensure their ability to generalize in real-world scenarios. The quality and diversity of a dataset are determining factors in the performance and robustness of the resulting model. A quality dataset not only reflects the variability and complexity of the video surveillance environment, but also covers a wide range of conditions, such as different lighting conditions, climate variations and visual perspectives [79, 80].

The importance of a high-quality dataset is evident in the need to face specific video surveillance challenges, such as accurate detection of moving objects, identification of suspicious behaviors and facial recognition in different conditions [79]. The diversity of the dataset allows the model to learn how to deal with complex situations that may arise in the real world, ensuring its adaptability [80].

Table 3: Useful video surveillance datasets

Dataset	Description	Size	Article
UCF-Crime	Contains real scenarios of video abnormalities	1900 videos with 13 categories of anomalies	[81]
RWF-2000	Contains videos captured by real surveillance cameras	2000 videos with abnormal behavior	[82]
Real Life Violence Situations Dataset	Contains videos with violence and normal everyday videos	1000 violence videos and 1000 non-violence videos	[83]
XD-Violence	Large-scale audio-visual dataset for violence detection	4754 videos with audio	[84]
UBI-Fights	Dataset containing a diversity of violence scenarios	216 videos with violence and 784 of everyday situations	[85]

The use of datasets specialized in anomaly detection, such as the datasets mentioned in the table 3, not only allows the creation of more efficient surveillance systems in the identification of suspicious behaviour, but also drives continuous progress in this area, providing a solid basis for research and devel-

opment of more advanced security techniques [80].

The practice of training models with multiple complementary data sets is crucial for improving robustness and generalization of computer vision models. The use of diversified datasets allows models to learn how to handle a wide variety of scenarios, contexts, and characteristics, thereby improving their ability to recognize in real-world conditions [80].

For example, a model that needs to perform action recognition may benefit from training with a dataset of this type, such as UCF101 [86]. This dataset covers a wide range of human actions captured in videos, providing a solid basis for the development of models capable of identifying movements and behaviors [86]. On the other hand, if the model to be developed needs to perform facial recognition, a dataset to be incorporated into the training could, for example, be the SCface [87]. This dataset contains a wide variety of images of human faces, covering different ages, ethnicities and facial expressions [87]. The approach of using complementary datasets not only enriches the knowledge of the model, but also contributes to its ability to deal with specific challenges. However, it is essential to maintain a balance and ensure that the datasets are representative of the actual application conditions to avoid unwanted biases [79, 80].

2.2.5 Machine and Deep Learning models for video surveillance

In the current context, society faces increasingly complex challenges related to the detection of violence and vandalism in data from video surveillance systems. It is essential to anticipate and monitor unwanted events in the technological scene in order to improve security and prevent incidents [19]. The development of **AI** models for this purpose plays a crucial role in this regard, and the techniques of **DL** emerge as protagonists by presenting remarkable results in the detection of patterns associated with violence and vandalism in surveillance videos [28]. These techniques have the ability to process large volumes of data, represent complex functions, and identify relevant patterns and characteristics in visual data [57]. The use of more conventional approaches reveals limitations in the manipulation of extensive datasets and in the understanding of complex relationships between them, so they are not intended for these problems [49].

In this context, it is important to highlight the model architectures embedded in the **DL** techniques most relevant for the detection of violence and vandalism in video surveillance. Given the focus of this dissertation on visual data, specific architectures adapted to handle this type of information will be addressed. In addition to the singular architectures, compositions that combine different approaches, known as hybrid architectures, will be discussed with the aim of improving the effectiveness in the detection of unwanted behaviour.

2.2.6 Singular architectures

Multilayer Perceptron (MLP)

The **MLP** architecture is a widely used class of **ANN** that stands out for its ability to learn complex patterns in datasets [88]. Composed of several layers of neurons, **MLP** is characterized by having a hierarchical structure, in which each layer processes and extracts characteristics of input, contributing to the creation of more sophisticated representations as information is transmitted through the network [88].

The **MLP** structure is usually composed of an input layer, one or more hidden layers, and an output layer [88]. This model architecture has a feedforward nature, which stands out by the unidirectional flow of information through the layers of the network [89].

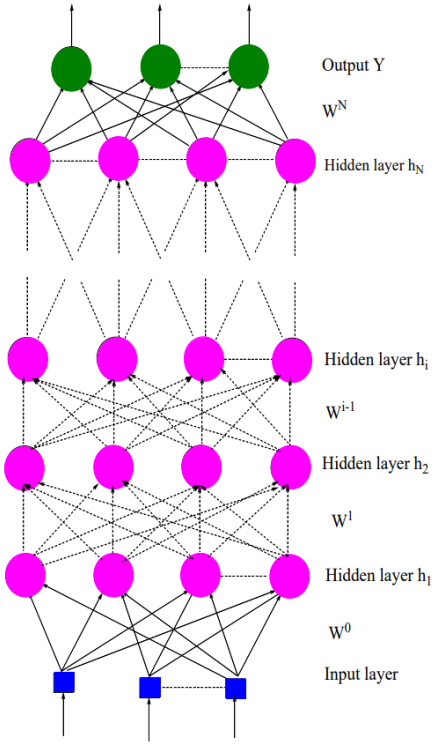


Figure 12: Example of an architecture of **MLP**, taken from [89]

This flow begins in the input layer, passes through the hidden layers and ends in the output layer. In each layer, the neurons are interconnected with all neurons in the next layer and each connection has an assigned weight. The inclusion of nonlinear activation functions enables **MLP** to be able to learn non-linear relationships in data [88, 89].

Each layer processes the information and acquires complex hierarchical representations of the data, enabling the neural network to extract relevant features into video surveillance images [90].

CNN

CNN have increasingly emerged as a powerful tool in the area of **ML**, especially in the **CV**. Its architecture, inspired by the human visual cortex, has enabled significant advances in tasks such as image recognition [91], object detection [92], among several others.

The basic architecture of a **CNN** consists of several distinct layers, each playing a specific role in the learning process. The convolutionary layer is responsible for processing and extracting features in the data through convolutional operations, allowing patterns to be detected. The non-linearity layer introduces non-linarity into the algorithm by applying activation functions, such as the *ReLU*, *Tanh*, or *Sigmoid* functions. The pooling layer, in turn, reduces the dimensional space of the data, simplifying the computational complexity of the network. Finally, the fully connected layer acts as a traditional neural network, using the extracted features to make predictions or final classifications [93, 94].

The training of the **CNN** is conducted through the backpropagation algorithm, in which the error between the network predictions and the actual labels is propagated to adjust the connections weights. This iterative process enables the network to improve its internal representations and improve its performance over time [94].

The flexibility of the **CNN** architecture allows these to be adapted according to the specific requirements of each problem. With layers such as convolutional, non-linear, pooling and fully connected, the structure can be adjusted to optimize the extraction of relevant features, especially in complex datasets, such as images [93, 95]. The ability of **CNN** to extract important features from data makes them adaptable to a variety of problems, allowing their application in different contexts, such as the aforementioned [94].

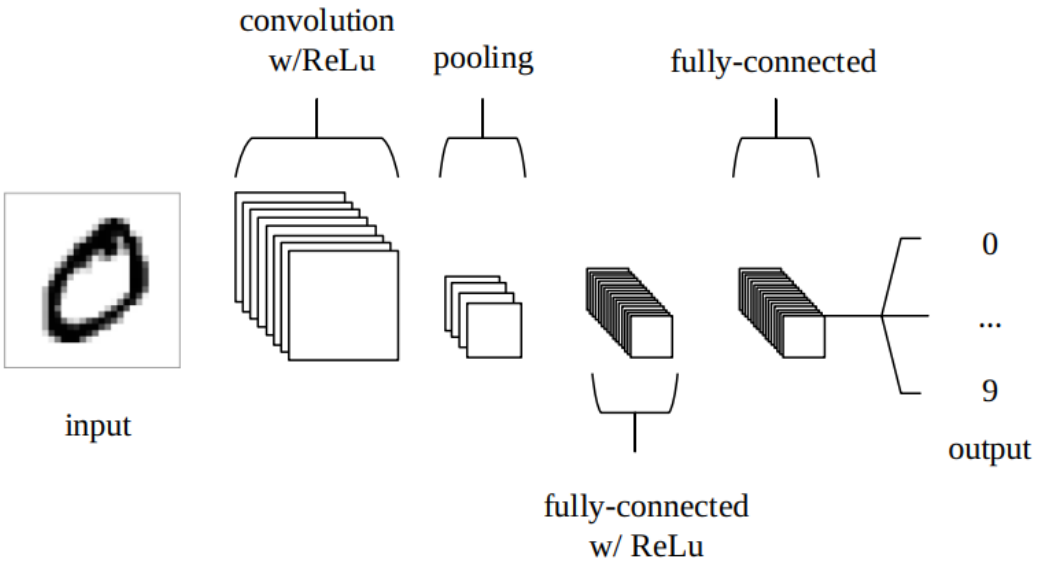


Figure 13: Example of an architecture **CNN**, adapted to the digit recognition problem, taken from [94]

Recurrent Neural Network (RNN)

RNN are a class of architectures specially designed to handle sequential and temporal data. Unlike the traditional **ANN**, **RNN** have recurring connections, allowing information to be retained and propagated over time, being effective in tasks where order and time dependency are crucial [96].

The central element of **RNN** is the recurring neuron, which has a kind of internal memory. This neuron receives inputs from the current data and also from its own output in the previous time stage, establishing a time relationship. **RNN** processes data sequentially, one time step at a time [96]. Each step considers the current input and retained information from previous steps, allowing the modeling of time patterns. The ability of **RNN** to maintain an internal memory makes them suitable for learning sequence dependencies, which is valuable in tasks where the order of elements is crucial [96].

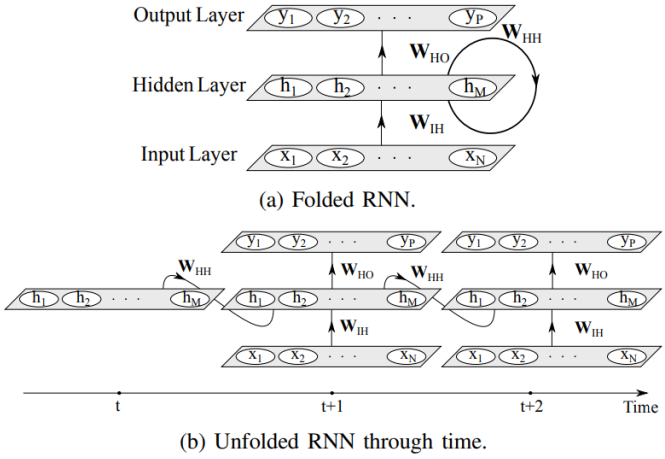


Figure 14: Example of an architecture **RNN** and how it behaves over time, taken from [96]

RNN are widely used for time series prediction, natural language processing, speech recognition, and machine translation, for example [96]. All these use cases involve tasks such as those mentioned above, where there is an ordering of the elements [96]. Although traditional **RNN** are not the primary choice for direct image processing, variants and hybrid approaches can be explored in specific contexts [95].

Other problems relating to this architecture arise during training. These networks may face problems with regard to gradient vanishing or explosion, making it difficult for the network to learn long-term dependencies. Standard **RNN** may also have difficulty dealing with very long sequences due to loss of information over time. This can be mitigated by techniques such as gradient clipping or by more advanced architectures such as **Long Short-Term Memory (LSTM)** and **Gated Recurrent Unit (GRU)**, which will be addressed next [96].

LSTM

RNN face difficulties in effectively capturing long-term time dependencies, which can be essential in various tasks. **LSTM** emerged as a solution to this specific challenge, featuring a more advanced architecture capable of preserving information for prolonged periods [97, 98].

One of the main innovations in **LSTM** is the inclusion of memory cells, which are composed of three essential gates: the input gate, the forget gate, and the output gate [97]. The input gate has the function of regulating the amount of new information that will be stored in the memory cell, being responsible for deciding which information from the input is to be remembered. The forget gate allows the memory cell to discard irrelevant or less important information. This role is crucial to keeping only the most relevant information over time. Finally, the output gate controls the amount of information that will be sent to the

memory cell output. This gate is responsible for deciding which information that is retained in the memory cell should be sent to the next layer [98].

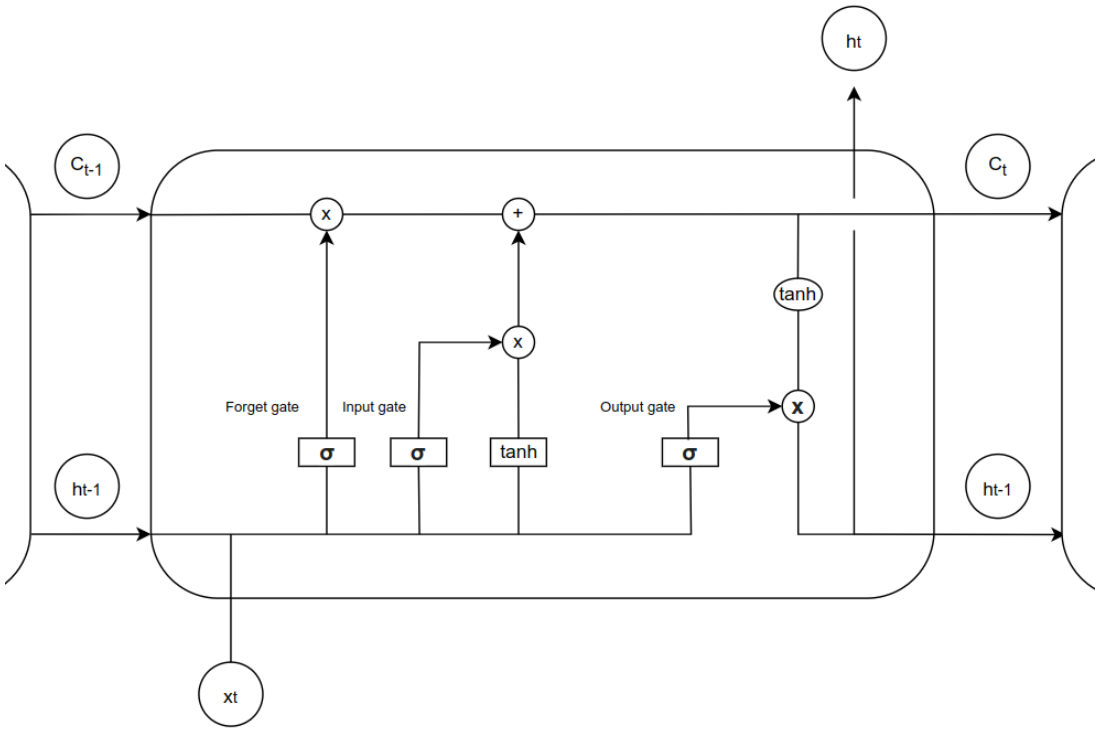


Figure 15: Architecture of a block of a **LSTM** network, adapted from [98]

The **LSTM** architecture works through a sequential operation where at every moment of time, the gates (input, forget and output) are triggered. These gates are controlled by activation functions (usually sigmoid) that determine the information to retained, forgotten or forwarded. Then the memory cell is updated, storing new relevant information and discarding the less important ones. Finally, the output value of the memory cell is calculated and transmitted to the next time step [97].

This memory cell structure allows **LSTM** to capture and maintain relevant information for long periods, solving the problem of gradient vanishing. This model architecture is able to learn long-term temporal dependencies, making them suitable for tasks where understanding the historical context is essential, for example, in surveillance videos [99].

GRU

The **GRU** architecture is an advanced variant of the **RNN** that was created with the aim of overcoming the challenges in long-term time dependency modeling, as well as the **LSTM** networks [100]. **GRU** have the ability to retain relevant information for long periods of time, but have a slightly more simplified architecture compared to the **LSTM** [100].

This architecture introduces two main mechanisms: update (z_t) and reset (r_t). The update mechanism controls the amount of information from the previous memory cell to retained, while the reset mechanism decides how much information of the current input should be forgotten [100]. **GRU** uses a single memory cell, eliminating the need for a separate memory cell and a forgetting gate, which simplifies architecture compared to **LSTM** [100].

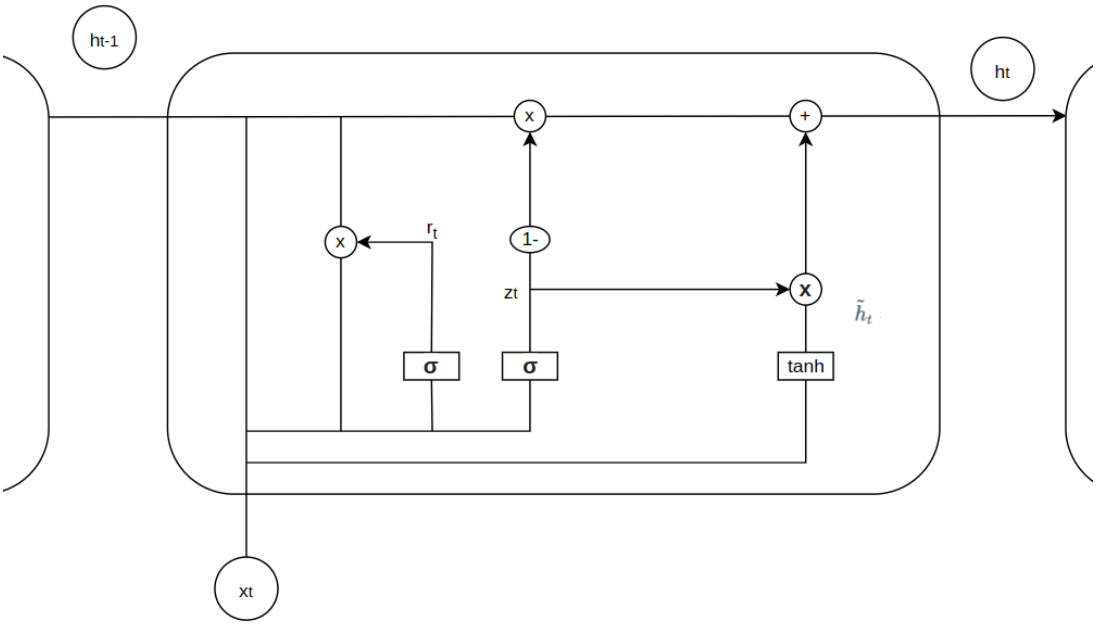


Figure 16: Architecture of a block of a **GRU** network, adapted from [100]

The basic operation of a **GRU** can be summarized as follows: At each time step, the **GRU** calculates two values. One for the update gate (z_t) and another for the reset gate (r_t). These gates control the amount of information that will be retained or discarded, as mentioned above. After this, a hidden state candidate is calculated ($\tilde{h}_t$), which represents the new information to be considered in the current timeframe. Next, the current hidden state (h_t) is updated considering the update and reset ports [100].

Fundamentally, the gates of a **GRU** enable the combination of past and present information to be controlled, resulting in an effective way of dealing with temporal dependencies [100]. This feature makes **GRU** valuable in a variety of tasks, such as natural language processing [101], time series analysis, and other applications where the time order of data is important [102].

Transformer

The Transformer model architecture, originally developed for natural language processing tasks [103], demonstrates to be highly versatile and efficient when applied to datasets from video surveillance [104]. The inherent flexibility of the Transformer models, together with their ability to process variable-length

data sequences, offers significant opportunities to improve surveillance video analysis, allowing a deeper, contextualized understanding of captured information [104].

The Transformer introduced a crucial innovation with the use of attention mechanisms [103], boosting its effectiveness in various tasks. Unlike recurring sequential approaches and convolutions, the Transformer capitalizes on the Multi-Head attention mechanism. This mechanism, by means of queries and pairs key-value, maps parameters into outputs, using dot product operations in a parallel and efficient way [103].

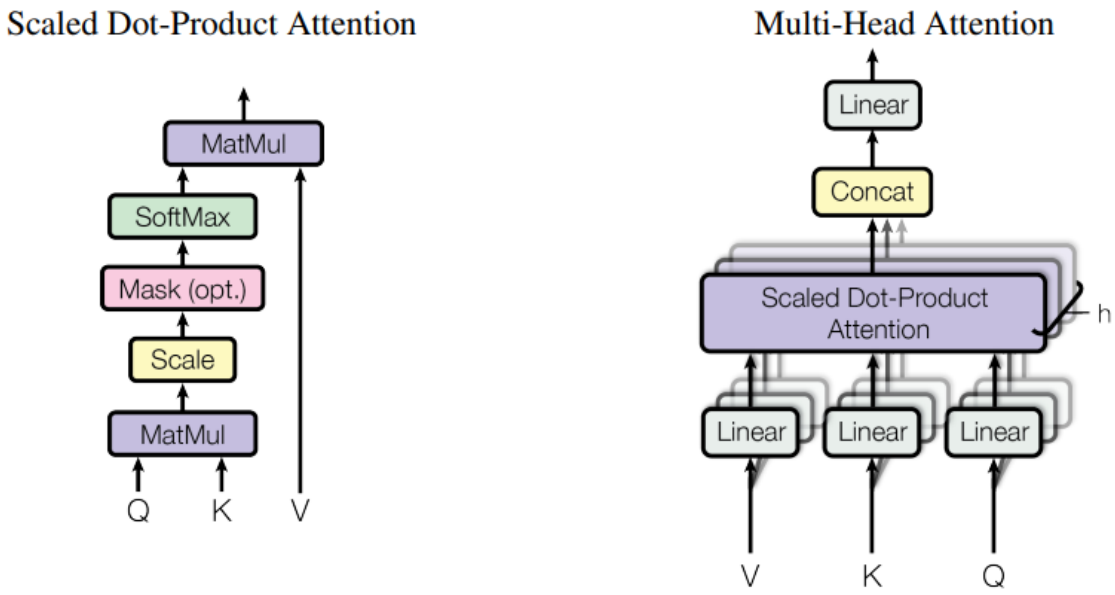


Figure 17: Attention mechanisms used in the Transformer architecture, taken from [103]

In a Transformer model, both mechanisms, Scaled Dot-Product Attention and Multi-Head Attention (present in the figure 17), are usually used together. The Transformer architecture incorporates Multi-Head Attention as an extension of the basic attention mechanism (Scaled Dot-Product Attention). The Multi-Head attention mechanism allows the exploration of different spaces of representation of data and positions, finding dependencies between distinct points in the data. This approach is beneficial for processing global dependencies in sequential data quickly and effectively [103].

The general architecture of the Transformer is composed of a encoder and a decoder, both of which are made up of layers. In the encoder, a sub-layer applies the attention mechanism to the inputs, followed by a feedforward layer. On the decoder side, there are three sub-layers, two of which are similar to the encoder, and one that applies the Multi-Head attention mechanism to values after the transformation of the encoder [103].

A crucial feature for preserving the order of data in sequence, given the Transformer's non-sequential approach, is Positional Encoding. Included in each layer of the encoder and decoder, this encoding ensures proper consideration of the order of the data during processing [103].

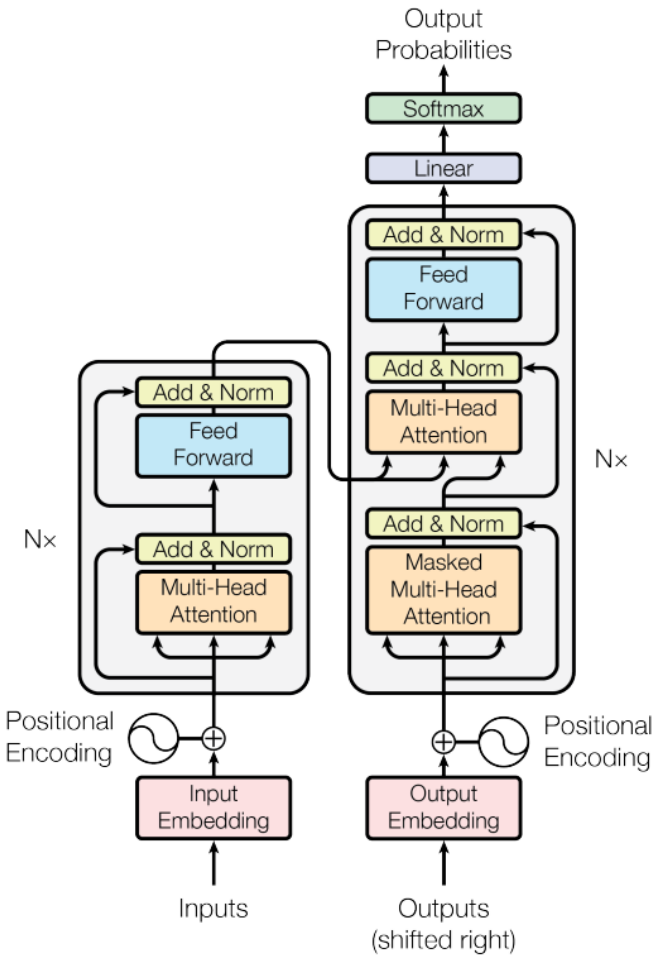


Figure 18: Example of a Transformer architecture, taken from [103]

In cases involving video sequences, such as video surveillance, attention in the Transformer model is extended to incorporate the temporal dimension. This allows the model to capture information not only in spatial terms, but also over time, enriching the understanding of dynamic events [104].

The attention mechanism in the Transformer, adapted to the visual context, stands out as a versatile tool for **CV** tasks. Its ability to capture complex spatial and temporal relationships, take advantage of the power of parallelism and contextual attention, represents a significant milestone in the field of neural networks and makes it a powerful choice for image and video analysis [103, 104].

2.2.7 Hybrid architectures

Hybrid architectures within **ML** symbolize a strategic fusion of distinct models and approaches with a view to improving performance in specific tasks. This combined approach seeks to harness the best of different worlds by integrating unique features from different architectures to enhance efficiency and effectiveness in specific applications [95].

These architectures highlight the adaptable and flexible nature of **DL**, highlighting that the strategic combination of distinct paradigms can result in remarkable progress [57]. By exploring this fusion of techniques, researchers and programmers have the opportunity to shape more robust and effective models to address the complex challenges presented in the real world [57].

CNN-RNN

Approaches that combine **CNN** with **RNN** represent an effective strategy for capturing both spatial and temporal information in video surveillance data [105]. This initial combination can evolve into more advanced architectures, such as combining **CNN** with **LSTM** or **GRU** [102, 106], further improving the model's ability to understand complex patterns over time [107].

This approach has been developed to deal with complex tasks that require the capture of spatial and temporal patterns [105, 106]. By combining the efficiency of **CNN** in the extraction of visual characteristics with the ability of the **RNN** to model long-term time dependencies, this architecture is promising [107].

The function of **CNN** layers is to extract relevant spatial characteristics from input data, such as images [107]. This ability enables the network to learn hierarchical representations, identifying complex visual patterns on different scales. The **RNN** units come into action to process the sequence of characteristics extracted by the **CNN**. This combination allows the architecture to deal efficiently with time dependencies, retaining important information over time [18, 99].

This hybrid architecture has proven to be highly effective in video recognition tasks, as it allows to understand both visual and sequential characteristics over time [107]. This approach is especially useful for identifying actions in videos and tracking moving objects [106].

Despite the advances achieved by combining these architectures in various applications, there are still challenges to be overcome, such as fine hyperparameter adjustment and computational complexity [106]. However, the potential of this architecture to cope with complex tasks continues to motivate research and innovation, paving the way for the continuous development of more robust and efficient models [18, 99].

Transformer-RNN

This hybrid architecture emerges as a revolutionary approach in combining the Transformers, recognized for their effectiveness in natural language processing, with the **RNN**, specialized in modeling temporal dependencies [108]. This strategic merger aims to overcome specific challenges in video surveillance scenarios, in which understanding both spatial and temporal information is fundamental [104].

The Transformer block acts as an encoder, capturing spatial information and building robust representations of visual features. Their ability to focus allows the network to focus on different parts of the scene, highlighting areas of interest for spatial understanding [103]. The **RNN** units enter the scene to model time dependencies between the consecutive frames of a video. This is crucial to understanding the evolution of scenes over time. Choosing between **LSTM** or **GRU** depends on the specific needs of the task, considering factors such as time complexity and computational efficiency [104]. Contextual attention, a fundamental principle of the Transformer, is strategically applied to the **RNN** layers. This allows the network to devote more attention to certain parts of the temporal sequence, dynamically adapting to changing conditions in the video surveillance scene [103, 104].

The ability of this architecture to model complex time dependencies makes it suitable for the detection of time anomalies in surveillance videos [108]. It is able to identify unusual patterns that may indicate suspicious activities or abnormal events. By incorporating contextual time information, tracking of moving objects is also improved [108]. This is essential in situations where time continuity is crucial for an accurate understanding of the trajectories of objects. On the other hand, in tasks that require understanding of intrigued temporal patterns, such as acting or behavior recognition, this architecture also stands out, learning rich, hierarchical temporal representations [104].

The combination of the Transformer with **RNN** offers a revolutionary approach, but there are still challenges to overcome, such as the efficient management of extensive time sequences and adaptation to dynamic contexts [104, 108].

2.3 Literature Review

In this section, several literature studies are presented dealing with the integration of intelligent surveillance systems in the field of video surveillance. This bibliographic review plays a key role in supporting and guiding this research, which aims to develop innovative solutions in the context of intelligent surveillance. The following is a concise summary of each of the studies analysed, followed by a critical analysis to highlight valuable information relating to each of these investigations.

The research carried out by Liang Ye et al. [28] focuses on the identification of violence in campuses

through the use of image and acoustic characteristics. To achieve this goal, a **Convolutional 3D (C3D)** neural network is employed for the extraction and classification of characteristics [109]. The study emphasizes the importance of incorporating visual and acoustic data into surveillance for the detection of violent incidents. In order to develop this study, the authors needed to address the absence of public databases on campus violence, and had to meticulously design and create their own database of videos that deals with this topic. This experimental data creation process involved a combination of scenarios where role interpretation was performed, specifically simulating instances of violence on campuses, such as physical assaults, and routine activities of daily life, like walking and running. In total, the authors gathered a substantial dataset consisting of 12,448 frames of violence on campuses and an equivalent number of frames portraying daily life activities. As mentioned earlier, this investigation uses a **C3D** network to extract features and sort images taken from videos. The methodology used by the developed model involves the analysis of 16 frames of video images simultaneously, resulting in the extraction of vectors of characteristics of 4096 dimensions. As for the acoustic data, the sound was taken from the captured videos, using three different audio databases, which were combined with the video. To identify speech patterns related to violent incidents, **Mel-frequency cepstral coefficients (MFCCs)** are used as acoustic characteristics [110]. Classifiers have been created for each data type and to deal with conflicts in evidence by combining the results of activity recognition and emotion recognition in speech, the authors suggest an improved Dempster-Shafer algorithm [111]. In this study, five-fold cross validation was used, with the results indicating a remarkable effectiveness of the proposed method, achieving a accuracy of 97.00% when combining the video data with a specific emotional database. On the other hand, another contribution from this study is the progress provided by the improved fusion algorithm, which increases the accuracy by 10.79% relative to the existing fusion rule.

In the work produced by Waseem Ullah et al. [18], the integration of **Artificial Intelligence of Things (AIoT)** into video surveillance is addressed, where a two-flow neural network is proposed for the recognition of abnormalities in **Big Video Data (BVD)** of surveillance [112], which combines edge and cloud computing. The first stream is designed to detect instant anomalies on **IoT** devices with limited resources, while the second stream performs a thorough analysis of the anomaly. The researchers in this study faced a similar challenge to all studies in this area, where they had to select appropriate datasets to assess their proposed structure. Due to the limitations of existing datasets in terms of size and accuracy, the authors chose to use a dataset already mentioned (table 3), the UCF-Crime [81], for their performance assessment. This dataset is specifically designed to detect abnormal activities and includes 13 real-world scenarios, such as traffic accidents, thefts and assaults. The UCF-Crime [81] dataset consists of 1900

unedited surveillance videos, of which 950 depict abnormal events and 900 represent normal activities. To avoid bias in datasets, the authors also included the RWF-2000 [82] dataset in their evaluation. This dataset presents 2000 real-life surveillance videos obtained through YouTube, each with a duration of 5 seconds and presenting different resolutions at 30 frames per second. RWF-2000 [82] mainly involves abnormal behaviors, such as crowd activities, interactions between two people, and multiple people scenarios. The present investigation uses a dual-flow neural network to detect abnormalities [113], as mentioned above. The initial stream is specifically designed to quickly identify anomalies on **IoT** devices with limited resources, while the second stream is customized for in-depth anomaly analysis, especially suitable for cloud computing services. After an anomaly is detected, the edge device promptly notifies the relevant departments and transmits the abnormal frames to the cloud analysis center for a comprehensive evaluation in the subsequent phase. The cloud analysis center then uses a deep two-flow neural network for further analysis. This two-flow network is modeled by integrating space-time and optical flow characteristics into consecutive frames. These combined characteristics are then subjected to a two-way layer **LSTM** [114]. Consequently, the result is an advanced anomaly detection system capable of categorizing anomalies, including aggression and abuse, in extensive surveillance video data. The experimental results of the proposed model, tested on the datasets mentioned above, reveal a superior accuracy in the detection of abnormal events when compared with cutting-edge techniques, having achieved an increase of accuracy of 9.88% compared to the second best reference method. These numbers are promising and reveal a remarkable breakthrough and the feasibility of this architecture in intelligent video surveillance systems.

The research conducted by Arif Warsi et al. [115] focuses on the instantaneous identification of firearms using the **YOLO**-V3 algorithm [69]. This research highlights the importance of using **DL** techniques and real-time surveillance for the detection of firearms [116]. Regarding the dataset used, the researchers adopted a meticulous approach to the creation of the dataset for training and validation of their firearm detection system. To ensure a diversified and comprehensive dataset, the authors carefully selected their own set of images, specifically with firearms in different positions and orientations. This dataset has been carefully mixed with the well-known ImageNet [117] dataset, popular for its extensive collection of images labelled in various categories. The model developed focuses on the use of the **YOLO**-V3 algorithm for the instant detection of firearms [69]. **YOLO** is an object detection system that stands out for its high speed and accuracy. It divides an image into a grid and detects bounding boxes and class probability for each cell of the grid [69]. The **YOLO**-V3 model is trained in the dataset mentioned above to identify firearms in different configurations, taking into account variations in rotations, scales and shapes. The speed of the algorithm and its suitability for real-time applications make it a powerful tool for instant firearms

detection. Subsequently, this methodology is compared to the Faster RCNN approach, another known object detection model [70]. The results of this study prove the effectiveness of **YOLO** in the detection of firearms in different scenarios. In addition, it emphasizes the superior speed and accuracy displayed by the **YOLO**-V3 compared to the Faster RCNN.

The study by Ruimin Ke et al. [118] investigates the feasibility of using cloud computing in the context of video surveillance in parking lots [119]. The researchers have developed a system that incorporates a **Single Shot MultiBox Detector (SSD)** [120] along with various algorithms to detect the occupation of a parking lot. The **SSD**, being an object detection algorithm based on **DL**, demonstrates exceptional proficiency in real-time object recognition [121]. To improve detection performance in parking surveillance applications, the researchers used TensorFlow Lite and converted a pre-trained **SSD** model that was originally trained in the Pascal VOC dataset [121, 122]. Pascal VOC is a well-established dataset for object detection tasks, and its use in the training of the initial model is important to formulate a robust object-detection prototype. To further refine and adapt the model for parking surveillance, the researchers introduced a new dataset called MIO-TCD (Miovision traffic camera dataset) [123]. This dataset consists of 110,000 surveillance camera images and was initially designed for the training of traffic object detection models. MIO-TCD presents a variety of challenging traffic detection scenarios, including night scenes, truncated vehicles, low-resolution images, shadows and more [123]. This approach using **SSD** is combined with a technology to remove the background from the image, with edge computing at the base of this process. The system is specifically designed taking into account cloud computing, thus facilitating the detection of parking space occupation in real time. This is achieved through the combination of environmental sensors, cameras and **IoT** devices, which collectively improve the efficiency of parking management. Through three-month field tests, the results show encouraging results, with an average accuracy rate of 95.6% in various real-world scenarios. Consequently, this research serves as a testimony to the advantages offered by cloud computing in the field of surveillance [119]. It also emphasizes the importance of ensuring the efficiency and reliability of systems in the field of smart city applications.

Finally, the research conducted by I. P. Febin et al. [124] focuses on the identification of violence in surveillance videos, presenting a sequential technique that uses **Motion Boundary Scale-invariant Feature Transform (MoBSIFT)** and motion filtering [125]. The suggested technique ignores non-violent activities through filtering, thereby increasing accuracy and effectiveness in the detection of violence [126]. As in the studies discussed earlier, the lack of realistic data in action recognition datasets has been a significant obstacle to work in this area. The most commonly used datasets, such as KTH [127], Weizmann [128], and IXMAS [129], have limitations that include static backgrounds, absence of images with

obstructed elements, predefined time segmentation, and involvement of only one actor. Although these datasets are good references, they may not accurately reflect real-life scenarios, which calls for more realistic datasets. To deal with these limitations, the researchers used the Hockey and Movies datasets [130]. The Hockey dataset consists of 1000 ice hockey fight videos, distributed equally, i.e. 500 videos containing fights and 500 videos where there are no violent acts, with each video consisting of 50 frames. The Movies dataset includes 200 videos, with 100 depicting fights and the remaining 100 the opposite. Fight videos are extracted from action movies, while non-violent videos are obtained from other action recognition datasets. In this way, some of the above-mentioned issues are addressed, such as the presence of variability in resolution and illumination in the Movies dataset, addressing some the shortcomings of traditional datasets. A new method is presented in this study, which suggests a sequential technique to detect violence using **MoBSIFT** and motion filtration [125]. To reduce the processing of non-violent actions, videos are first subjected to a motion filtering algorithm based on time derivatives. Then only the filtered frames go through the feature extraction. The **MoBSIFT** descriptor is formed by the combination of the **Scale-invariant Feature Transform (SIFT)**, an optical flow characteristic histogram and a motion delimitation histogram [125]. Next, the **Bag of Visual Words (BoVW)** approach is used which tries to represent images efficiently by converting voluminous sets of high-dimensional local descriptors into vectors with fixed characteristics [131]. Finally, to classify the images are compared the **Support Vector Machine (SVM)** classification models, random forest and AdaBoost. The use of this innovative methodology exceeds current approaches, especially in its ability to handle camera movements and its reduced time complexity, having shown better results in the Movies dataset.

2.3.1 Critical analysis of the studies

The studies covered in this literary review provide valuable insights into the integration of intelligent surveillance systems, each with distinct approaches to addressing specific challenges. However, it is crucial to highlight some critical considerations regarding these researches.

Some of these surveys [28, 115], faced the common challenge of the lack of public datasets suitable for their fields of study. Thus, they had to develop experimental datasets. While creating new datasets is crucial to generating advances in this area, it can bring prejudices and constraints regarding real-world representativity. This limitation may affect the generalization and applicability of the proposed solutions in broader contexts. The size and diversity of datasets can also question the robustness and validity of the results obtained. Assessments carried out on larger and more varied datasets would be crucial for a more comprehensive understanding of the performance of the proposed models.

All of these articles are based on **ML** and **DL** techniques to achieve satisfactory results. Solutions are presented through the use of hybrid architectures, where they combine several models of this area to create a functional technology, taking advantage of the most positive features of each. However, it is important to note that training **DL** models often requires large datasets, as already mentioned, and significant computational resources. Adapting these solutions to resource-limited environments can be challenging and is a matter of great importance.

Focusing further on the implementation of these technologies in real environments, it can be observed that although these studies show encouraging results in controlled environments the possibility and effectiveness of these solutions in real world scenarios may be different. All these studies must be subjected to field tests to observe how the developed solutions behave with real data. Practical aspects must also be taken into account when implementing these intelligent surveillance systems, such as the maintenance, scalability and costs associated with the implementation of these systems, which must be thoroughly analysed.

A crucial element that remains inadequately addressed in most studies is the explainability of the models used. Although the emphasis has predominantly been on the performance of machine learning and deep learning methodologies, there has been insufficient focus on the decision-making processes of these models. This deficiency in transparency raises ethical dilemmas, particularly in high-stakes environments such as intelligent surveillance systems, where biased predictions could result in significant repercussions.

For example, the absence of explainability may lead to the inadvertent perpetuation or exacerbation of societal biases, potentially resulting in the disproportionate identification of certain racial or ethnic groups as more likely to engage in violent behavior. Such biases not only compromise the equity of surveillance systems but also pose serious ethical challenges, which could culminate in discrimination or wrongful profiling. Enhancing the interpretability and explainability of these models is essential for identifying and alleviating such biases, thereby fostering more just systems. This dissertation also aims to explore this area, focusing on the significance of explainability in intelligent surveillance models.

Table 4: Comparison between studies

Study	Creation of experimental datasets	Usage of pre-trained models	Presence of explainability analysis
[28]	Yes	No	No
[18]	No	Yes (EfficientNet)	No
[115]	Yes	Yes (YOLO-V3, Faster R-CNN)	No
[118]	No	Yes (MobileNet)	Yes
[124]	No	No	Yes

Although there are some adversities in the studies discussed, many positive aspects can be drawn from them. For example, [18] highlights the importance of integrating **AIoT** and edge computing technologies into surveillance systems. This approach optimizes local processing for **IoT** devices with limited resources, while performing more detailed analytics in the cloud. This methodology presents an efficient combination of technologies already used in the real world, making it feasible to adopt distributed surveillance systems with an architecture similar to the one presented. You can also observe the use of specific and already established algorithms in the **ML** area to meet the objectives of each investigation. For example, the use of the **YOLO**-V3 algorithm for instant firearm detection [69] highlights the need for fast and accurate algoritms in real-time surveillance scenarios.

In conclusion, this chapter provided an in-depth immersion in the essential concepts related to the current state of intelligent video surveillance, with a special emphasis on video surveillance and **AI**. Throughout this chapter, the fundamental features of video surveillance have been investigated, from the basic technologies and methods to the critical evaluation of research articles relevant to the scope of this dissertation.

Chapter 3

Materials and methods

This chapter discusses the materials and methods used in this work. Initially, the origin and characteristics of the datasets used are outlined, along with a preliminary analysis of this data, as described in section 3.1. Next, in section 3.2, the techniques adopted for data pre-processing are detailed, with the aim of constructing specific datasets for the case study. This stage includes preparing, cleaning and organizing the data, ensuring its suitability for subsequent analysis and modelling. Finally, section 3.3 mentions the technologies and tools used to develop the source code for this dissertation. This includes programming languages, libraries, development environments and other tools necessary for the practical implementation of the proposed methodologies.

3.1 Data exploration

In this section, the details of the exploitation of the "Real Life Violence Situations Dataset" [83] for the development of the violence and vandalism detection system using artificial intelligence are presented. In addition, a detailed justification of the choice of this particular dataset is provided.

The "Real Life Violence Situations Dataset" [83] is a comprehensive dataset that brings together a variety of videos captured in real situations of violence. This dataset contains a wide range of contexts, including street fights, fighting events, vandalism, among other violence scenarios, making it a valuable source of information for training and evaluating violence detection models.

The choice of the Real Life Violence Situations Dataset for this project was based on several important considerations:

- **Representativeness:** This dataset is highly representative of real situations of violence, covering a wide range of contexts and scenarios. This is crucial to ensure that the detection system developed is able to deal with a variety of real-world situations.

- **Availability and accessibility:** The dataset is readily available and publicly accessible, which facilitates access for research and development. In addition, it comes with detailed annotations, which simplifies the process of training and evaluating models.

- **Class balance:** The dataset presents a reasonable balance between the two classes of interest: non-violent videos and violent videos. The presence of a significant amount of examples in both classes is essential to ensure that the model is able to effectively learn the distinctive characteristics of each class.

- **Data quality:** Despite the variety of scenarios, the videos maintain a reasonable quality, allowing for an accurate analysis of the actions and events contained therein. This is crucial to ensure that the model is able to extract relevant features and perform accurate detection.

All these reasons led to this dataset being chosen to help create the models to be developed. Other alternatives were also explored, but this dataset was the one with the most satisfactory conditions.

To explore this dataset, an initial analysis was carried out to understand its structure and content. This analysis included a visual exploration of the videos, where a variety of videos were watched to gain a visual understanding of the situations of violence represented in the dataset. This allowed for greater familiarization with the different types of violence and scenarios present in the videos. A statistical analysis of the data was also carried out to understand characteristics such as the average length of the videos, their average resolution, among other relevant aspects.

The content of the chosen dataset is made up of 1000 videos of violence and 1000 videos of non-violence, all collected from YouTube videos. The videos of violence in this dataset present a variety of real-life situations of street fights in different environments and conditions. This diversity is crucial to ensure that the model is exposed to a wide range of violence contexts, allowing for accurate detection in real-world scenarios.



Figure 19: Frames extracted from a violent video in the dataset

In addition, the non-violence videos in this dataset are collected from various different human actions, such as sports, various human interactions, among others. The inclusion of these non-violence videos is important to ensure that the model is trained not only to identify violent situations, but also to properly distinguish between violent and non-violent actions. This diversity of human actions in the non-violent videos contributes to the robustness of the model, making it capable of discerning between different types of behavior in surveillance videos.



Figure 20: Frames extracted from a non-violent video in the dataset

By combining these different types of videos in the dataset, it is possible to create a robust and representative training environment that enables models to effectively identify violence in a variety of real-world situations. This multi-faceted approach contributes to the development of a reliable and efficient

violence detection system, with significant potential applications in intelligent surveillance and public safety.

After the visual exploration of the data, a comprehensive statistical analysis was conducted on the videos in the dataset, seeking to provide valuable insights into their fundamental characteristics. This analysis aimed to understand various aspects of the videos, including their duration, number of frames, resolution and frame rate (**Frames Per Second (FPS)**). By examining these statistics, it was possible to gain a detailed insight into the composition of violent and non-violent videos, allowing for a deeper understanding of the nuances present in the data.

Table 5: Violence and Nonviolence Video Statistics

Category	Violence videos	Non-violence videos	Total
Number of Videos	1000	1000	2000
Average Duration (s)	5.403	5.102	5.253
Average Number of Frames	159.8	127.6	143.7
Average Resolution (px)	(599, 481)	(418, 318)	(508, 400)
Average FPS	29.6	24.9	27.2

The dataset consists of a total of 2000 videos, divided equally between violent and non-violent videos. On average, the violence videos are 5,403 seconds long, with 159,8 frames on average. The average resolution of these videos is approximately 599 pixels wide by 481 pixels high, with an average frame rate of 29,6 **FPS**. On the other hand, non-violence videos have a slightly shorter average duration, at around 5,102 seconds, and an average number of frames of approximately 127,6. The average resolution of these videos is approximately 418 pixels wide by 318 pixels high, with an average **FPS** of 24,9.

When considering the data set as a whole, the average total length of the videos is approximately 5,253 seconds, with an average number of frames of around 143,7. The total average resolution is approximately 508 pixels wide by 400 pixels high, and the total average frame rate (**FPS**) is approximately 27,2.

Statistical analysis of the videos revealed a number of distinctive patterns and characteristics that are essential to understanding the nature of the data. It was possible to observe that the violence videos have a higher average number of frames and average resolution compared to the non-violence videos, suggesting greater visual richness and detail in the violence scenes. The difference in average frame rate between the two types of videos is also notable, reflecting possible variations in the speed and fluidity of the actions depicted. Taken together, these observations highlight the diversity and complexity of the violence and non-violence videos present in the dataset. This statistical analysis provides a solid basis for

the next stage of the process, which will involve implementing video processing techniques.

3.2 Data preparation

In preparing and processing the data, a series of critical steps are carried out to ensure that the videos are ready to be used effectively in training and evaluating vandalism and violence detection models.

Firstly, the videos are divided into groups of frames to capture a varied representation of the visual content of the videos, dividing the video into regular intervals to capture different moments over time. These moments are represented by the frames, which are extracted at equal intervals of time, determined by the total number of frames desired, which is 32 frames per video. This process works as follows:

- **Determining the total number of frames in the video:** First, it is necessary to obtain the total number of frames in the video. This gives an idea of the total duration of the video, to know exactly where to extract the frames.
- **Calculating the interval between frames:** Next, the desired interval between the frames to be extracted is calculated. This value is calculated by dividing the total number of frames in the video by the desired number of frames to be extracted. This gives the interval between frames to be chosen in the next step.
- **Choosing the frames:** With the total number of frames and the interval between frames, you can calculate at which specific moments to capture a frame. For example, if the video has 160 frames and 32 frames need to be extracted, the interval between frames will be 5 frames (160 frames / 32 frames = 5 frames).
- **Extracting the frames:** Once the specific moments when the frames need to be captured are determined, they are extracted from the video at those moments. This can be done by moving to the position of the desired frame in the video and saving the frame present at that position along with the rest of the extracted frames.

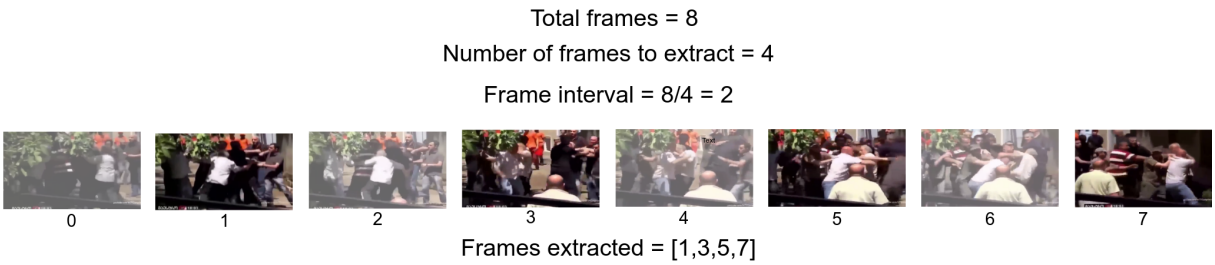


Figure 21: Process of choosing the frames to extract

The extracted frames are then resized. The resolution of the extracted frames is set to ensure that important details are captured properly, but without overloading the processing. The resolution chosen for the extracted frames was 128x128.

Next, the frames were normalized, which refers to the process of adjusting the pixel values of each frame to a specific range, usually between 0 and 1. This is done by dividing each pixel value by the maximum possible pixel intensity value (255 for each color channel in color images).

After this, the dataset is created, which involves compiling all the frames extracted from the violent and non-violent videos, along with their respective labels. For each group of frames, a label is associated indicating whether it belongs to the violence class or the non-violence class. This association is made manually, based on the category of the original video from which the frame was extracted.

Once the dataset has been compiled, the frame labels are converted into one-hot encoded vectors. This is done to represent the classes in binary form. Basically, each position in the vector corresponds to a class. The position corresponding to the current class is assigned the value 1, while all other positions are assigned the value 0. This helps the model to understand and work with the classes more efficiently during training, allowing it to calculate the probability of each class independently, making it easier to interpret the results and make decisions.

In conclusion, after all the pre-processing and preparation of the data, a dataset is obtained ready to build and train violence detection models. The resulting dataset will consist of frames extracted from the original videos, each representing a specific moment in time. Each frame will be associated with a label indicating whether it belongs to the violence or non-violence class, represented in one-hot encoded format. In addition, the path of the original video associated with each frame will be kept, ensuring traceability and data integrity.

3.3 Technologies

Throughout this project, the source code used in the data exploration and processing stages, as well as in the design, training and evaluation of ML models, was developed using the Python programming language (version 3.10). During development, some libraries were included, such as *pandas*, *matplotlib*, *tensorflow*, *keras*, *numpy*, *seaborn*, *scipy*, *sklearn*, *opencv* and *ipython*.

The environment used to create, adjust and run the models featured a robust hardware configuration, including a powerful Nvidia Quadro RTX 8000 graphics card with 48GB of RAM, together with an Intel(R) Core(TM) i9-10900K CPU with 3.70GHz and a total of 64GB of RAM.

Chapter 4

Experiments

This chapter details all the key stages and considerations related to the experiments conducted as part of this study. Initially, in the section 4.1, the process of training the models is described. Subsequently, the experimental scenarios considered are outlined, presenting the process of implementing the candidate models, exploring the design and development of each one in the section 4.2. Later, in section 4.3, the evaluation metrics used to measure the performance and effectiveness of the proposed models are discussed. The importance of these metrics in the critical and comparative analysis of the results obtained is highlighted.

Finally, the pipeline adopted for running the experiments and storing the results is presented in section 4.4. This pipeline is essential for the efficient organization of the experimental process, guaranteeing the integrity and traceability of the data and results obtained. A detailed understanding of each of these steps is crucial for a comprehensive and well-founded analysis of the approaches adopted and the results achieved throughout this study.

4.1 Training process

Firstly, it is important to note that each frame of the videos has dimensions of 128x128x3, which results in a considerably large input for the models to be developed. This significant size contributes to the excessive consumption of computing resources during training. Therefore, part of the challenge in this training process is to find effective strategies to reduce the complexity of the data without compromising the information it contains.

Given the voluminous and intricate nature of the videos, the dataset was divided into batches, a strategy that proved indispensable for ensuring the computational efficiency and scalability of the training process. With a total of 2000 videos available, each batch was made up of a set of 32 videos, each containing 32 frames. This meant that there were 62 complete batches (i.e. batches containing all 32 videos) and,

consequently, one partial batch that was not used.

To train the models from the batches, it was essential to replicate the behaviour of the fit method. To do this, cycles were implemented that iterated over the epochs, allowing the model to learn progressively from the training data. Within these cycles, the train on batch approach was adopted, which made it possible to feed the model with each batch of data individually, adjusting its parameters according to the information contained in each set of examples. It is important to emphasise that, when creating each batch, it was ensured that there was a balanced amount of videos representing violence and non-violence. This approach ensures that the model is exposed to a variety of representations.

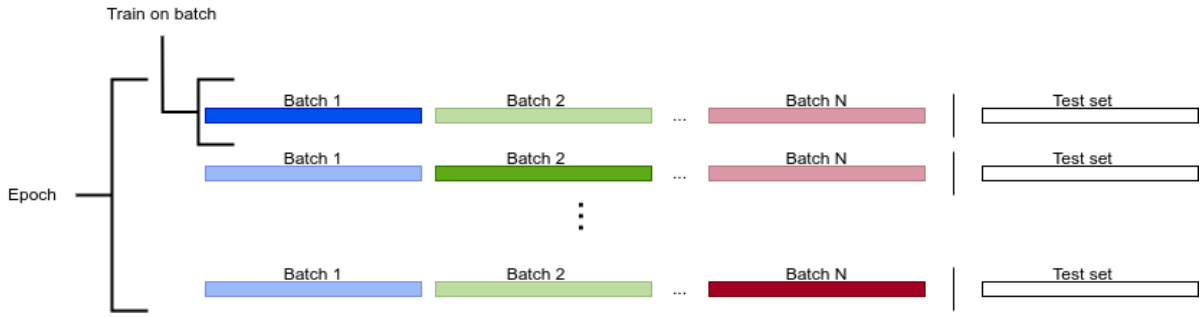


Figure 22: Visual representation of the training process

A crucial strategy adopted during training was the introduction of a shuffle within each batch. This was done to mitigate any tendency for the model to learn from specific patterns of data presentation. Randomising the order of the image sets within each batch helps the model to learn in a more complete and generalised way.

In addition, randomising the order of the batches was implemented throughout the epochs. This practice adds an additional layer of variety to the training process, allowing the model to be trained in different contexts and data sequences. This not only reduces the likelihood of the model overfitting specific batch patterns, but also promotes more robust and adaptive learning.

During the epochs, the same set of test batches was used, consisting of 7 test batches with a total of 224 videos, ensuring a consistent assessment of the model's performance throughout training. This practice makes it possible to monitor the model's progress in relation to the test data, offering valuable insights into its ability to generalise and its performance on unseen data. By maintaining consistency in the set of test batches across epochs, a fair and accurate assessment of the model at different stages of training is ensured, contributing to a more informed and refined iteration of the model development process.

To ensure the robustness and adequate generalisation of the models during training, a proactive ap-

proach was adopted to deal with the challenge of overfitting. Recognising the tendency of models to become overly adapted to training data, to the detriment of their ability to generalise to new data, data augmentation techniques were incorporated into the training pipeline.

```
datagen = ImageDataGenerator(
    rotation_range=20, # Rotate the image randomly in the range -20 to +20
        degrees
    width_shift_range=0.1, # Random horizontal translation of up to 10% of
        the total width
    height_shift_range=0.1, # Random vertical translation of up to 10% of
        the total height
    shear_range=0.2, # Applies shear of up to 20 degrees
    zoom_range=0.2, # Random zoom of up to 20%
    horizontal_flip=True, # Random horizontal mirroring
    fill_mode='nearest' # Fill pixels outside the image boundaries
)
```

Listing 4.1: Code excerpt for data augmentation.

These techniques play a crucial role in diversifying and enriching the training data set by introducing variations and disturbances in the video frames. By applying transformations such as random rotations, horizontal and vertical inversions, zooms and offsets, it is possible to generate a wider range of training examples from the original data.

Given that training the models requires a considerable amount of time, a total of 70 epochs was set for each model. This number was strategically selected to balance the need to allow the models to learn adequately from the available data with the feasibility of training multiple models within a reasonable time-frame. By setting 70 epochs, it was possible to efficiently evaluate the performance of all candidate models, ensuring a thorough and informed comparative analysis of the different approaches and techniques for detecting violence and vandalism in intelligent surveillance systems.

4.2 Candidate models

In the process of developing the candidate models, various artificial intelligence approaches and techniques were explored for detecting violence and vandalism in intelligent surveillance systems. Each candidate model was designed based on different algorithms and architectures, exploring a variety of supervised learning techniques.

For each candidate model, criteria were defined for selecting features and parameters, taking into account the nature of the data available and the specific objectives of detecting violence and vandalism. The implementation of each model was carried out using widely recognized and tested machine learning libraries and tools.

For each candidate model, several versions were developed, each with specific adjustments to the parameters and characteristics selected. These multiple iterations were key to exploring different configurations and refinements, seeking to identify the most effective combination of elements for detecting violence and vandalism. Subsequently, these versions were meticulously compared and evaluated based on predictive performance metrics, to determine the best definition of the model in terms of its ability to accurately and efficiently identify events of interest in surveillance images.

4.2.1 Conv3D

The Conv3D approach is a powerful technique used for video analysis and processing, based on the application of three-dimensional convolutions to volumes of data representing temporal sequences of images. This technique is a natural extension of conventional **CNN**, which are widely used in computer vision tasks on static images.

By applying three-dimensional convolutions, the Conv3D approach allows the model to learn spatial and temporal characteristics directly from the video data, capturing complex patterns of movement and change over time. This is essential for tasks such as action detection, human activity recognition and, in this case, detecting violence and vandalism in surveillance videos.

In addition to three-dimensional convolutions, Conv3D networks, like the models that will be discussed later, generally also include pooling and dropout layers to improve the model's performance and generalization capacity.

Pooling layers are used to reduce the dimensionality of data volumes along spatial and temporal dimensions. This is achieved by selecting values that are representative of specific regions of the input volume. For example, in a three-dimensional pooling layer, a max or average pooling operation can be applied to three-dimensional cubes of the input volume. This helps to reduce the number of parameters in the model, making it more computationally efficient and less susceptible to overfitting.

On the other hand, dropout layers are used as a regularization technique to avoid overfitting during model training. During this process, a random fraction of the neurons in a layer are temporarily deactivated, meaning that their contributions are temporarily removed from the calculation of forward propagation and the adjustment of weights during backpropagation. This forces the network to learn more robust and

distributed representations of the data, thus reducing the likelihood of overfitting.

Therefore, by adding pooling and dropout layers to the architectures of the models developed, they become more capable of learning discriminative and robust representations of video sequences, thus improving their ability to detect patterns of violence and vandalism in surveillance videos.

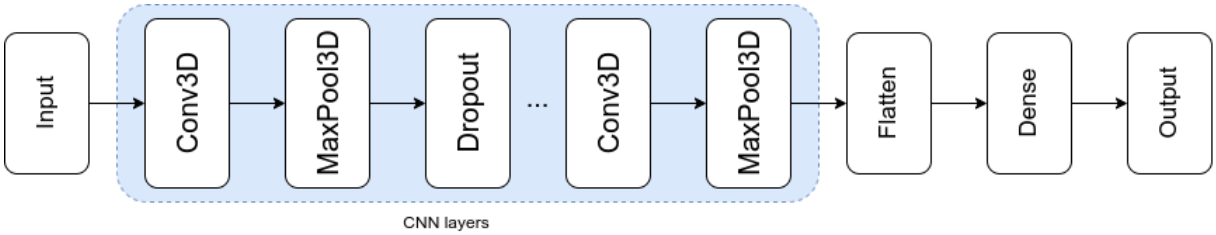


Figure 23: Example of a Conv3D model architecture

4.2.2 ConvLSTM

ConvLSTM is a hybrid approach that takes advantage of **CNN** and **LSTM** networks, designed specifically to deal with data sequences, such as videos, that have both spatial and temporal characteristics. In this approach, ConvLSTM2D layers are used to learn spatio-temporal patterns directly from video data.

The ConvLSTM2D layers combine the principles of the three-dimensional convolutions of **CNN** with the memory and information flow control units of **LSTM** networks. This allows the model to learn hierarchical and dynamic representations of temporal image sequences, capturing both the spatial structure and temporal relationships between video frames. By applying ConvLSTM2D layers, the model is able to process each frame of the video adaptively, taking into account not only the spatial characteristics of the scene, but also the temporal information throughout the sequence of frames. This is essential for video analysis tasks such as detection of violence and vandalism in surveillance videos.

In addition to using dropout throughout the network, this architecture also makes use of recurrent dropout. This is a variation of dropout designed specifically for recurring networks. Instead of randomly deactivating units at each time step, recurrent dropout consistently deactivates the same recurring units at all time steps, ensuring that temporal dependencies are learned consistently over time. This helps to regularize the network and avoid overfitting, thus improving the model's ability to generalize to test data.

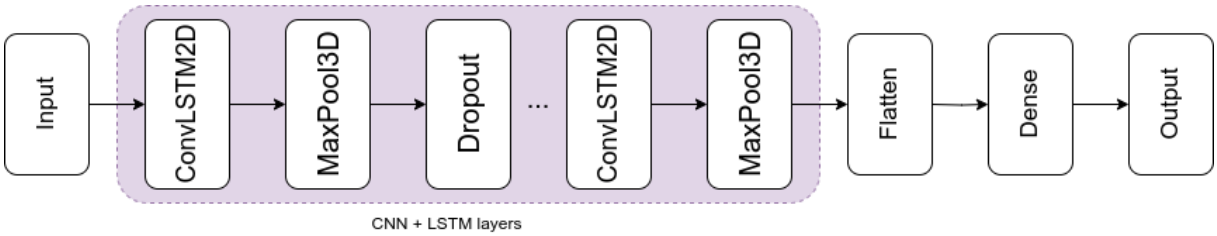


Figure 24: Example of a ConvLSTM model architecture

4.2.3 LRCN

The **Long-term Recurrent Convolutional Network (LRCN)** approach is a hybrid technique that combines elements of **CNN** and **RNN**, specifically **LSTM**, for video analysis. In this approach, Conv2D layers are combined with a TimeDistributed layer and then **LSTM** layers are employed to model the resulting time sequence.

At the start of the network, the Conv2D layers are responsible for extracting spatial characteristics from the individual video frames. These layers apply convolution operations to detect relevant visual patterns in each frame, such as edges, textures and shapes, generating feature maps that represent the visual information contained in the video. The Conv2D layers are wrapped in a temporal envelopment layer (TimeDistributed) which is used to apply an operation to each time step of the input data, maintaining the temporal information. This means that subsequent layers will be applied to each frame of the video independently. Finally, **LSTM** layers are used to model the temporal sequence of features extracted by the convolutional feature maps. The **LSTM** layers are capable of capturing long-term temporal dependencies, allowing the model to learn the temporal dynamics of the videos and identify complex patterns over time.

One of the main advantages of the **LRCN** approach is its ability to capture both spatial and temporal information in an integrated way, taking advantage of **CNN** ability to extract visual features and **LSTM** ability to model temporal sequences. This can lead to more robust and effective models in video classification.

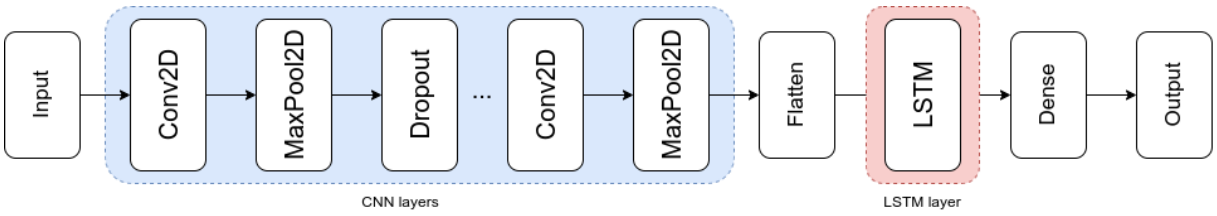


Figure 25: Example of a **LRCN** model architecture

4.2.4 CNN-BiLSTM

The **Convolutional-Bidirectional Long Short-Term Memory (CNN-BiLSTM)** approach is a strategy that, like **LRCN**, combines elements of **CNN** and **RNN**, but makes use of a **Bidirectional Long Short-Term Memory (BiLSTM)** architecture. In this approach, the Conv2D layers are followed by a temporal envelopment layer (TimeDistributed), and then **BiLSTM** layers are employed to model the resulting temporal sequence.

As in the previous model, the Conv2D layers are initially used to extract spatial features from the individual video frames. These layers apply convolution operations to identify important visual patterns in each frame, such as edges, textures and shapes, generating feature maps that represent the visual information contained in the video. As in the previous model, the Conv2D layers are wrapped in a TimeDistributed layer, used to apply the subsequent operations to each individual frame. This allows the operations to be repeated for each frame, keeping the temporal information intact.

BiLSTM layers are then used to model the temporal sequence of features extracted by the convolutional layers. **BiLSTM** is a variation of **LSTM** that processes the input sequence in two directions: from past to future and from future to past. This allows the model to capture contextual information from both sides of each time instant, which can be especially useful for video analysis tasks, where temporal contextualization is crucial.

This approach has advantages over the previous one due to its ability to efficiently capture spatial and temporal information in an integrated way, taking advantage of **CNN**s' ability to extract visual features and **BiLSTM**s' ability to model temporal sequences in a bidirectional way.

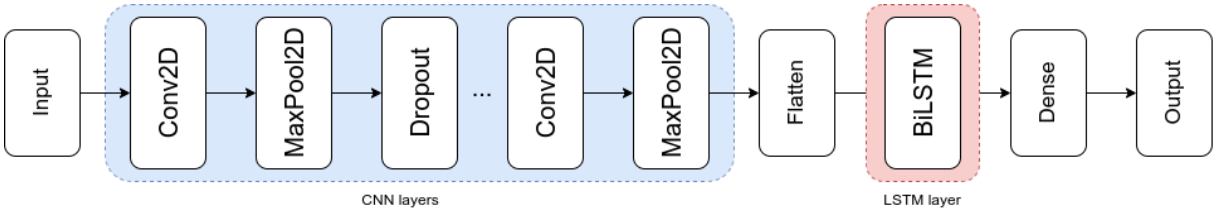


Figure 26: Example of a **CNN-BiLSTM** model architecture

4.2.5 MobileNet-BiLSTM

To compare the models trained from scratch with a pre-trained model, the MobileNet-BiLSTM architecture was created. This strategy, like the **CNN-BiLSTM** approach, makes use of the **BiLSTM** architecture. MobileNet [132] is a model initially trained on the ImageNet dataset [117], an extensive database that is

well known in the computer vision community. ImageNet contains millions of images labeled in thousands of categories, covering a wide range of real-world objects and scenes. This prior training, also known as transfer learning, allows MobileNet to learn to identify general visual characteristics that are useful for a wide variety of image recognition tasks.

In this model, the MobileNet layers pre-trained on the ImageNet dataset are followed by a TimeDistributed layer, and then **BiLSTM** layers are used to model the resulting time sequence. To make the most of transfer learning, it is common to make some layers of the pre-trained model trainable. In the case of MobileNet-BiLSTM, this means that after loading the pre-trained MobileNet into ImageNet, some of its layers are unfrozen, allowing their weights to be updated during training with the data specific to the new task.

Initially, the MobileNet layers are used to extract spatial features from the individual video frames. MobileNet, being an efficient and lightweight **CNN**, applies convolution operations to identify important visual patterns in each frame, such as edges, textures and shapes, generating feature maps that represent the visual information contained in the video. As with the previous model, the MobileNet layers are wrapped in a TimeDistributed layer, used to apply subsequent operations to each frame individually. This allows operations to be repeated for each frame, keeping the temporal information intact.

The **BiLSTM** layers are then used to model the temporal sequence of the features extracted by the convolutional layers. This approach has advantages over the previous one due to its ability to efficiently capture spatial and temporal information in an integrated manner, taking advantage of MobileNet's ability to extract visual features and **BiLSTM**'s ability to model temporal sequences in a bidirectional manner. In addition, the use of MobileNet pre-trained on ImageNet makes it possible to transfer knowledge from a vast image database, improving the efficiency and accuracy of the model in video analysis tasks.

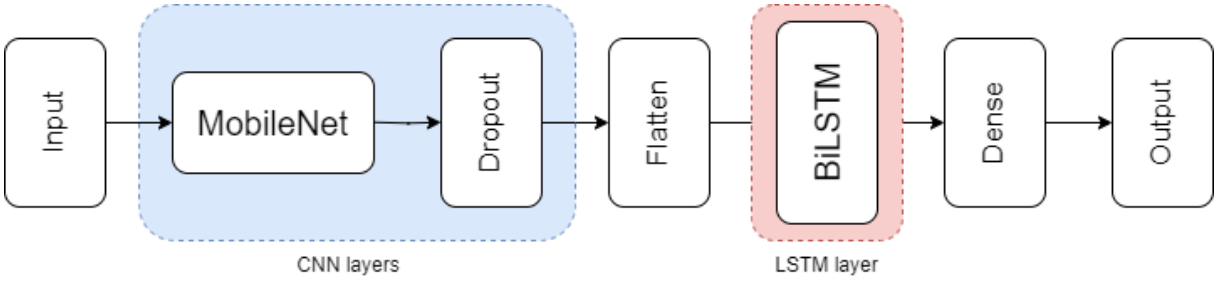


Figure 27: Example of a MobileNet-BiLSTM model architecture

4.3 Evaluation metrics

After implementing the models and optimizing their hyperparameters, which involved carrying out several experiments, it is essential to analyze the results obtained and evaluate the most promising candidate models. It is necessary to determine which are the best candidate models among all those tested in the experiments, thus resulting in a selection process. As this is a video classification problem, metrics such as accuracy and loss are the main ones to be evaluated [133].

Accuracy is a fundamental metric that measures the proportion of correct predictions in relation to the total number of predictions made by the model. In the case of detecting violence and vandalism in videos, accuracy indicates the model's ability to correctly classify frames as containing or not containing violent or vandalism activities [133].

Loss, on the other hand, measures the model's error in its predictions. In the context of detecting violence and vandalism in videos, loss quantifies how much the model's predictions deviate from the true classes of the frames. In particular, the Binary Cross-Entropy Loss function is used, which is suitable for binary classification problems, as is the case here. A lower loss indicates a more accurate model in its predictions [133].

Below are the formulas for calculating each of these metrics:

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{4.1}$$

In the accuracy formula, tp represents true positives, fp false positives, tn true negatives and fn false negatives.

$$Loss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{4.2}$$

In the Binary Cross-Entropy Loss formula, n represents the total number of samples, y_i denotes the true label of the i-th sample (either 0 or 1), and $\hat{y}_i$ indicates the probability predicted by the model that the i-th sample belongs to the positive class (violence).

4.4 Pipeline followed

This chapter details the complete pipeline followed to develop **DL** models to detect violence and vandalism in videos. The creation of an efficient and well-structured pipeline is essential to guarantee the quality and robustness of the models, as well as allowing the process to be easily replicated.

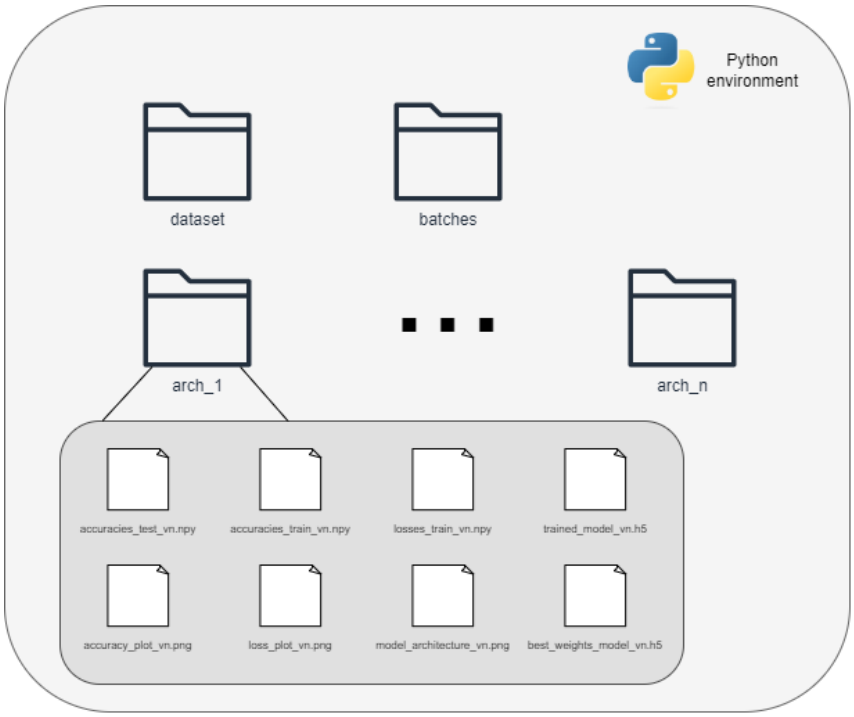


Figure 28: Pipeline used throughout the study

1. **Environment configuration:** Initially, a Python environment is created with all the libraries needed for the job, including libraries for video processing, data manipulation, building and training deep learning models.

2. **Data preparation:** Based on the chosen dataset, the data was pre-processed and organized into batches. Each video was segmented into 128x128 frames, collecting 32 frames per video. The data was then stored in batches of features and corresponding labels, organized in *.npy* files, which contain the numpy arrays with the video frames and their labels, respectively.

3. **Model creation and training:** The model creation and training process were executed within a single file named *model.py*. This file outlines the model's architecture and the parameters used for training. It is essential to make adjustments to this file for each unique architecture and for every different iteration of the same architecture. Throughout the training process, the loss and accuracy metrics for both the training and test sets at each epoch were saved in *.npy* files. The top-performing model (with the highest accuracy on the test set) was stored in *.h5* files, along with the model's state at the conclusion of the final executed epoch. All of this data was stored in a directory named after the architecture, which also included a visualization of the model to be trained. This directory housed all versions associated with the specific architecture.

This structure provides a clear and detailed overview of the process followed in the development of

DL models for detecting violence and vandalism in videos, highlighting the importance of each stage and the care taken to keep this whole process organized.

Chapter 5

Results and Discussion

The following chapter highlights the results achieved after conducting a series of experiments, meticulously configured as outlined in the previous chapter. These results are the fruit of a validation process that included the application of evaluation metrics to observe the behaviour of the models developed.

As previously mentioned, several experimental scenarios were outlined, each of which involved running numerous iterations for four different deep learning models: Conv3D, ConvLSTM, **LRCN** and **CNN-BiLSTM**. This chapter of results and discussion is structured so as to initially highlight the results provided by each type of model, in the 5.1 section. Within this first section, the results of the experiments carried out on each model considered are presented, totalling four subsections corresponding to each model.

In the 5.3 section, an explainability analysis of the models is carried out using the **Grad-CAM** method. This analysis not only validates the effectiveness of the models, but also ensures greater transparency and interpretability in the results obtained.

Finally, the 5.2 section presents a comparative analysis of the best models for each candidate model. In this section, graphs representing the performance of the various models are also presented in order to identify the best model developed, offering a clear visualisation of the results obtained for a better understanding and interpretation of them.

5.1 Model results

This section presents the results obtained from the development and evaluation of **DL** models for detecting violence and vandalism in surveillance videos. Analysing the results includes evaluating the performance of the models in terms of accuracy and loss.

Firstly, the configuration of the experiments and the parameters used to train the three best models for each architecture are presented. Next, the results obtained for each architecture are described, verifying the performance of the best model and identifying the factors that contributed to the best results. Finally,

the implications of the results in the context of intelligent surveillance systems are discussed.

For each parameter, a range of values was established, from which the best models were selected based on their accuracy. The variables that were changed between versions of the same models include convolutional filters, kernel size, pooling layers, dropout layers and **LSTM** units in **LSTM**-based architectures. Convolutional filters and kernel size influence the model's ability to extract features from input data, while pooling layers decrease dimensionality and computational load. The dropout layers help prevent overfitting by randomly deactivating a fraction of the neurons during training, and the **LSTM** units are responsible for managing temporal dependencies in the data.

Some configurations were kept throughout the tests because they showed more satisfactory results than the alternatives. These configurations include the use of MaxPooling layers, ReLU activation functions and Softmax output layers, which are essential components for effective performance in video classification tasks.

5.1.1 Conv3D

The results presented in the subsequent table, show the performance of different versions of the Conv3D model, varying in terms of convolution filters, convolution kernel sizes, pooling and dropout layers. The main metric analysed was accuracy, which indicates the proportion of correct predictions made by the model in relation to the total number of predictions.

Table 6: Results from the Conv3D model

Version	Conv. Filters	Conv. Kernel Sizes	Pooling Layers	Dropout Layers	Accuracy
#8	32, 64, 64, 128	All (3, 3, 3)	(1, 2, 2), (1, 2, 2), (1, 4, 4) and (1, 4, 4)	All 0.1	91.071%
#1	32, 64, 128, 128	All (3, 3, 3)	(1, 2, 2), (1, 2, 2), (1, 4, 4) and (1, 4, 4)	All 0.2	90.625%
#3	32, 64, 128, 128	All (3, 3, 3)	(1, 2, 2), (1, 2, 2), (1, 4, 4) and (1, 4, 4)	All 0.1	89.732%

The version #8 demonstrated the highest accuracy, reaching 91.071%. The configuration of this model includes convolution filters of 32, 64, 64, and 128, all with kernel sizes (3, 3, 3). The pooling layers were applied with sizes (1, 2, 2), (1, 2, 2), (1, 4, 4), and (1, 4, 4), and all dropout layers were set with a rate of

0.1. This outcome suggests that a lower dropout value may have contributed to a better generalization of the model.

In all versions, it is important to emphasize that pooling was applied only in the spatial dimensions of the videos (with pooling window sizes of (1, x, x)), while the temporal dimension was preserved. This is crucial when working with videos, as it allows the model to maintain the integrity of temporal sequences while reducing spatial dimensionality, improving computational efficiency, and avoiding the loss of critical temporal information.

Increased pooling in the final layers seems to have favoured better generalisation and performance of the model, as evidenced by the three best versions having this pooling feature. This provides a better abstraction of the important features of the videos, reducing the spatial dimensionality more aggressively and consequently helping the model to focus on the most relevant features.

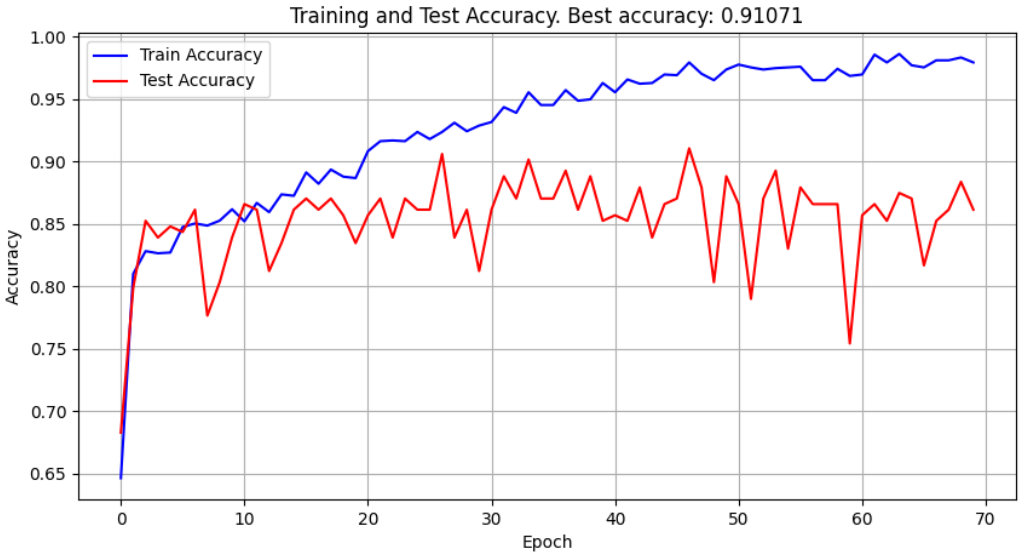


Figure 29: Training and test accuracy of the best Conv3D model

Upon analyzing the accuracy graph, it is evident that the test accuracy exhibits a variation above the average. This variation can be better understood by observing the loss graph (figure 30), which reveals an increasing behavior in the test loss. Such behavior indicates that the model may not be able to generalize well, and this instability in accuracy reflects the model's difficulties in maintaining consistent performance when evaluated with new data.



Figure 30: Training and test loss of the best Conv3D model

Despite the good accuracies achieved by the models, the test loss yields results below expectations. It is observed that the test loss is increasing in relation to the training loss, indicating that the models may be facing overfitting issues. This implies that although the model performs well on the training dataset, it struggles to generalize to unseen data during training. This difference suggests that the model is memorizing patterns from the training set instead of learning generalizable features, which is an important aspect to be improved in order to ensure robust performance in real-world situations of violence and vandalism detection.

Although the Conv3D model has shown good results in terms of accuracy, it may not be the most suitable for all situations of detecting violence and vandalism in surveillance videos. The architecture appears to have difficulties in capturing the features required for accurate detection. Therefore, other approaches based on hybrid architectures may provide a more robust and integrated understanding, resulting in potentially superior performance for the task of detecting events in videos.

5.1.2 ConvLSTM

The ConvLSTM architecture has shown promising results in identifying violence and vandalism in surveillance videos. The table below displays the different versions of the model along with their respective accuracies.

Table 7: Results from the ConvLSTM model

Version	Conv. Filters	Conv. Kernel Sizes	Pooling Layers	Dropout Layers	Accuracy
#3	16, 20, 24, 28	All (3, 3)	All (1, 2, 2)	All 0.1	92.857%
#4	20, 24, 28, 32	All (3, 3)	All (1, 2, 2)	All 0.1	91.964%
#5	8, 16, 32, 32	All (3, 3)	All (1, 2, 2)	All 0.1	91.071%

The ConvLSTM model results show different versions with variations in convolutional filter settings, kernel sizes, pooling layers and dropout layers. Version #3 obtained the highest accuracy with 92.857%, followed by version #4 with 91.964% and version #5 with 91.071%. All the versions show very close results, suggesting consistency in the model's ability to generalise to new data. It can be seen that the major difference between the various versions is in the number of filters along the ConvLSTM2D layers.

Compared to the models of the other architectures, the ConvLSTM models have a smaller number of filters in their convolutional layers. The limitation in the number of filters is due to the fact that ConvLSTM2D layers require more computing resources compared to other models. This is because ConvLSTM layers combine convolution operations with the memory and information flow control operations of LSTM networks. As a result, increasing the number of filters can significantly increase the computational load, making model training and inference more time-consuming and requiring more hardware resources. It was therefore necessary to find a balance between the complexity of the model and the resources available to ensure efficient training and good generalisation to new data.

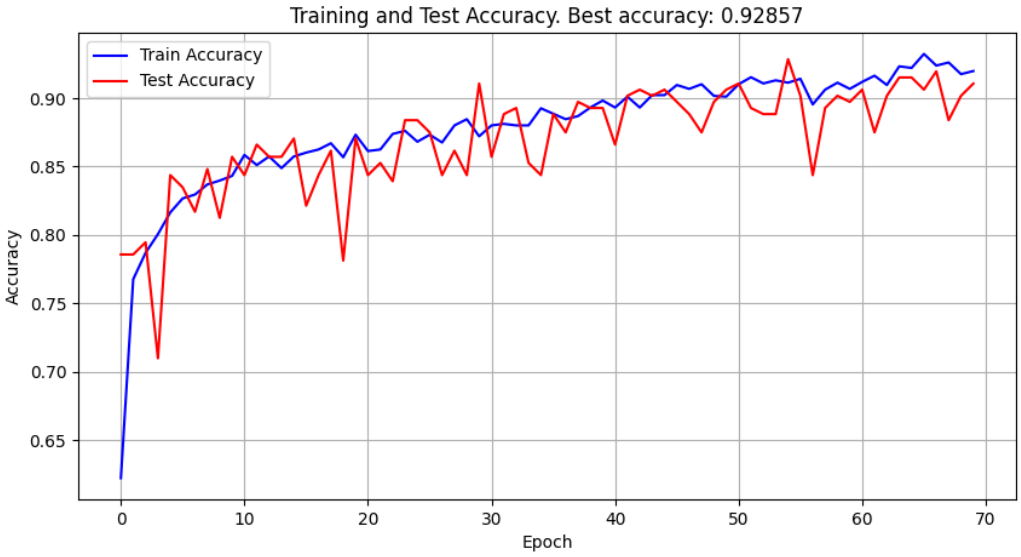


Figure 31: Training and test accuracy of the best ConvLSTM model

The high and consistent accuracy of the ConvLSTM model throughout the training epochs reflects the effectiveness of the ConvLSTM layers in capturing both the spatial and temporal characteristics of surveillance videos. The model's ability to maintain high accuracy during training is crucial to ensure its capability in accurately identifying violence and vandalism events in various scenarios.



Figure 32: Training and test loss of the best ConvLSTM model

The results of the ConvLSTM model demonstrate that the test loss is within expected and presents a balanced behavior, reflecting the model's ability to generalize well on the test data. The stability of the test

loss throughout the training epochs suggests that the model is effectively learning, avoiding overfitting, and adapting well to the training data. This stability is crucial to ensure that the model can be confidently applied in real-world situations.

In summary, the results of the ConvLSTM model indicate a solid performance and effective generalization capability, making it a promising option for violence and vandalism detection in surveillance videos. The stability and consistency of the model throughout the training demonstrate its reliability and ability to provide accurate and consistent results in different application scenarios.

5.1.3 LRCN

The results displayed in table 8 illustrate the performance of various iterations of the LRCN model, which integrates components of **CNN** and **LSTM** for video analysis.

Table 8: Results from the **LRCN** model

Version	Conv. Filters	Conv. Kernel Sizes	Pooling Layers	Dropout Layers	**LSTM** units	Accuracy
#6	32, 64, 128, 128	All (3, 3)	All (2, 2)	All 0.1	128	92.857%
#3	32, 64, 128, 128	All (3, 3)	(2, 2), (2, 2), (4, 4) and (4, 4)	All 0.1	64	92.441%
#13	32, 64, 64, 128	All (3, 3)	All (2, 2)	All 0.2	128	91.964%

The most optimal version, version #6, achieved an accuracy of 92.857%. The configuration of this model includes convolutional filters of sizes 32, 64, 128, and 128, all with kernel sizes of (3, 3). The pooling layers were uniformly applied with sizes (2, 2), and all dropout layers were set with a rate of 0.1. Additionally, version #6 utilizes 128 **LSTM** units. This combination appears to have been effective, resulting in a good balance between spatial and temporal feature extraction.

One important observation about the **LRCN** model is that, unlike previous models, pooling is only possible in two dimensions. This is because the Conv2D layers operate on each frame individually, applying MaxPooling only on the spatial dimensions (height and width) of each image, and not on the temporal dimension. This approach is suitable for capturing the spatial features of the frames before moving on to temporal modeling through the **LSTM** layers.

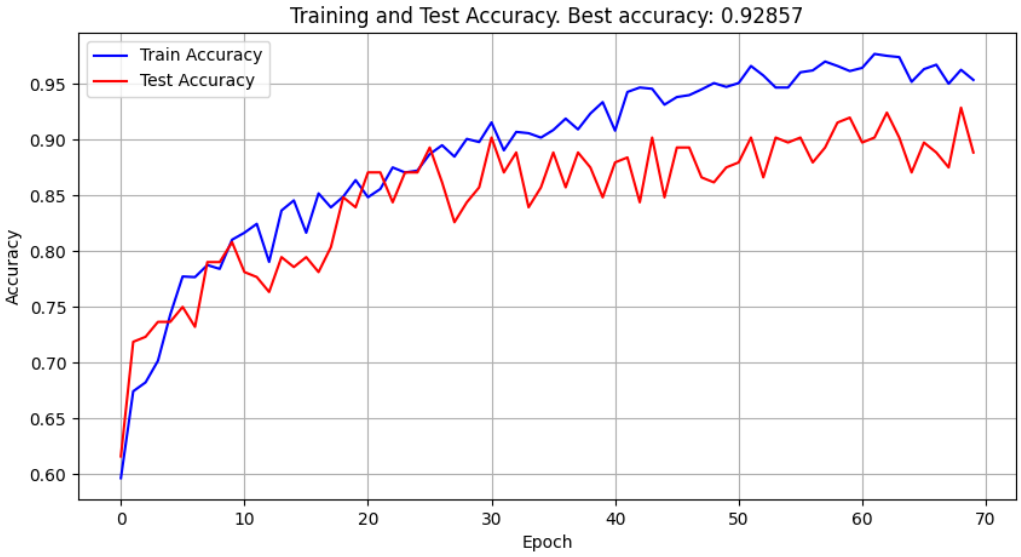


Figure 33: Training and test accuracy of the best **LRCN** model

Upon analyzing the accuracy graph, it can be observed that the test accuracy shows minimal variation, indicating a consistent performance. This stability can be attributed to the proper balance between regularization techniques (such as dropout) and the ability of **LSTM** to capture long-term temporal dependencies.



Figure 34: Training and test loss of the best **LRCN** model

The test loss appears to be stable with a more balanced and normal training. This suggests that **LRCN** models are capable of achieving good generalization, maintaining stable performance on unseen data during training.

The results obtained with **LRCN** models are quite promising for the task of violence and vandalism detection in surveillance videos. The 92.857% accuracy indicates that the **LRCN** architecture is effective in capturing both spatial and temporal characteristics of videos, using a combination of convolutional (Conv2D) and **LSTM** layers.

5.1.4 CNN-BiLSTM

The outcomes achieved using the CNN-BiLSTM model demonstrate outstanding performance in detecting violence and vandalism in surveillance videos. The table provided below illustrates the various versions of the model along with their corresponding accuracies.

Table 9: Results from the **CNN-BiLSTM** model

Version	Conv. Filters	Conv. Kernel Sizes	Pooling Layers	Dropout Layers	**LSTM** units	Accuracy
#9	32, 64, 128, 128	All (3, 3)	All (2, 2)	All 0.1	64	93.750%
#1	32, 64, 128, 128	All (3, 3)	All (2, 2)	All 0.2	64	92.411%
#5	32, 64, 128, 128	All (3, 3)	(2, 2), (2, 2), (4, 4) and (4, 4)	All 0.1	128	92.411%

The results indicate that model version #9 achieved the highest accuracy of 93.750%, standing out as the most effective among those tested. All the best versions utilized a consistent configuration of convolutional filters and kernel sizes, with variations in pooling layers, dropout, and **LSTM** units. Specifically, versions with larger pooling in the final layers (such as version #5) did not show significant improvements compared to the configuration with similar pooling throughout the network (versions #9 and #1).

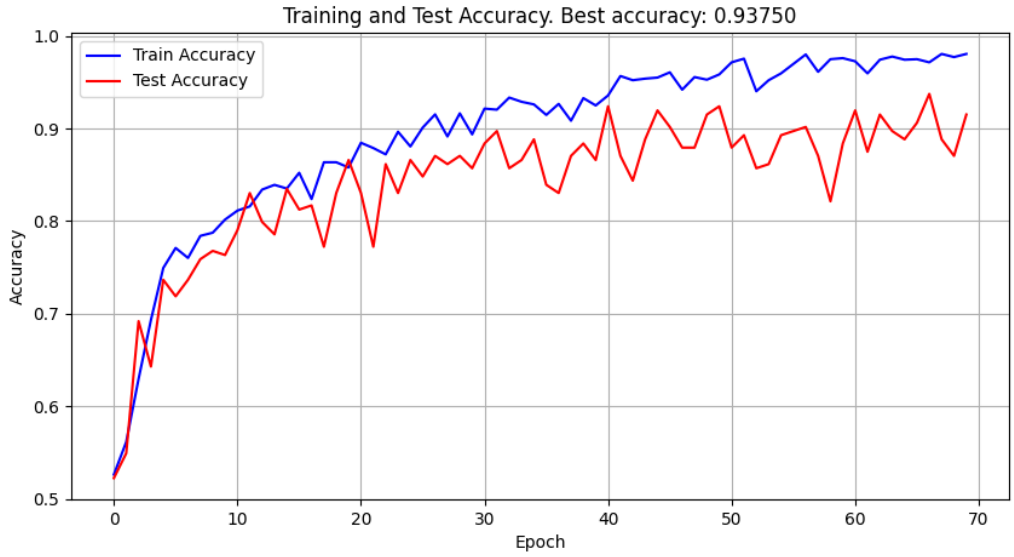


Figure 35: Training and test accuracy of the best **CNN-BiLSTM** model

The test accuracy variation is minimal, indicating a robust and consistent performance of the model across different executions and test data. This consistency in accuracy suggests that the model is reliable and can be expected to provide stable results in different real-world application scenarios. The high and consistent accuracy throughout the training epochs reflects the effectiveness of the convolutional layers and **BiLSTM** layers in capturing both the spatial and temporal features of surveillance videos.



Figure 36: Training and test loss of the best **CNN-BiLSTM** model

Maintaining stable test loss levels is of utmost importance as it indicates that the model is not merely

memorizing the training data, but rather learning generalizable representations that can be applied to new data. This is crucial for the practical application of the model, where it will be exposed to previously unseen surveillance videos and will need to accurately identify instances of violence and vandalism.

The results obtained with the MobileNet-BiLSTM model are very encouraging, indicating the effectiveness of this approach in detecting violence and vandalism in surveillance videos. This combination takes advantage of the previous training carried out on the MobileNet network, with millions of images, with the BiLSTM architecture, resulting in improved performance and robustness in video analysis tasks..

5.1.5 MobileNet-BiLSTM

The results shown in table 10 illustrate the performance of various iterations of the MobileNet-BiLSTM model, which integrates components of the MobileNet [132] image classification model and the **BiLSTM** architecture for video analysis.

Table 10: Results from the Mobile-BiLSTM model

Version	Unfrozen layers	Dropout	**LSTM** units	Accuracy
#3	Last 30	0.2	128	97.768%
#1	Last 30	0.2	64	97.768%
#2	Last 30	0.2	32	97.321%

The best model was version 3, which achieved an accuracy of 97.768%. This version had the last 30 layers of the MobileNet model unfrozen, where the dropout after this network was 0.2 and used 128 bidirectional LSTM units. The configuration of this model, with selective layer training and appropriate dropout, allowed for effective feature extraction and temporal modeling, resulting in superior performance compared to other versions.

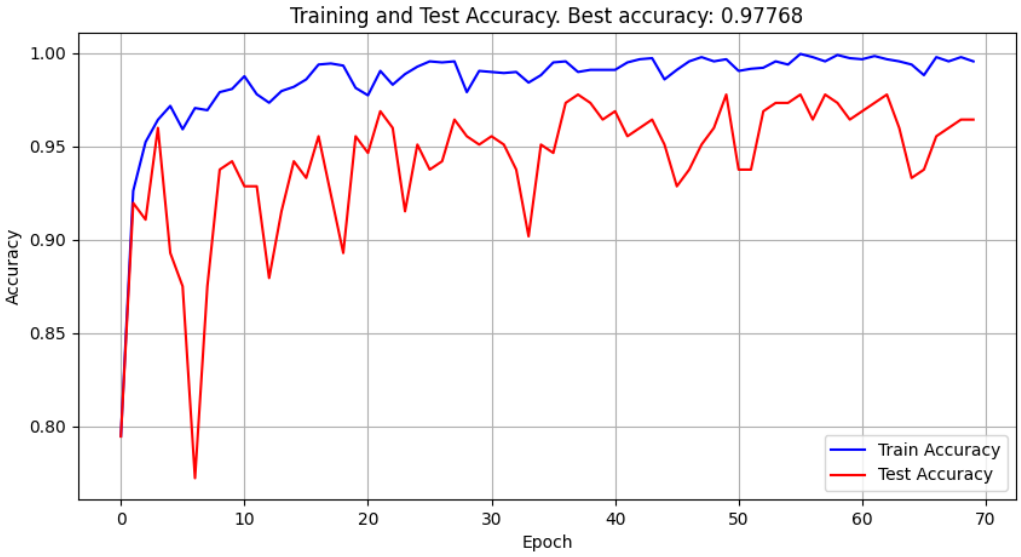


Figure 37: Training and test accuracy of the best MobileNet-BiLSTM model

As expected, the accuracy in the tests is slightly lower than in the training, due to the presence of data not seen during training. However, the results are remarkable, indicating robust and consistent model performance across different runs and test datasets. The high and consistent accuracy across training epochs reflects the effectiveness of using pre-trained models and BiLSTM layers in capturing both the spatial and temporal characteristics of surveillance videos.

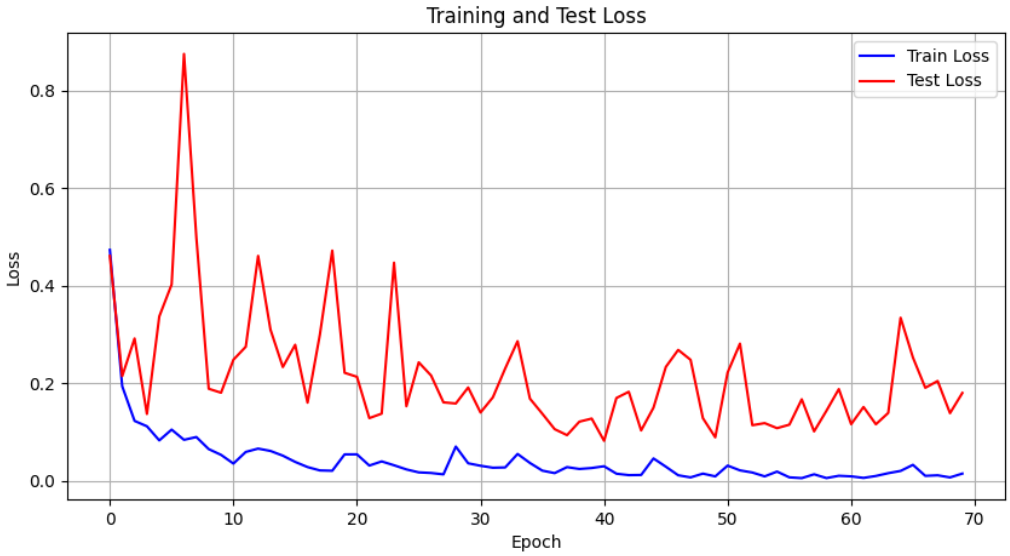


Figure 38: Training and test loss of the best MobileNet-BiLSTM model

The loss during testing shows some peaks, however, the loss generally maintains stable values, which

indicates balanced and normal training. This suggests that the model is able to achieve good generalization while maintaining consistent performance on data not observed during training.

5.2 Comparative analysis

This section compares the best models of the various types of architecture, with the aim of visualising the differences that make these models adaptable to the task of intelligent video surveillance.

Table 11: Comparison of the results from the different models

Model	Version	Conv. Filters	Conv. Kernel Sizes	Pooling Layers	Dropout Layers	**LSTM** units	Accuracy
CNN-BiLSTM	#9	32, 64, 128, 128	All (3, 3)	All (2, 2)	All 0.1	64	93.750%
ConvLSTM	#3	16, 20, 24, 28	All (3, 3)	All (1, 2, 2)	All 0.1	None	92.857%
LRCN	#6	32, 64, 128, 128	All (3, 3)	All (2, 2)	All 0.1	128	92.857%
Conv3D	#8	32, 64, 64, 128	All (3, 3, 3)	(1, 2, 2), (1, 2, 2), (1, 4, 4) and (1, 4, 4)	All 0.1	None	91.071%

The results table reveals that the **CNN-BiLSTM** model achieved the highest accuracy, registering an impressive 93.750%. This architecture takes advantage of the ability of convolutional networks to extract important spatial characteristics from video frames, while **BiLSTM** units effectively capture temporal information, allowing for a robust and accurate analysis of patterns of violence and vandalism.

For its part, the ConvLSTM model also demonstrated outstanding performance, with an accuracy of 92.857%. The positioning of the **LSTM** layers, together with the convolutional layers, along the network, as opposed to after the convolutional layers such as the **LRCN** and **CNN-BiLSTM** models, allows for efficient modelling of spatio-temporal patterns in the videos, contributing to accurate detection of unwanted events.

The **LRCN** approach, which combines **CNN** and **LSTM** elements, also achieved an accuracy of 92.857%, identical to the ConvLSTM model, but showed worse results in terms of loss than the aforementioned model.

Finally, the Conv3D model, with an accuracy of 91.071%, uses three-dimensional convolutions to directly analyse data volumes representing temporal image sequences. Although it still performs respectably, its effectiveness may be somewhat lower due to the problems mentioned above.

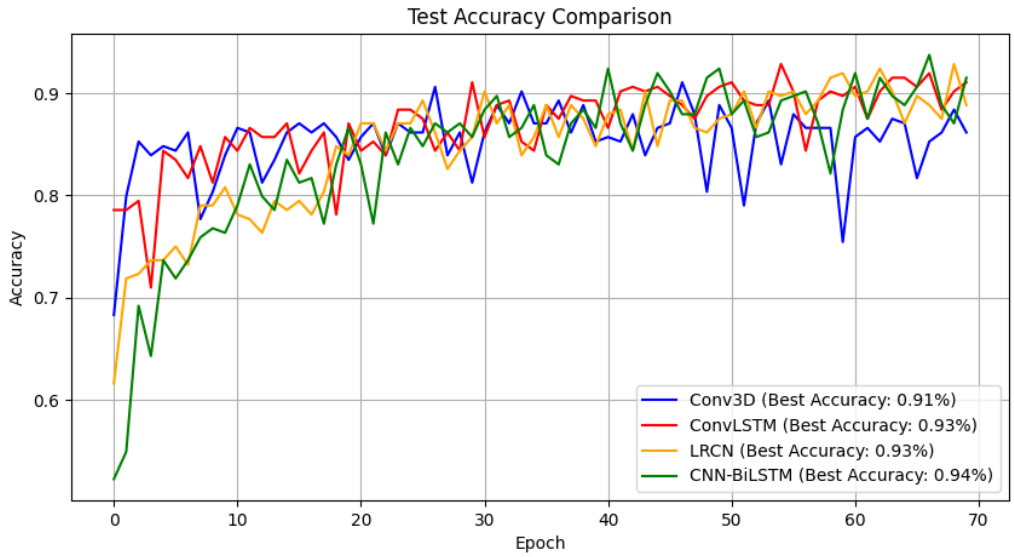


Figure 39: Comparison between the test accuracy of the four models over the epochs

Over the course of the epochs, it is possible to observe distinct variations in test accuracy between the different models. The Conv3D model showed a more unstable performance, with more pronounced fluctuations throughout the epochs, and its test accuracy fell short of expectations compared to the other models. In contrast, the other models showed a more stable trend over time, with consistent increases in accuracy during the first few epochs, followed by stabilisation at a high level. This analysis suggests that, although all the models achieved good overall performance, Conv3D may require additional adjustments to improve its stability and generalisation capacity.

With regard to loss, it can be seen that the ConvLSTM model, figure 32, stood out for keeping the test loss curve closer to the training loss curve throughout the epochs. This suggests that the ConvLSTM model was able to generalise more effectively, avoiding overfitting and maintaining a balance between performance on training and test data. On the other hand, the other models showed a larger, but normal, discrepancy between the training and test loss curves, particularly the Conv3D model (figure 30), which may be overfitting the training data. This analysis underscores the effectiveness of the ConvLSTM model in its generalisation capacity and highlights its robustness in dealing with data not seen during training.

In the models presented, both Conv3D and ConvLSTM perform convolutions in three dimensions, while **CNN-BiLSTM** and **LRCN** apply convolutions in two dimensions. Convolutions in three dimensions, as

used in the Conv3D and ConvLSTM models, take into account not only the spatial dimensions (height and width) of the images, but also the temporal dimension (depth), which represents the sequence of frames in a video. This approach allows the model to capture spatio-temporal characteristics while performing convolutions on the input data. On the other hand, the **CNN-BiLSTM** and **LRCN** models only perform convolutions on the spatial dimensions, leaving the temporal dimension to the **LSTM** layers after the convolutions. It can be seen from the results that both approaches are valid, with neither appearing to have any significant advantages over the other.

To summarise, the results highlight the effectiveness of approaches combining convolutional and recurrent networks for detecting violence and vandalism in surveillance videos. The **CNN-BiLSTM** model stands out as the most promising choice in this set of experiments, demonstrating an exceptional ability to learn and generalise complex patterns in video data.

In order to compare the performance of a pre-trained model with the models developed from scratch, the two best models trained from scratch (ConvLSTM and **CNN-BiLSTM**) were compared with MobileNet-BiLSTM.

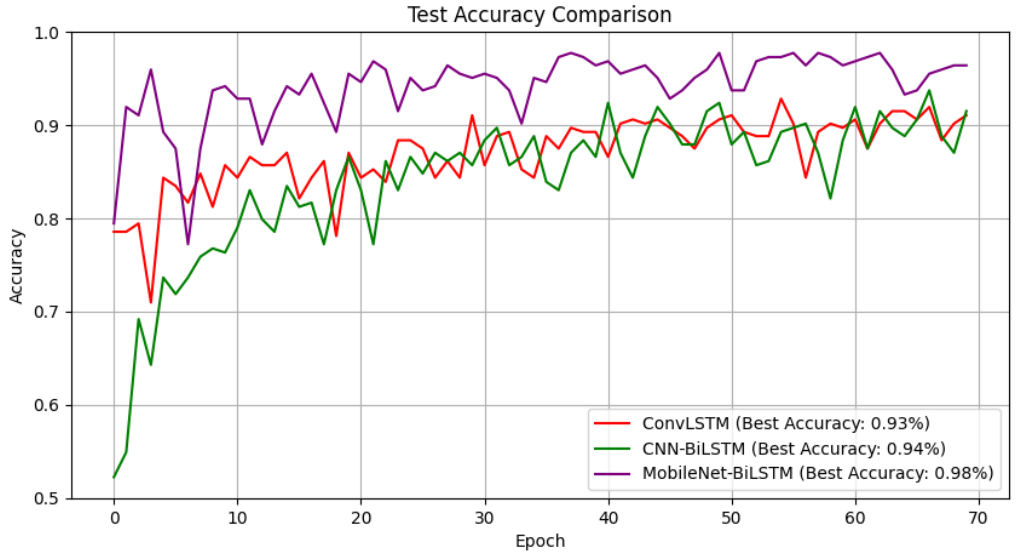


Figure 40: Comparison between the test accuracy of the two best models made from scratch with MobileNet-BiLSTM

The comparison of the two top-performing models developed from scratch, ConvLSTM and **CNN-BiLSTM**, with the MobileNet-BiLSTM model, reveals that the MobileNet-BiLSTM demonstrates superior performance. As illustrated in the graph, the MobileNet-BiLSTM achieved the highest accuracy, standing out as the most effective approach for detecting violence and vandalism in surveillance videos.

It was expected that the pre-trained model would show superior results, for a few reasons. Firstly, the use of MobileNet, which has been pre-trained on a vast ImageNet dataset, allows the model to transfer the knowledge acquired from millions of images, providing a solid basis for the extraction of visual features. In addition, MobileNet is optimized for efficiency and performance, ensuring that the convolutional layers identify complex visual patterns more effectively. Combining these capabilities with **BiLSTM** units, which capture temporal sequences with high precision, results in a more robust and accurate model compared to models developed from scratch.

Preliminary tests have been carried out on other datasets, but the results presented do not yet indicate that the models are capable of being used for a real task. Therefore, additional investment in more varied datasets and longer training times is needed to achieve the desired robustness and accuracy of the violence and vandalism detection models.

Based on these results, it is clear that creating a robust video surveillance system would require training these models with significantly larger datasets and over a considerably longer period of time. Using extensive datasets would allow the models to capture a greater diversity of scenarios and conditions, thus improving their ability to generalize and accurately detect events of violence and vandalism. Different datasets provide alternative situations necessary for the model to generalize to all situations. For example, this dataset focuses mainly on videos recorded by cell phones or television programs, while other datasets with different types of images, such as still images from video surveillance cameras, would provide more robust training for the models.

In addition, a longer training time would help to refine the model's parameters, further improving its accuracy and robustness, which is essential for effective application in real video surveillance environments.

5.3 Analysis of Explainability

After presenting and discussing the results obtained using **ML** models for detecting violence and vandalism, it is essential to dedicate a section to explaining these models and how they work. To address this, we used the **Grad-CAM** technique, which allows us to visualize which regions of the image contributed most to the model's decision.

This analysis uses the MobileNet-BiLSTM model to detect violence in videos. The **Grad-CAM** technique was applied to generate heat maps that highlight the regions of interest in the video, indicating which parts of the frames contributed most to the prediction of violence. Initially, the pre-trained model and the

video of interest are loaded. The video is segmented into frames, which are then resized to the dimensions expected by the model. Next, the internal MobileNet model is accessed, and it is essential to apply **Grad-CAM** directly to the convolutional layers of the MobileNet network. For each video frame, the **Grad-CAM** is calculated. This involves passing the frame through the model, obtaining the activations and gradients of the layer of interest, and creating a weighted heat map based on the gradients. The heat map is then superimposed on the original frame, creating a visualization that highlights the most influential regions for the model's decision. Finally, smoothing and coloring techniques are used to improve the readability of the heat maps.

The frames of the original videos and their respective heat maps generated with **Grad-CAM** are shown below. The images are organized sequentially, showing how the model identifies areas of interest over time.

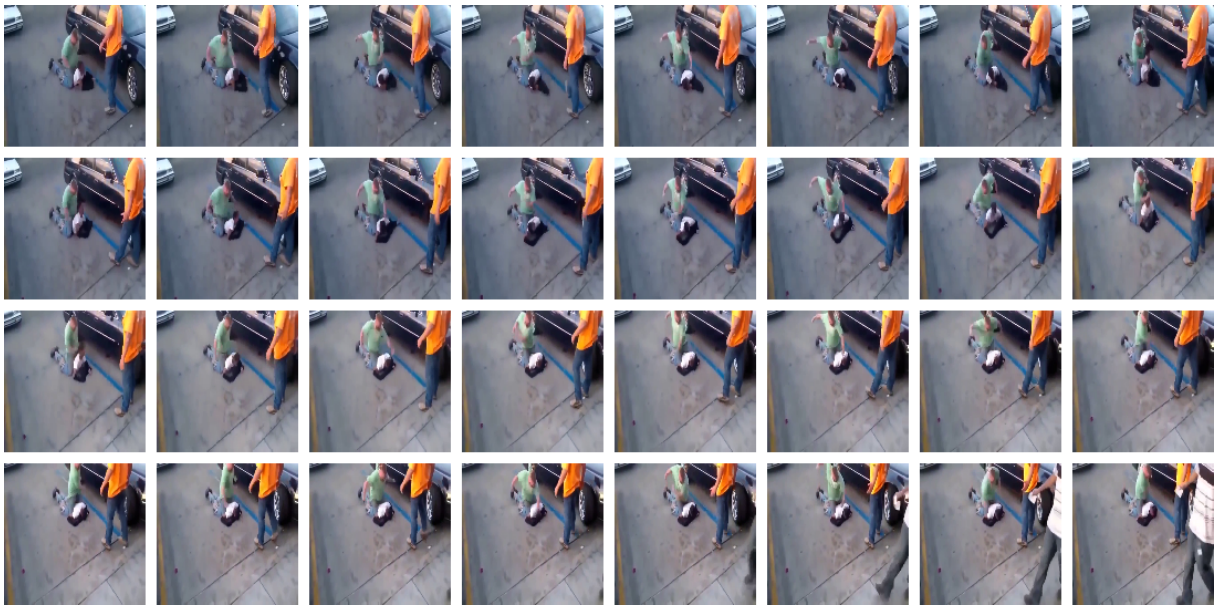


Figure 41: Original frames, obtained from a violence video

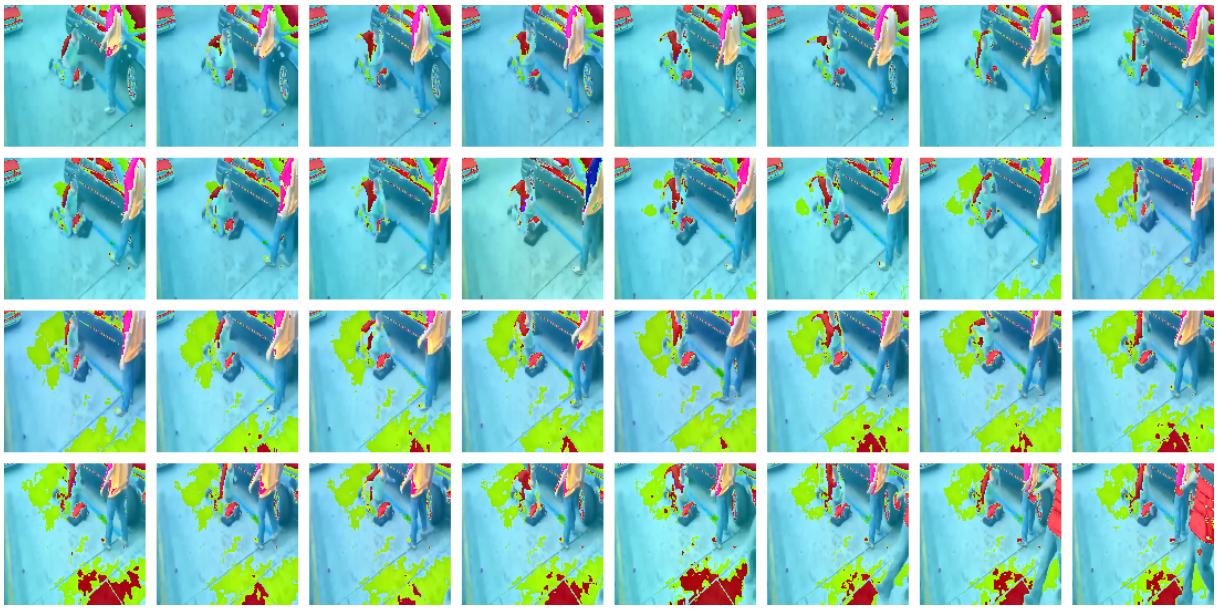


Figure 42: Heat maps obtained with **Grad-CAM**

The heat maps clearly show where the model focuses its attention when making predictions about the images. In particular, we observed that the model highlights areas of the frames that correspond to critical events, such as punches and violent interactions. This indicates that the model is using important visual characteristics to detect specific activities. In these frames, the heat maps show regions of high activation around areas where intense movements or physical interactions occur. For example, in images where punches are visible, the heat map often highlights the areas around the arms of the people involved. This is crucial for the accuracy of the model, as it correctly identifies the relevant locations that contribute to the prediction of violent activity.

Visualizing the heat maps helps validate the model's behaviour and identify possible areas for improvement. For example, if the heat map does not focus on the expected areas, it may indicate that the model needs more training or more representative data to improve its detection capacity.

The application of **Grad-CAM** provides valuable insight into the inner workings of the model, allowing it to identify and understand the regions that influence its decisions. This technique not only improves interpretability, but can also help in the validation and refinement of models, ensuring that they are focusing on the correct characteristics of the input data.

Chapter 6

Conclusion and Future Work

Violence and vandalism are problems that are present in our daily lives, and issues such as urban safety are constantly being addressed. The occurrence of aggressive and destructive behavior in the urban environment can generate significant risks for public safety, especially when such behavior happens frequently and intensely. In particular, this work has focused on the issue of intelligent surveillance, which currently includes several important stages, one of the main ones being the detection of violence and vandalism through the use of **AI**. The first activities carried out in this study involved an in-depth investigation of the problems related to urban security. The various crime prevention strategies that are currently in use were analyzed and existing solutions for video surveillance were investigated. This study was also dedicated to evaluating the possibility of integrating advanced **AI** concepts with these solutions, with the aim of achieving optimum levels of security.

Various aspects of video surveillance were investigated, including the different types of cameras used, exploring the various categories of cameras that exist, as well as the data capture methods they employ. The most modern devices are capable of recording images and videos in high definition, using advanced sensors to capture precise details, as well as incorporating **AI**-based technologies to assist in their tasks. In addition, data storage systems have been explored, from local solutions to more modern options that use cloud computing, allowing surveillance data to be stored and accessed remotely. This whole idea of surveillance raises important ethical considerations, such as the privacy of monitored individuals, informed consent, and the need for clear regulations to ensure that video surveillance is used fairly and responsibly, protecting both public safety and citizens' rights.

After this initial investigation into video surveillance, the study looked at the application of **AI** in this context. Fundamental concepts of **ML**, **DL** and **CV** were explored. The analysis included how these fields can be integrated into video surveillance systems to improve the accuracy and efficiency of event detection. Also discussed were the different types of datasets used to train **AI** models, ensuring that they are robust and representative of real surveillance scenarios. In addition, several applicable model architectures were

examined, which are particularly effective at processing and analyzing visual data. These technologies enable video surveillance systems to perform advanced tasks such as suspicious behavior detection and predictive analysis, making them powerful tools for public safety.

To complement this analysis, five research articles related to the theme of this work were explored. These articles cover subjects such as the appropriate treatment of data, the use of contextual data such as audio, techniques to avoid overfitting and the adjustment of model hyperparameters. In this way, this critical analysis made it possible to define objectives capable of significantly complementing the main objective of this dissertation. In other words, the techniques present in these studies have been adapted to be applied in this work to improve the effectiveness of intelligent video surveillance systems, ensuring more effective detection of violence and vandalism.

Next, the tasks carried out were of a practical nature, consisting of analyzing and preparing the datasets, designing the models and carrying out the experiments. At this stage, an in-depth analysis of the dataset used was initially carried out, assessing the quality of the data to determine whether it was suitable for the purpose of detecting violence and vandalism. This analysis included checking the integrity and completeness of the data. In addition, a detailed statistical analysis of the videos was carried out, examining aspects such as their duration, number of frames, among others. With regard to data processing, the videos were divided into frames with equal intervals, ensuring a uniform sampling of the events captured. 32 frames were extracted per video, each with a resolution of 128x128 pixels. This approach made it possible to capture enough detail for the analysis, while maintaining adequate computational efficiency. With this methodology, it was possible to prepare a robust and representative dataset, essential for the development and training of **AI** models focused on detecting violence and vandalism.

Finally, several advanced **AI** models for detecting violence and vandalism in video surveillance systems were built and evaluated. Models such as Conv3D, ConvLSTM, **LRCN**, **CNN-BiLSTM** and the pre-trained MobileNet-BiLSTM model were developed and tested. Each of these models uses a specific combination of **CNN**, **LSTM** networks and, in the case of MobileNet-BiLSTM, a pre-trained architecture for efficient extraction of visual features and temporal analysis.

These approaches were designed to capture both spatial and temporal features of the videos, allowing for detailed analysis of suspicious movements and behaviors. Combining convolutions with LSTMs provided a better understanding of complex sequential patterns, while the pre-trained MobileNet-BiLSTM model took advantage of prior knowledge to quickly adapt to the specific problem of detecting anomalies in surveillance videos.

The results show that of the models built from scratch, the **CNN-BiLSTM** model achieved the highest

accuracy, registering an impressive 93.750%. This architecture takes advantage of the ability of convolutional networks to extract important spatial features from video frames, while the **BiLSTM** units effectively capture temporal information. With regard to loss, it was observed that the ConvLSTM model stood out for keeping the test loss curve closer to the training loss curve throughout the epochs. This suggests that the ConvLSTM model was able to generalize more effectively, avoiding overfitting and maintaining a balance between performance on training and test data. On the other hand, the other models showed a larger but normal discrepancy between the training and test loss curves, particularly the Conv3D model, which may be overfitting the training data.

The MobileNet-BiLSTM model was expected to show the best results, achieving an accuracy of 97.768%, standing out as the most effective solution among those tested. The use of the pre-trained MobileNet network on the ImageNet dataset provides a robust foundation for the extraction of visual features, which combined with BiLSTM units that capture temporal sequences, makes the model more robust and accurate compared to those developed from scratch.

The main scientific achievement of this research is the recognition of the MobileNet-BiLSTM model as the great architecture for identifying violence and vandalism in surveillance videos. The effectiveness of this model underscores its potential practicality in scenarios where precise and dependable detection is essential. Additionally, the use of pre-trained architectures highlights the importance of leveraging established models to enhance performance and reliability in detecting such critical activities.

Furthermore, this study offers additional contributions to the existing research analyzed, such as the comparison of pre-trained models with those created from the ground up, highlighting the trade-offs between utilizing existing knowledge and employing tailored training methods. Pre-trained models can learn faster and have enhanced generalization capabilities, whereas models developed from scratch demand greater computational resources and time investment. Additionally, the examination of explainability enhances the understanding of the model's decision-making processes, ensuring that detection systems maintain not only accuracy but also transparency and comprehensibility for end-users, a crucial aspect in high-stakes surveillance contexts.

In summary, the results obtained demonstrate the significant potential of artificial intelligence in improving public safety through more intelligent and effective video surveillance systems. Each model made a unique contribution to the accurate detection of suspicious behavior, offering promising solutions for practical implementation in real monitoring environments.

Based on the results presented, it is clear that creating a robust video surveillance system will require training these models with significantly larger datasets and over a considerably longer period of time. In the

future, the exploration of comprehensive datasets will enable the models to cover a greater diversity of scenarios and conditions, thereby improving their ability to generalize and accurately detect events of violence and vandalism. In addition, a longer training period would help improve the model's parameters, further increasing its accuracy and robustness, which are fundamental aspects for effective application in real video monitoring scenarios. It is therefore recommended that future efforts focus on expanding datasets and extending training periods to ensure that violence and vandalism detection models can operate reliably in a variety of surveillance situations.

Bibliography

[1] T. Ribeiro, P. Oliveira, and M. Rodrigues. Next-generation surveillance: Exploring the intersection of artificial intelligence and security. In Kohei Arai, editor, *Intelligent Systems and Applications*, pages 522–535, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-66428-1. doi: 10.1007/978-3-031-66428-1_33.

[2] P. A. Hancock. On the nature of vigilance. *Human Factors*, 59(1):35–43, 2017. doi: 10.1177/0018720816655240.

[3] J. D. Osofsky. The impact of violence on children. *The Future of Children*, pages 33–49, 1999. doi: 10.2307/1602780.

[4] Emily A. Greenfield and Nadine F. Marks. Identifying experiences of physical and psychological violence in childhood that jeopardize mental health in adulthood. *Child Abuse & Neglect*, 34(3): 161–171, 2010. doi: 10.1016/j.chiabu.2009.08.012.

[5] World Health Organization. Violence prevention: The evidence, 2010. URL https://www.who.int/publications/i/item/violence-prevention-the-evidence.

[6] Arnold P. Goldstein. *The Psychology of Vandalism*. Springer Science & Business Media, 2013.

[7] Sistema de Segurança Interna. Relatorio anual de segurança interna (rasi), 2022. URL https://www.portugal.gov.pt/pt/gc23/comunicacao/documento?i=relatorio-anual-de-seguranca-interna-2022-.

[8] E. L. Piza, B. C. Welsh, D. P. Farrington, and A. L. Thomas. Cctv surveillance for crime prevention: A 40-year systematic review with meta-analysis. *Criminology & Public Policy*, 18(1):135–159, 2019. doi: 10.1111/1745-9133.12419.

[9] Martin Gill and Angela Spriggs. Assessing the impact of cctv. *Home Office Research Study*, 2005. URL https://api.semanticscholar.org/CorpusID:154642508.

[10] G. J. Smith. Behind the screens: Examining constructions of deviance and informal practices among cctv control room operators in the uk. *Surveillance & Society*, 2(2/3), 2004. doi: 10.24908/ss.v2i2/3.3384.

[11] C. J. Howard, T. Troscianko, I. D. Gilchrist, A. Behera, and D. C. Hogg. Suspiciousness perception in dynamic scenes: A comparison of cctv operators and novices. *Frontiers in Human Neuroscience*, 7:441, 2013. doi: 10.3389/fnhum.2013.00441.

[12] T. Murphy. The admissibility of cctv evidence in criminal proceedings. *International Review of Law, Computers & Technology*, 13(3):383–404, 1999. doi: 10.1080/13600869955044.

[13] A. Hampapur, L. Brown, J. Connell, S. Pankanti, A. Senior, and Y. Tian. Smart surveillance: Applications, technologies and implications. In *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia*, volume 2, pages 1133–1138, Singapore, 2003. doi: 10.1109/ICICS.2003.1292637.

[14] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer London, 1 edition, 2010. doi: 10.1007/978-1-84882-935-0.

[15] Weiyi Shi. The application of image processing in the criminal investigation. In *Proceedings of the 2016 4th International Conference on Machinery, Materials and Information Technology Applications*, pages 139–142. Atlantis Press, 2017/01. ISBN 978-94-6252-285-5. doi: 10.2991/icmmita-16.2016.28.

[16] Jordi Blanes i Vidal and Tom Kirchmaier. The effect of police response time on crime clearance rates. *The Review of Economic Studies*, 85(2):855–891, 2018. doi: 10.1093/restud/rdx044.

[17] D. Almeida, K. Shmarko, and E. Lomas. The ethics of facial recognition technologies, surveillance, and accountability in an age of artificial intelligence: A comparative analysis of us, eu, and uk regulatory frameworks. *AI and Ethics*, 2(3):377–387, 2022. doi: 10.1007/s43681-021-00077-w.

[18] Waseem Ullah, Amin Ullah, Tanveer Hussain, Khan Muhammad, Ali Asghar Heidari, Javier Del Ser, Sung Wook Baik, and Victor Hugo C. De Albuquerque. Artificial intelligence of things-assisted two-stream neural network for anomaly detection in surveillance big video data. *Future Generation Computer Systems*, 129:286–297, 2021. doi: 10.1016/j.future.2021.10.033.

[19] N. Haering, P. L. Venetianer, and A. Lipton. The evolution of video surveillance: An overview. *Machine Vision and Applications*, 19(5-6):279–290, 2008. doi: 10.1007/s00138-008-0152-0.

[20] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed. A review of video surveillance systems. *Journal of Visual Communication and Image Representation*, 77:103116, 2021. doi: 10.1016/j.jvcir.2021.103116.

[21] M. P. Ashby. The value of cctv surveillance cameras as an investigative tool: An empirical analysis. *European Journal on Criminal Policy and Research*, 23(3):441–459, 2017. doi: 10.1007/s10610-017-9341-6.

[22] Fredrik Nilsson. *Intelligent Network Video: Understanding Modern Video Surveillance Systems*. CRC Press, 2008. doi: 10.1201/9781420061574.

[23] G. Szwoch, P. Dalka, A. Ciarkowski, P. Szczuko, and A. Czyzewski. Visual object tracking system employing fixed and ptz cameras. *Intelligent Decision Technologies*, 5(2):177–188, 2011. doi: 10.3233/IDT-2011-0105.

[24] R. Bodor, A. Drenner, P. Schrater, and N. Papanikolopoulos. Optimal camera placement for automated surveillance tasks. *Journal of Intelligent and Robotic Systems*, 50:257–295, 2007. doi: 10.1007/s10846-007-9164-7.

[25] A. Aakerberg, K. Nasrollahi, and T. B. Moeslund. Real-world super-resolution of face-images from surveillance cameras. *IET Image Processing*, 16(2):442–452, 2022. doi: 10.1049/ipr2.12359.

[26] S. Arora, K. Bhatia, and V. Amit. Storage optimization of video surveillance from cctv camera. In *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, pages 710–713. IEEE, 10 2016. doi: 10.1109/NGCT.2016.7877503.

[27] J. Y. Byun, A. Nasridinov, and Y. H. Park. Internet of things for smart crime detection. *Contemporary Engineering Sciences*, 7(15):749–754, 2014. doi: 10.12988/ces.2014.4685.

[28] L. Ye, T. Liu, T. Han, H. Ferdinando, T. Seppänen, and E. Alasaarela. Campus violence detection based on artificial intelligent interpretation of surveillance video sequences. *Remote Sensing*, 13 (4):628, 2021. doi: 10.3390/rs13040628.

[29] D. A. Rodríguez-Silva, L. Adkinson-Orellana, F. J. Gonz'lez-Castano, I. Armino-Franco, and D. Gonz'lez-Martinez. Video surveillance based on cloud storage. In *2012 IEEE Fifth International Conference on Cloud Computing*, pages 991–992. IEEE, 6 2012. doi: 10.1109/CLOUD.2012.44.

[30] DJ Neal. Video surveillance in the cloud? *International Journal on Cryptography and Information Security*, 2(3):1–19, 2012. doi: 10.48550/arXiv.1512.00070.

[31] C. R. Rachana, R. Banu, G. A. Ahammed, and B. D. Parameshachari. Cloud computing–a unified approach for surveillance issues. In *IOP Conference Series: Materials Science and Engineering*, volume 225, page 012073. IOP Publishing, 8 2017. doi: 10.1088/1757-899X/225/1/012073.

[32] T. Winkler and B. Rinner. Privacy and security in video surveillance. In *Intelligent Multimedia Surveillance: Current Trends and Research*, pages 37–66. Springer, Berlin, Heidelberg, 2013. doi: 10.1007/978-3-642-41512-8_3.

[33] Q. M. Rajpoot and C. D. Jensen. Security and privacy in video surveillance: Requirements and challenges. In *IFIP International Information Security Conference*, pages 169–184. Springer Berlin Heidelberg, 6 2014. doi: 10.1007/978-3-642-55415-5_14.

[34] A. A. Adams and J. M. Ferryman. The future of video analytics for surveillance and its ethical implications. *Security Journal*, 28:272–289, 2015. doi: 10.1057/sj.2012.48.

[35] Evan Selinger and Bernard Leong. The ethics of facial recognition technology. *Forthcoming in The Oxford Handbook of Digital Ethics, ed. Carissa Véliz*, 2021. doi: 10.2139/ssrn.3762185.

[36] A. Simpson. Surveillance, cctv and body-worn cameras in mental health care. *Journal of Mental Health*, 32(2):369–372, 2023. doi: 10.1080/09638237.2023.2194988.

[37] Clarion Security Systems. How many cctv cameras are in london?, 2022. URL https://tinyurl.com/2mp583c4.

[38] I. Evans. Report: London no safer for all its cctv cameras. *The Christian Science Monitor*, page 1, 2012.

[39] N. Samala, B. S. Katkam, R. S. Bellamkonda, and R. V. Rodriguez. Impact of ai and robotics in the tourism sector: A critical insight. *Journal of Tourism Futures*, 8(1):73–87, 2020. doi: 10.1108/JTF-07-2019-0065.

[40] X. Qin and Z. Jiang. The impact of ai on the advertising process: The chinese experience. *Journal of Advertising*, 48(4):338–346, 2019. doi: 10.1080/00913367.2019.1652122.

[41] M. Haenlein and A. Kaplan. A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. *California Management Review*, 61(4):5–14, 2019. doi: 10.1177/0008125619864925.

[42] S. Muggleton. Alan turing and the development of artificial intelligence. *AI Communications*, 27(1): 3–10, 2014. doi: 10.3233/AIC-130579.

[43] A. M. Turing. Computing machinery and intelligence. In *Parsing the Turing Test*, pages 23–65. Springer, Dordrecht, 2009. doi: 10.1007/978-1-4020-6710-5_3.

[44] J. E. (Hans) Korteling, G. C. van de Boer-Visschedijk, R. A. M. Blankendaal, R. C. Boonekamp, and A. R. Eikelboom. Human- versus artificial intelligence. *Frontiers in Artificial Intelligence*, 4, 2021. ISSN 2624-8212. doi: 10.3389/frai.2021.622364.

[45] S. Makridakis. The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms. *Futures*, 90:46–60, 2017. doi: 10.1016/j.futures.2017.03.006.

[46] Fangxing Li and Yan Du. From alphago to power system ai: What engineers can learn from solving the most complex board game. *IEEE Power and Energy Magazine*, 16(2):76–84, 2018. doi: 10.1109/MPE.2017.2779554.

[47] Katja Grace, John Salvatier, Allan Dafoe, Baobao Zhang, and Owain Evans. When will ai exceed human performance? evidence from ai experts, 2018.

[48] S. O. Abioye, L. O. Oyedele, L. Akanbi, A. Ajayi, J. M. D. Delgado, M. Bilal, and A. Ahmed. Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. *Journal of Building Engineering*, 44:103299, 2021. doi: 10.1016/j.jobe.2021.103299.

[49] J. M. Helm, A. M. Swiergosz, H. S. Haeberle, J. M. Karnuta, J. L. Schaffer, V. E. Krebs, and P. N. Ramkumar. Machine learning and artificial intelligence: Definitions, applications, and future directions. *Current Reviews in Musculoskeletal Medicine*, 13:69–76, 2020. doi: 10.1007/s12178-020-09600-8.

[50] I. El Naqa and M. J. Murphy. What is machine learning? In *Machine Learning in Radiation Oncology*, pages 3–11. Springer International Publishing, 2015. doi: 10.1007/978-3-319-18305-3_1.

[51] Gopinath Rebala, Ajay Ravi, and Sanjay Churiwala. *Machine Learning Definition and Basics*, pages 1–17. Springer International Publishing, 2019. doi: 10.1007/978-3-030-15729-6_1.

[52] B. Mahesh. Machine learning algorithms - a review. *International Journal of Science and Research (IJSR)*, 9(1):381–386, 2020. doi: 10.21275/ART20203995.

[53] B. Liu and B. Liu. Supervised learning. In *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*, pages 63–132. Springer, Berlin, Heidelberg, 2011. doi: 10.1007/978-3-642-19460-3_3.

[54] G. James, D. Witten, T. Hastie, R. Tibshirani, and J. Taylor. Unsupervised learning. In *An Introduction to Statistical Learning: with Applications in Python*, pages 503–556. Springer International Publishing, 2023. doi: 10.1007/978-3-031-38747-0_12.

[55] M. A. Wiering and M. Van Otterlo. Reinforcement learning and markov decision processes. In *Adaptation, Learning, and Optimization*, volume 12, page 729. Springer, Berlin, Heidelberg, 2012. doi: 10.1007/978-3-642-27645-3.

[56] J. E. van Engelen and H. H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109 (2):373–440, 2020. doi: 10.1007/s10994-019-05855-6.

[57] S. Dong, P. Wang, and K. Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021. doi: 10.1016/j.cosrev.2021.100379.

[58] E. M. Onyema, P. K. Shukla, S. Dalal, M. N. Mathur, M. Zakariah, and B. Tiwari. Enhancement of patient facial recognition through deep learning algorithm: Convnet. *Journal of Healthcare Engineering*, 2021, 2021. doi: 10.1155/2021/5196000.

[59] I. N. Silva, D. H. Spatti, R. A. Flauzino, L. Liboni, and S. Alves. *Artificial Neural Networks: A Practical Course*. Springer, 2016. doi: 10.1007/978-3-319-43162-8.

[60] A.K. Jain, Jianchang Mao, and K.M. Mohiuddin. Artificial neural networks: a tutorial. *Computer*, 29(3):31–44, 1996. doi: 10.1109/2.485891.

[61] M. M. Mehdy, P. Y. Ng, E. F. Shair, N. I. Saleh, and C. Gomes. Artificial neural networks in image processing for early detection of breast cancer. *Computational and Mathematical Methods in Medicine*, 2017. doi: 10.1155/2017/2610628.

[62] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017. doi: 10.48550/arXiv.1708.07747.

[63] Xue Feng, Yu Jiang, Xi Yang, Min Du, and Xiaolin Li. Computer Vision Algorithms and Hardware Implementations: A Survey. *Integration*, 69:309–320, 2019. doi: 10.1016/j.vlsi.2019.07.005.

[64] Vishnu Kakani, Van Hoan Nguyen, B. Praveen Kumar, Hyundong Kim, and Visweswara Rao Pasupuleti. A Critical Review on Computer Vision and Artificial Intelligence in Food Industry. *Journal of Agriculture and Food Research*, 2:100033, 2020. doi: 10.1016/j.jafr.2020.100033.

[65] D. I. Patrício and R. Rieder. Computer Vision and Artificial Intelligence in Precision Agriculture for Grain Crops: A Systematic Review. *Computers and Electronics in Agriculture*, 153:69–81, 2018. doi: 10.1016/j.compag.2018.08.001.

[66] A. A. Khan, A. A. Laghari, and S. A. Awan. Machine Learning in Computer Vision: A Review. *EAI Endorsed Transactions on Scalable Information Systems*, 8(32):e4–e4, 2021. doi: 10.4108/eai.21-4-2021.169418.

[67] Qiang Li, Wenjia Cai, Xiaogang Wang, Yu Zhou, David D. Feng, and Min Chen. Medical Image Classification with Convolutional Neural Network. In *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 844–848. IEEE, 2014. doi: 10.1109/ICARCV.2014.7064414.

[68] Jason Brownlee. *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery, 2019.

[69] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma. A review of yolo algorithm developments. *Procedia Computer Science*, 199:1066–1073, 2022. doi: 10.1016/j.procs.2022.01.135.

[70] R. Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. doi: 10.48550/arXiv.1504.08083.

[71] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing, 2015. doi: 10.48550/arXiv.1505.04597.

[72] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017. doi: 10.48550/arXiv.1703.06870.

[73] Tristan M. Ward, Pietro Mascagni, Yohan Ban, Guy Rosman, Nicolas Padoy, Octavio Meireles, and Daniel A. Hashimoto. Computer Vision in Surgery. *Surgery*, 169(5):1253–1256, 2021. doi: 10.1016/j.surg.2020.10.039.

[74] Alun Preece. Asking 'why'in ai: Explainability of intelligent systems–perspectives and challenges. *Intelligent Systems in Accounting, Finance and Management*, 25(2):63–72, 2018. doi: 10.1002/isaf.1422.

[75] Adrien Bibal, Michael Lognoul, Alexandre De Streel, and Benoît Frénay. Legal requirements on explainability in machine learning. *Artificial Intelligence and Law*, 29:149–169, 2021. doi: 10.1007/s10506-020-09270-4.

[76] Aniek F Markus, Jan A Kors, and Peter R Rijnbeek. The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. *Journal of biomedical informatics*, 113:103655, 2021. doi: 10.1016/j.jbi.2020.103655.

[77] David Charte, Francisco Charte, Maria J del Jesus, and Francisco Herrera. An analysis on the use of autoencoders for representation learning: Fundamentals, learning task case studies, explainability and challenges. *Neurocomputing*, 404:93–107, 2020. doi: 10.1016/j.neucom.2020.04.057.

[78] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[79] Guruh Fajar Shidik, Edy Noersasongko, Adi Nugraha, Pulung Nurtantio Andono, Jumanto Jumanto, and Ernes Jatmiko Kusuma. A systematic review of intelligence video surveillance: Trends, techniques, frameworks, and datasets. *IEEE Access*, 7:170457–170473, 2019. doi: 10.1109/ACCESS.2019.2955387.

[80] Mahmoud A. Ezzat, Mohamed A. Abd El Ghany, Sultan Almotairi, and Mostafa A. M. Salem. Horizontal review on video surveillance for smart cities: Edge devices, applications, datasets, and future trends. *Sensors*, 21(9):3222, 2021. doi: 10.3390/s21093222.

[81] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6479–6488, 2018. doi: 10.48550/arXiv.1801.04264.

[82] Ming Cheng, Kuiyuan Cai, and Ming Li. RWF-2000: an open large scale video database for violence detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4183–4190, 2021. doi: 10.1109/ICPR48806.2021.9412502.

[83] Mohamed M. Soliman, Mohamed H. Kamal, Mohamed A. E. M. Nashed, Yasser M. Mostafa, Bassant S. Chawky, and Dina Khattab. Violence recognition from videos using deep learning techniques. In *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pages 80–85, 2019. doi: 10.1109/ICICIS46948.2019.9014714.

[84] Peng Wu, Jie Liu, Yun Shi, Yifan Sun, Feng Shao, Zhiyong Wu, and Zhiyong Yang. Not only look, but also listen: Learning multimodal violence detection under weak supervision. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX*, pages 322–339. Springer International Publishing, 2020. doi: 10.1007/978-3-030-58577-8_20.

[85] Baptiste Degardin and Hugo Proença. Iterative weak/self-supervised classification framework for abnormal events detection. *Pattern Recognition Letters*, 145:50–57, 2021. doi: 10.1016/j.patrec.2021.01.031.

[86] Khurram Soomro, Amir R. Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. doi: 10.48550/arXiv.1212.0402.

[87] Mislav Grgic, Krešimir Delac, and Sonja Grgic. Scface–surveillance cameras face database. *Multimedia Tools and Applications*, 51:863–879, 2011. doi: 10.1007/s11042-009-0417-2.

[88] H. Taud and J. F. Mas. Multilayer perceptron (mlp). In *Geomatic Approaches for Modeling Land Change Scenarios*, pages 451–455. Springer, 2018. doi: 10.1007/978-3-319-60801-3_27.

[89] H. Ramchoun, Y. Ghanou, M. Ettaouil, and M. A. Janati Idrissi. Multilayer perceptron: Architecture optimization and training. *International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI)*, 2016. doi: 10.9781/ijimai.2016.415.

[90] H. Foroughi, B. S. Aski, and H. Pourreza. Intelligent video surveillance for monitoring fall detection of elderly in home environments. In *2008 11th International Conference on Computer and Information Technology*, pages 219–224. IEEE, 2008. doi: 10.1109/ICCITECHN.2008.4803020.

[91] R. Chauhan, K. K. Ghanshala, and R. C. Joshi. Convolutional neural network (cnn) for image detection and recognition. In *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pages 278–282. IEEE, 2018. doi: 10.1109/ICSCCC.2018.8703316.

[92] Zhe Cai, Qiang Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV*, volume 14, pages 354–370. Springer International Publishing, 2016. doi: 10.1007/978-3-319-46493-0_22.

[93] Shereen Albawi, Thaker A Mohammed, and Safaa Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. IEEE, 2017. doi: 10.1109/ICEngTechnol.2017.8308186.

[94] Kieran O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015. doi: 10.48550/arXiv.1511.08458.

[95] P. Dhruv and S. Naskar. Image classification using convolutional neural network (cnn) and recurrent neural network (rnn): A review. In *Machine Learning and Information Processing: Proceedings of ICMLIP 2019*, pages 367–381. Springer, 2020. doi: 10.1007/978-981-15-1884-3_34.

[96] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017. doi: 10.48550/arXiv.1801.01078.

[97] Yajie Yu, Xuan Si, Chaoran Hu, and Jun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019. doi: 10.1162/neco_a_01199.

[98] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020. doi: 10.1016/j.physd.2019.132306.

[99] Waqas Ullah, Absar Ullah, Taqdir Hussain, Ziauddin Ahmad Khan, and Sung Wook Baik. An efficient anomaly recognition framework using an attention residual lstm in surveillance videos. *Sensors*, 21 (8):2811, 2021. doi: 10.3390/s21082811.

[100] Saeed Nosouhian, Faezeh Nosouhian, and Alireza K. Khoshouei. A review of recurrent neural network architecture for sequence learning: Comparison between lstm and gru. *Preprints*,

2021. doi: 10.20944/preprints202107.0252.v1. URL https://doi.org/10.20944/preprints202107.0252.v1.

[101] Silvia Cascianelli, Gabriele Costante, Thomas Andrew Ciarfuglia, Paolo Valigi, and Mario Luca Fravolini. Full-gru natural language video description for service robotics applications. *IEEE Robotics and Automation Letters*, 3(2):841–848, 2018. doi: 10.1109/LRA.2018.2793345.

[102] Asad Ullah, Khan Muhammad, Wenrui Ding, Vasile Palade, Ihtesham Ul Haq, and Sung Wook Baik. Efficient activity recognition using lightweight cnn and ds-gru network for surveillance applications. *Applied Soft Computing*, 103:107102, 2021. doi: 10.1016/j.asoc.2021.107102.

[103] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017. doi: 10.48550/arXiv.1706.03762.

[104] A. Hussain, T. Hussain, W. Ullah, and S. W. Baik. Vision transformer and deep sequence learning for human activity recognition in surveillance videos. *Computational Intelligence and Neuroscience*, 2022. doi: 10.1155/2022/3454167.

[105] Salih Vosta and Kin Choong Yow. A cnn-rnn combined structure for real-world violence detection in surveillance cameras. *Applied Sciences*, 12(3):1021, 2022. doi: 10.3390/app12031021.

[106] Zachary Baird, Michael K. McDonald, Subramani Rajan, and Soo J. Lee. A cnn-lstm network for augmenting target detection in real maritime wide area surveillance radar data. *IEEE Access*, 8: 179281–179294, 2020. doi: 10.1109/ACCESS.2020.3025144.

[107] Şule Aktı, Gazi Aksoy Tataroğlu, and Hazım Kemal Ekenel. Vision-based fight detection from surveillance cameras. In *2019 Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE, 2019. doi: 10.1109/IPTA.2019.8936070.

[108] Waqas Ullah, Tariq Hussain, Fida U. M. Ullah, Moon Young Lee, and Sung Wook Baik. Transcnn: Hybrid cnn and transformer mechanism for surveillance anomaly detection. *Engineering Applications of Artificial Intelligence*, 123:106173, 2023. doi: 10.1016/j.engappai.2023.106173.

[109] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015. doi: 10.48550/arXiv.1412.0767.

[110] M. S. Likitha, S. R. R. Gupta, K. Hasitha, and A. U. Raju. Speech-based human emotion recognition using mfcc. In *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 2257–2260. IEEE, 17. doi: 10.1109/WiSPNET.2017.8300161.

[111] F. Xiao. Generalization of dempster–shafer theory: A complex mass function. *Applied Intelligence*, 50:3266–3275, 20. doi: 10.1007/s10489-019-01617-y.

[112] S. Brayne. Big data surveillance: The case of policing. *American Sociological Review*, 82(5): 977–1008, 2017. doi: 10.1177/0003122417725865.

[113] N. Xu, A. A. Liu, Y. Wong, Y. Zhang, W. Nie, Y. Su, and M. Kankanhalli. Dual-stream recurrent neural network for video captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, 29 (8):2482–2493, 2018. doi: 10.1109/TCSVT.2018.2867286.

[114] B. Roy and H. Cheung. A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–6. IEEE, 2018. doi: 10.1109/ATNAC.2018.8615294.

[115] A. Warsi, M. Abdullah, M. N. Husen, M. Yahya, S. Khan, and N. Jawaid. Gun detection system using yolov3. In *2019 IEEE International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, pages 1–4. IEEE, 2019. doi: 10.1109/ICSIMA47653.2019.9057329.

[116] G. Sreenu and S. Durai. Intelligent video surveillance: A review through deep learning techniques for crowd analysis. *Journal of Big Data*, 6(1):1–27, 2019. doi: 10.1186/s40537-019-0212-5.

[117] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2009. doi: 10.1109/CVPR.2009.5206848.

[118] R. Ke, Y. Zhuang, Z. Pu, and Y. Wang. A smart, efficient, and reliable parking surveillance system with edge artificial intelligence on iot devices. *IEEE Transactions on Intelligent Transportation Systems*, 22(8):4962–4974, 2021. doi: 10.1109/TITS.2020.2984197.

[119] N. Chen, Y. Chen, E. Blasch, H. Ling, Y. You, and X. Ye. Enabling smart urban surveillance at the edge. In *2017 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 109–119, 17. doi: 10.1109/SmartCloud.2017.24.

[120] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. *SSD: Single Shot MultiBox Detector*, page 21–37. Springer International Publishing, 2016. doi: 10.1007/978-3-319-46448-0_2.

[121] C. Ning, H. Zhou, Y. Song, and J. Tang. Inception single shot multibox detector for object detection. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 549–554, 17. doi: 10.1109/ICMEW.2017.8026312.

[122] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111: 98–136, 2015. doi: 10.1007/s11263-014-0733-5.

[123] Z. Luo, F. Branchaud-Charron, C. Lemaire, J. Konrad, S. Li, A. Mishra, and P. M. Jodoin. Mio-tcd: A new benchmark dataset for vehicle classification and localization. *IEEE Transactions on Image Processing*, 27(10):5129–5141, 2018. doi: 10.1109/TIP.2018.2848705.

[124] I. P. Febin, K. Jayasree, and P. T. Joy. Violence detection in videos for an intelligent surveillance system using mobsift and movement filtering algorithm. *Pattern Analysis and Applications*, 23(2): 611–623, 2020. doi: 10.1007/s10044-019-00821-3.

[125] R. Hannane, A. Elboushaki, K. Afdel, P. Naghabhushan, and M. Javed. An efficient method for video shot boundary detection and keyframe extraction using sift-point distribution histogram. *International Journal of Multimedia Information Retrieval*, 5:89–104, 16. doi: 10.1007/s13735-016-0095-6.

[126] S. Ojha and S. Sakhare. Image processing techniques for object tracking in video surveillance – a survey. In *2015 International Conference on Pervasive Computing (ICPC)*, pages 1–6. IEEE, 15. doi: 10.1109/PERVASIVE.2015.7087180.

[127] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, pages 32–36 Vol.3, 2004. doi: 10.1109/ICPR.2004.1334462.

[128] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 1395–1402 Vol. 2, Beijing, China, 2005. doi: 10.1109/ICCV.2005.28.

[129] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104:249–257, 2006. ISSN 1077-3142. doi: 10.1016/j.cviu.2006.07.013.

[130] E. Bermejo Nievas, O. Deniz Suarez, G. Bueno García, and R. Sukthankar. Violence detection in video using computer vision techniques. In P. Real, D. Diaz-Pernil, H. Molina-Abril, A. Berciano, and W. Kropatsch, editors, *Computer Analysis of Images and Patterns. CAIP 2011*, volume 6855 of *Lecture Notes in Computer Science*. Springer, 2011. doi: 10.1007/978-3-642-23678-5_39.

[131] Qiqi Zhu, Yanfei Zhong, Bei Zhao, Gui-Song Xia, and Liangpei Zhang. Bag-of-visual-words scene classifier with local and global features for high spatial resolution remote sensing imagery. *IEEE Geoscience and Remote Sensing Letters*, 13(6):747–751, 2016. doi: 10.1109/LGRS.2015.2513443.

[132] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. URL http://arxiv.org/abs/1704.04861.

[133] 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. doi: 10.5121/ijdkp.2015.5201.

Appendix A
Data processing code

This appendix presents a Python code excerpt that describes a script for processing videos, extracting frames and creating batches of data. The *get_frames* function is responsible for getting frames from a video, resizing and normalizing them before adding them to a list of frames. The *create_batches* function then organizes these frames into batches, associating class labels (1 for violence and 0 for non-violence), and saves each batch as separate numpy files.

```
# Function to get video frames
def get_frames(video_path):

    frames_list = []

    # Read video file
    video_reader = cv2.VideoCapture(video_path)

    # Get the total number of frames in the video
    video_frames_count = int(video_reader.get(cv2.CAP_PROP_FRAME_COUNT))

    # Calculate the distance between each frame
    skip_frames_window = max(int(video_frames_count/FRAMES_NUMBER), 1)

    repeat_last_frame = False

    # If the total number of frames is less than FRAMES_NUMBER, repeat the
        last frame
    if video_frames_count < FRAMES_NUMBER:
        repeat_last_frame = True
        skip_frames_window = 1

    # Iterate over frames
    for frame_counter in range(FRAMES_NUMBER):

        # Change the current position
        if not repeat_last_frame or frame_counter < video_frames_count:
            video_reader.set(cv2.CAP_PROP_POS_FRAMES, frame_counter *
                skip_frames_window)
        else:
            # Repeat the last frame
            video_reader.set(cv2.CAP_PROP_POS_FRAMES, (video_frames_count -
                1))

        # Read the frame
        success, frame = video_reader.read()
```

```
34
35          if not success:
36              break
37
38          # Change the frame resolution to the chosen one
39          resized_frame = cv2.resize(frame, (FRAME_HEIGHT, FRAME_WIDTH))
40
41          # Normalizes the pixel values of the frame
42          normalized_frame = resized_frame / 255
43
44          # Add to the list of frames
45          frames_list.append(normalized_frame)
46
47      video_reader.release()
```

Listing A.1: Code excerpt to get video frames.

```
# Function to create batches and save each batch in a numpy file
def create_batches():

    # Get a list of the names of the videos in a class
    files_list_violence = os.listdir(os.path.join(DATASET_PATH, "Fight"))
    files_list_non_violence = os.listdir(os.path.join(DATASET_PATH, "
        NonFight"))

    # Iterate over the range of video samples, with a step of BATCH_SIZE//2
    for i in range(0, VIDEOS_NUMBER, BATCH_SIZE//2):
        # Select a subset of "Fight" videos for this batch
        sample_violence = files_list_violence[i : (i + BATCH_SIZE//2)]
        # Select a subset of "NonFight" videos for this batch
        sample_non_violence = files_list_non_violence[i : (i + BATCH_SIZE
            //2)]

        batch_frames = []
        batch_labels = []

        # Process each video in the "Fight" subset
        for j in sample_violence:
            batch_frames.append(get_frames(DATASET_PATH + "Fight/" + j))
            batch_labels.append(1)

        # Process each video in the "NonFight" subset
        for j in sample_non_violence:
            batch_frames.append(get_frames(DATASET_PATH + "NonFight/" + j))
            batch_labels.append(0)

        # Save the batch in a numpy file
        np.save(f"batchesTest/batch_{i//(BATCH_SIZE//2)}_features.npy",
            batch_frames)
        np.save(f"batchesTest/batch_{i//(BATCH_SIZE//2)}_labels.npy",
            batch_labels)
```

Listing A.2: Code excerpt to create the batches.

Appendix B
Training process code

This Python code snippet iteratively trains a neural network for video classification using batches of pre-processed frames. During each training epoch, the batches are loaded and subjected to data augmentation before being used to update the model weights with *train_on_batch*. After training each batch, the loss and accuracy metrics are recorded and, at the end of the epoch, the averages of these metrics are calculated for the training and test set. The model with the best weights in terms of test accuracy is saved periodically, and at the end of the process all the metrics are stored for later analysis.

```
# Iterate over the epochs
for epoch in range(EPOCHS):
    print(f"Epoch {epoch+1}/{EPOCHS}")

    # Variables to store data about the epochs
    epoch_losses_train = []
    epoch_accuracies_train = []

    # Shuffle the order of the batch indexes
    batch_indices = list(range(55))
    random.shuffle(batch_indices)

    for i in batch_indices:
        # Path to the features and labels file
        batch_features_path = f"batches/batch_{i}_features.npy"
        batch_labels_path = f"batches/batch_{i}_labels.npy"

        # Check if the files exist
        if os.path.exists(batch_features_path) and os.path.exists(
            batch_labels_path):
            # Load batches of features and labels
            batch_features = np.load(batch_features_path)
            batch_labels = np.load(batch_labels_path)
            # Apply data augmentation to each image
            augmented_images = []

            for video in batch_features:
                # Generates a set of random transformations
                params = datagen.get_random_transform((128, 128, 3))
                # Applies the same transformation to all the images in the
                    video
                augmented_video = [datagen.apply_transform(image, params)
                    for image in video]
                augmented_images.append(augmented_video)
```

```
                del augmented_video

            batch_features = np.array(augmented_images)
            del augmented_images

            # Obtain the shuffle index
            indices = list(range(len(batch_features)))
            random.shuffle(indices)

            # Shuffling features and labels
            batch_features = batch_features[indices]
            batch_labels = batch_labels[indices]

            # Train the model on batch
            loss_train, accuracy_train = model.train_on_batch(
                batch_features, batch_labels)

            # Save the loss and accuracy
            epoch_losses_train.append(loss_train)
            epoch_accuracies_train.append(accuracy_train)

            # Clearing memory
            del indices
            del batch_features
            del batch_labels
            del loss_train
            del accuracy_train

        else:
            print(f"Files for batch {i} not found.")

    # Calculate this epoch's average loss and accuracy
    mean_loss_train = np.mean(epoch_losses_train)
    mean_accuracy_train = np.mean(epoch_accuracies_train)
    print(f"Train Loss: {mean_loss_train}, Train Accuracy: {
        mean_accuracy_train}")

    # Save loss and accuracy averages
    losses_train.append(mean_loss_train)
    accuracies_train.append(mean_accuracy_train)
    del mean_loss_train
    del mean_accuracy_train

    # Variables to store data about the epochs
    epoch_losses_test = []
    epoch_accuracies_test = []

    for i in range(55, 62):
        # Path to the features and labels file
        batch_features_path = f"batches/batch_{i}_features.npy"
        batch_labels_path = f"batches/batch_{i}_labels.npy"
        accuracy_test = 0

        # Check if the files exist
        if os.path.exists(batch_features_path) and os.path.exists(
            batch_labels_path):
            # Load features and labels of the batch
            batch_features = np.load(batch_features_path)
```

```
            batch_labels = np.load(batch_labels_path)

            # Calculate the loss and accuracy in the test set
            loss_test, accuracy_test = model.evaluate(batch_features,
                batch_labels, verbose=0)

            # Save the loss and accuracy
            epoch_losses_test.append(loss_test)
            epoch_accuracies_test.append(accuracy_test)

            # Clearing memory
            del batch_features
            del batch_labels
            del loss_test
            del accuracy_test

        else:
            print(f"Files for batch {i} not found.")

    # Calculate this epochs's average loss and accuracy in the test set
    mean_loss_test = np.mean(epoch_losses_test)
    mean_accuracy_test = np.mean(epoch_accuracies_test)

    print(f"Test Loss: {mean_loss_test}, Test Accuracy: {mean_accuracy_test
        }")

    # Save loss and accuracy averages
    losses_test.append(mean_loss_test)
    accuracies_test.append(mean_accuracy_test)

    if mean_accuracy_test >= best_accuracy:
        # Save the trained model with the best weights
        model.save(f"{model_name}/best_weights_model_v{version_number}.h5")
        best_accuracy = mean_accuracy_test

    del mean_loss_test
    del mean_accuracy_test

    # Save the last trained model
    model.save(f"{model_name}/trained_model_v{version_number}.h5")

    # Save loss and accuracy data
    np.save(f"{model_name}/losses_train_v{version_number}.npy",
        losses_train)
    np.save(f"{model_name}/accuracies_train_v{version_number}.npy",
        accuracies_train)
    np.save(f"{model_name}/accuracies_test_v{version_number}.npy",
        accuracies_test)
    np.save(f"{model_name}/losses_test_v{version_number}.npy", losses_test)

    # Then the accuracy and loss over the epochs is ploted
```

Listing B.1: Code excerpt for the training process

Appendix C
Explainability analysis code

This Python code snippet provides a comprehensive approach to generating and visualizing **Grad-CAM** heat maps for understanding model predictions in video classification tasks. Initially, the code loads a pre-trained neural network and extracts frames from a video, which are then resized and processed for **Grad-CAM** analysis. For each frame, the *get_grad_cam* function computes the heatmap, highlighting the regions of the frame that influence the model's decision-making process. These heatmaps are then overlaid onto the original frames using the *apply_heatmap* function, producing visually interpretable images that emphasize areas of interest such as impactful movements or actions. Finally, the processed frames, including their corresponding heatmaps, are saved. This systematic approach ensures that the interpretability of the model's predictions is effectively demonstrated, providing valuable insights into the model's decision-making criteria.

```
# Generates the heat map using Grad-CAM for a specific layer in the
    internal model
def get_grad_cam(inner_model, image, layer_name):
    target_layer = inner_model.get_layer(layer_name)
    grad_model = tf.keras.models.Model(
        inputs=[inner_model.input],
        outputs=[target_layer.output, inner_model.output]
    )

    with tf.GradientTape() as tape:
        conv_outputs, predictions = grad_model(image, training=False)
        class_channel = predictions
        grads = tape.gradient(class_channel, conv_outputs)
        pooled_grads = tf.reduce_mean(grads, axis=(0, 1, 2))

    conv_outputs = conv_outputs[0]
    heatmap = np.mean(conv_outputs * pooled_grads, axis=-1)
    heatmap = np.maximum(heatmap, 0)

    if np.max(heatmap) > 0:
        heatmap /= np.max(heatmap)
    else:
        heatmap = np.zeros_like(heatmap)

    return heatmap

# Applies the heat map to the original frame and returns the resulting
```

```
    image
def apply_heatmap(heatmap, frame):
    heatmap = cv2.GaussianBlur(heatmap, (15, 15), 0)
    heatmap = cv2.resize(heatmap, (frame.shape[1], frame.shape[0]))
    heatmap = np.uint8(255 * heatmap)
    heatmap = cv2.applyColorMap(heatmap, cv2.COLORMAP_JET)
    superimposed_frame = heatmap * 0.4 + frame
    return np.uint8(superimposed_frame)

# Gets the internal model within the TimeDistributed layer
def get_inner_model(time_distributed_layer):
    if hasattr(time_distributed_layer, 'layer'):
        return time_distributed_layer.layer
    else:
        raise AttributeError("The TimeDistributed layer does not have a '
            layer' attribute".)

# Path to the trained model
model_path = MODEL_PATH

# Load the model
model = tf.keras.models.load_model(model_path)

# Get the TimeDistributed layer
time_distributed_layer = model.get_layer("time_distributed")

# Get the MobileNet network
inner_model = get_inner_model(time_distributed_layer)

# Path to the input video
video_path = VIDEO_PATH
cap = cv2.VideoCapture(video_path)
frames = []

FRAME_WIDTH, FRAME_HEIGHT = 128, 128
MAX_FRAMES = 32

# Get the total number of frames in the video
total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

# Calculate the indexes of the frames to be extracted
indices = np.linspace(0, total_frames - 1, MAX_FRAMES, dtype=int)

# Read and resize the equally spaced frames of the video
for idx in indices:
    cap.set(cv2.CAP_PROP_POS_FRAMES, idx)
    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame, (FRAME_WIDTH, FRAME_HEIGHT))
    frames.append(frame)

cap.release()
frames = np.array(frames)

# Creating the directory to save the frames with the heat map
output_dir = OUTPUT_DIR
os.makedirs(output_dir, exist_ok=True)
```

```
83
84 # Generate and apply the heat map for each frame of the video
85 for i, frame in enumerate(frames):
86     frame_input = np.expand_dims(frame, axis=0)
87     heatmap = get_grad_cam(inner_model, frame_input, layer_name="
       block_16_expand_relu")
88     superimposed_frame = apply_heatmap(heatmap, frame)
89     output_file = os.path.join(output_dir, f"heatmap_frame_{i}.png")
90     cv2.imwrite(output_file, superimposed_frame)
```

Listing C.1: Code excerpt for the explainability analysis