Universidade Estadual de Campinas
Instituto de Computação

Felipe Faria de Souza

Detection of Violent Events in Video Sequences Based on the Census Transform Histogram Operator

Detecção de Eventos Violentos em Sequências de Vídeos Baseada no Operador Histograma da Transformada Census

CAMPINAS
2020

Felipe Faria de Souza

Detection of Violent Events in Video Sequences Based on the Census Transform Histogram Operator

Detecção de Eventos Violentos em Sequências de Vídeos Baseada no Operador Histograma da Transformada Census

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

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

Supervisor/Orientador: Prof. Dr. Hélio Pedrini

Este exemplar corresponde à versão final da Dissertação defendida por Felipe Faria de Souza e orientada pelo Prof. Dr. Hélio Pedrini.

CAMPINAS
2020

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

So89d

Souza, Felipe Faria de, 1986-
Detection of violent events in video sequences based on the census transform histogram operator / Felipe Faria de Souza. – Campinas, SP : [s.n.], 2020.

Orientador: Hélio Pedrini.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

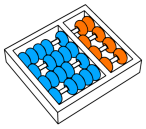
1. Videovigilância. 2. Descrição de eventos (Computação). 3. Visão por computador. 4. Aprendizado de máquina. I. Pedrini, Hélio, 1963-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Detecção de eventos violentos em sequências de vídeos baseada no operador histograma da transformada census
Palavras-chave em inglês:
Video surveillance
Events description (Computer science)
Computer vision
Machine learning
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Hélio Pedrini [Orientador]
Alexandre Gonçalves Silva
André Santanchè
Data de defesa: 06-08-2020
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0001-6391-5978
- Currículo Lattes do autor: http://lattes.cnpq.br/5101972851001549

Universidade Estadual de Campinas
Instituto de Computação



Felipe Faria de Souza

Detection of Violent Events in Video Sequences Based on the Census Transform Histogram Operator

Detecção de Eventos Violentos em Sequências de Vídeos Baseada no Operador Histograma da Transformada Census

Banca Examinadora:

- Prof. Dr. Hélio Pedrini
  Universidade Estadual de Campinas
- Prof. Dr. Alexandre Gonçalves Silva
  Universidade Federal de Santa Catarina
- Prof. Dr. André Santanchè
  Universidade Estadual de Campinas

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 06 de agosto de 2020

Resumo

Sistemas de vigilância em sequências de vídeo têm sido amplamente utilizados para o monitoramento de cenas em diversos ambientes, tais como aeroportos, bancos, escolas, indústrias, estações de ônibus e trens, rodovias e lojas. Devido à grande quantidade de informação obtida pelas câmeras de vigilância, o uso de inspeção visual por operadores de câmera se torna uma tarefa cansativa e sujeita a falhas, além de consumir muito tempo. Um desafio é o desenvolvimento de sistemas inteligentes de vigilância capazes de analisar longas sequências de vídeos capturadas por uma rede de câmeras de modo a identificar um determinado comportamento. Neste trabalho, foram propostas e avaliadas diversas técnicas de classificação, tendo como base o operador CENTRIST (Histograma da Transformada Census), no contexto de identificação de eventos violentos em cenas de vídeo. Adicionalmente, foram avaliados outros descritores tradicionais, como HoG (Histograma de Gradientes Orientados), HOF (Histograma do Fluxo Óptico) e descritores extraídos a partir de modelos de aprendizado de máquina profundo pré-treinados. De modo a permitir a avaliação apenas em regiões de interesse presentes nos quadros dos vídeos, técnicas para remoção do fundo da cena. Uma abordagem baseada em janela deslizante foi utilizada para avaliar regiões menores da cena em combinação com um critério de votação. A janela deslizante é então aplicada juntamente com uma filtragem de blocos utilizando fluxo óptico da cena. Para demonstrar a efetividade de nosso método para discriminar violência em cenas de multidões, os resultados obtidos foram comparados com outras abordagens disponíveis na literatura em duas bases de dados públicas (*Violence in Crowds* e *Hockey Fights*). A eficácia da combinação entre CENTRIST e HoG foi demonstrada em comparação com a utilização desses operadores individualmente. A combinação desses operadores obteve aproximadamente 88% contra 81% utilizando apenas HoG e 86% utilizando CENTRIST. A partir do refinamento do método proposto, foi identificado que avaliar blocos do quadro com a abordagem de janela deslizante tornou o método mais eficaz. Técnicas para geração de palavras visuais com codificação esparsa, medida de distância com um modelo de misturas Gaussianas e medida de distância entre agrupamentos também foram avaliadas e discutidas. Além disso, também foi avaliado calcular dinamicamente o limiar de votação, o que trouxe resultados melhores em alguns casos. Finalmente, formas de restringir os atores presentes nas cenas utilizando fluxo óptico foram analisadas. Utilizando o método de Otsu para calcular o limiar do fluxo óptico da cena a eficiência supera nossos resultados mais competitivos: 91,46% de acurácia para a base *Violence in Crowds* e 92,79% para a base *Hockey Fights*.

Abstract

Surveillance systems in video sequences have been widely used to monitor scenes in various environments, such as airports, banks, schools, industries, bus and train stations, highways and stores. Due to the large amount of information obtained via surveillance cameras, the use of visual inspection by camera operators becomes a task subject to fatigue and failure, in addition to consuming a lot of time. One challenge is the development of intelligent surveillance systems capable of analyzing long video sequences captured by a network of cameras in order to identify a certain behavior. In this work, we propose and analyze the use of several classification techniques, based on the CENTRIST (Transformation Census Histogram) operator, in the context of identifying violent events in video scenes. Additionally, we evaluated other traditional descriptors, such as HoG (Oriented Gradient Histogram), HOF (Optical Flow Histogram) and descriptors extracted from pre-trained deep machine learning models. In order to allow the evaluation only in regions of interest present in the video frames, we investigated techniques for removing the background from the scene. A sliding window-based approach was used to assess smaller regions of the scene in combination with a voting criterion. The sliding window is then applied along with block filtering using the optical flow of the scene. To demonstrate the effectiveness of our method for discriminating violence in crowd scenes, we compared the results to other approaches available in the literature in two public databases (Violence in Crowds and Hockey Fights). The combination of CENTRIST and HoG was demonstrated in comparison to the use of these operators individually. The combination of both operators obtained approximately 88% against 81% using only HoG and 86% using CENTRIST. From the refinement of the proposed method, we identified that evaluating blocks of the frame with the sliding window-based approach made the method more effective. Techniques for generating a codebook with sparse coding, distance measurement with a Gaussian mixture model and distance measurement between clusters were evaluated and discussed. Also we dynamically calculate the threshold for class voting, which obtained superior results in some cases. Finally, strategies for restricting the actors present in the scenes using optical flow were analyzed. By using the Otsu's method to calculate the threshold from the optical flow at the scene, the effectiveness surpasses our most competitive results: 91.46% accuracy for the Violence in Crowds dataset and 92.79% for the Hockey Fights dataset.

List of Figures

List of Tables

List of Abbreviations

2D	Two Dimensional
3D	Three Dimensional
AdaBoost	Adaptive Boosting
AutoThresh	Automatic Threshold Selection
BoW	Bag-of-Words
CCTV	Closed-circuit television
CENTRIST	Census Transform Histogram
CNN	Convolutional Neural Network
ConvNet	Convolutional Network
FPS	Frames per Second
GB	Gradient Boosting
GLCM	Gray Level Co-occurrence Matrix
GMM	Gaussian Mixture Models
GPU	Graphic Processing Unit
HEq	Histogram Equalization
HMDB51	Human Motion DataBase
HMOF	Histogram of Magnitude of the Optical Flow
HOF	Histogram of Optical Flow
HoG	Histogram of Gradients
HOOF	Histogram of Orientation of the Optical Flow
H.O.T.	High Order Terms
HOT	Histogram of Tracklets
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IWLD	Improved Weber Local Descriptor
LBP	Local Binary Pattern
LBPCM	Local Binary Pattern Co-Occurrence Matrix
LDA	Latent Dirichlet Allocation
LHOF	Local Histogram of Optical Flow
LHoG	Local Histogram of Gradients
LSTM	Long Short-Term Memory
MBH	Motion Boundary Histogram
MDT	Mixture of Dynamic Textures
MHI	Motion History Image
MoG	Mixture of Gaussians
moIWLD	Motion Improved Weber Local Descriptor
moSIFT	Motion Space Interest Points
moWLD	Motion Weber Local Descriptor
OViF	Oriented Violent Flows
PCA	Principal Component Analysis

RBF	Radial Basis Function
RF	Random Forest
RGB	Red-Green-Blue
RNN	Recurrent Neural Networks
SD	Substantial Derivative
SF	Social Force
SGD	Stochastic Gradient Descent
SIFT	Scale-Invariant Feature Transform
SRC	Sparse-Representation based Classification
STIP	Space Time Interest Points
SVM	Support Vector Machine
TRoF	Temporal Robust Features
UCF101	Action Recognition Dataset - University of Central Florida
UNICAMP	University of Campinas
ViF	Violent Flows
XGB	Extreme Gradient Boosting
WLD	Weber Local Descriptor

Contents

Chapter 1

Introduction

In this chapter, we present the problem investigated in this work, its associated research challenges, the main goals and contributions, some research questions, the publication list, as well as the text organization.

1.1 Problem Description

Video surveillance is an active research field whose main purpose is to monitor activities or behavior by means of electronic equipment, for instance, closed-circuit television (CCTV) cameras. The identification of anomalies in the scenes is a challenging task due to several factors, such as illumination conditions, low-resolution cameras, and occlusions.

Human action recognition [15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25] via video processing has been widely used in diverse domains, for instance, crime prevention, smart homes, health monitoring, human-computer interaction, among others. In particular, detection of violent scenes has received substantial interest in the last years.

For violence detection in crowded scenes, surveillance cameras are normally used for detection on dense populations. The task of generating meaningful description from scenes and actions can be considered quite complex due to the large number of possible events that involve crowd of people, although few of them could be considered aggressive or violent.

1.2 Research Challenges

Due to the advances in digital video technology, large volumes of data have been acquired, stored and transmitted, which makes it impracticable to verify their content by human operators. Such demand promotes the research and development of automatic video analysis systems to deal with massive amounts of videos in a fast and scalable way.

The task of analyzing and detecting abnormal patterns in video sequences relies on the amount of people present in the scene, presence of occlusion, difficulty in removing the background, camera motion, and resolution of the video sequences. Moreover, it is also dependent on the domain context, as the definition of normality and abnormality

is particular to each application. Such task is very challenging and demanding, even for human operators.

1.3 Objectives and Contributions

In this work, we describe and analyze methods for detecting violent events in video sequences. The process aims to evaluate a video sequence at high level of abstraction, without considering the position of people in the scene to identify abnormalities.

The recognition process consists of five main stages. Initially, the video sequences are pre-processed in order to improve the perception about objects in the scene, where operations to reduce the influence of lighting changes are applied. Then, the Census Transform Histogram (CENTRIST) descriptor [26] is used to extract a set of features from the video frames. The dimensionality from the extracted features is also tested in order to reduce redundant or noisy information without losing the most representative characteristics. Finally, the video frames are classified as violent or non-violent.

In order to evaluate the suitability of CENTRIST-based descriptors for violence detection on crowded scenes, we analyze the performance of the CENTRIST descriptor using several pre-processing filters and different classifiers. Furthermore, in addition to CENTRIST, HOG descriptor [27] is also combined to CENTRIST using two different approaches.

The first strategy is to concatenate both HoG and CENTRIST descriptors in only one vector. The second strategy applies the CENTRIST descriptor to the output matrix obtained with HoG descriptor. We also investigate alternatives to evaluate the scene using such descriptor. To refine our method, we evaluate a sliding window strategy for extracting features through small blocks of the frame. In addition, coding techniques are applied between the feature extraction and training steps. Then, we evaluate approaches based on optical flow to discriminate the regions of interest in the scene whose actors would be performing the action. Finally, we analyze how to dynamically calculate a voting threshold based on the value that maximizes the accuracy in the test set.

In order to validate our methodology, two public datasets are used. The results obtained with the proposed method are compared to other approaches available in the literature. As main contribution, we aim to achieve competitive recognition accuracy rates without demanding high computational resources.

1.4 Research Questions

In this work, we aim to answer the following research questions on violence detection highly focusing on crowd scenes:

- Are holistic techniques effective to detect violence, especially in crowd scenes?
- How does CENTRIST descriptor perform to detect violence?
- Which approaches using CENTRIST could improve its effectiveness?

- Can the combination of CENTRIST and other descriptors improve violence detection?

1.5 Publication

The following paper [28] was derived from the development of this research work:

- F. Souza, H. Pedrini. *Detection of Violent Events in Video Sequences based on Census Transform Histogram.* Conference on Graphics, Patterns and Images (XXX SIBGRAPI). Niterói-RJ, Brazil, pp. 323-329, October 17-20, 2017.

1.6 Text Organization

The remainder of this dissertation is organized as follows. Section 2 briefly describes some important concepts related to the topic under investigation. Section 3 presents the methodology proposed in this work, describing the pre-processing, the feature extraction, the feature reduction, as well as the classification process. Section 4 describes and analyzes the experimental results using CENTRIST descriptor against other approaches. Among them, HoG descriptor was used for direct comparison to other handcrafted descriptors. Techniques based on deep learning, such as Inception-based models, were also used to extract deep features used as feature vector for training. After some initial observations, we explore variations and combinations to extract more discriminative features. Section 5 includes some final remarks and directions for future work.

Chapter 2

Background

In this chapter, we describe some relevant concepts and works related to the topic under investigation.

2.1 Violent Actions

Research in the action recognition field [15, 16, 17, 18] has advanced significantly over the last decades. Experiments performed in early works were conducted on datasets containing simple actions performed by a single individual. Recent research focuses on more realistic scenarios, in particular, for crowded scenes.

Two datasets, named Violence in Crowds [13] and Hockey Fights [14], have been largely used by the community for evaluating violent event detection methods through the exploration of different visual features, such as texture, color, shape and motion [10, 13, 29, 30].

2.2 Video Surveillance

In the context of video-surveillance systems, there is a great interest in providing some automatic annotation of video archives. The solutions proposed to address this problem inherit from the techniques first designed for the goal of image classification and content retrieval. Local descriptors developed to describe image patches [31, 32], spatio-temporal information [33, 34] and motion clues [35] have been used to extract invariant features for videos, such as spatio-temporal interest points (STIP) [36, 37]. Some techniques address temporal motion model involving trajectories [35, 38, 39, 40, 41, 42, 43, 44].

To produce a single vector representing the video, the majority of the methods generate and aggregate large sets of local descriptors that enable the use of discriminative classifiers. Finally, to evaluate these local features, techniques derived from bag-of-words [45] are used as codebook techniques. In our work, we address some of these techniques, as described in Section 2.6.

Due to the combination of encoding techniques and the mentioned local descriptors, simple human actions can be successfully identified in a controlled environment. Such methods are also considered for the detection of actions in real movies and video clips.

2.2.1 Motion Analysis

In addition to shape, motion is definitely one of the most reliable sources of information for action recognition. However, it inevitably involves the background or camera motion when dealing with uncontrolled and realistic situations. Despite the progress, the existing descriptors suffer from incomplete handling of motion in the video sequence. Although some attempts have been made to compensate camera motion in several ways [1,32,35,43,46,47] to discriminate motion action caused by the camera, the task of compensating motion into video description remains an open question.

Jain et al. [1] built an action recognition framework incorporating motion clues. They separated dominant motion and the residual motion and tried to take into account the impact of camera movement and independent actions. The 2D parametric motion model describes the global (or dominant) motion between successive frames. Initially, the dominant motion is estimated and employed to separate the dominant flow from the optical flow. Then, kinematic features are introduced and used to obtain a more comprehensive description of visual motion. Thus, dense trajectories are presented to compute local descriptors as an action recognition approach. Figure 2.1 depicts the results before (left) and after (right) the process to suppress the camera interference, compensating for the camera motion to more appropriately evaluate events in the scene.

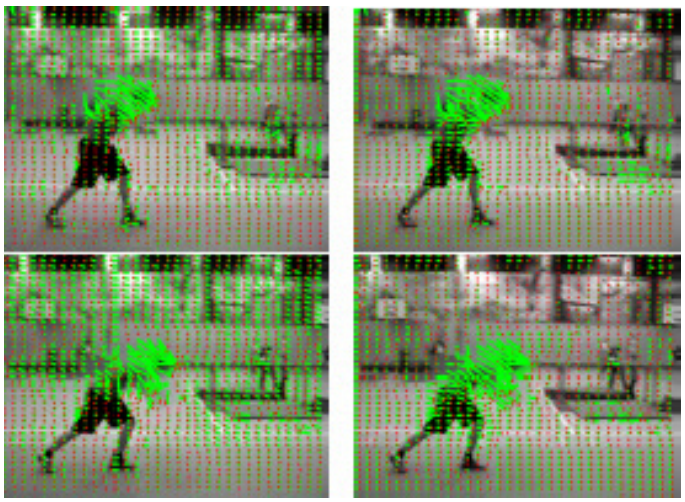


Figure 2.1: Optical flow field vectors (green vectors with red end points) before (left) and after (right) dominant motion compensation. Most flow vectors due to camera motion are suppressed after compensation. Source: Images extracted from Jain et al. [1].

.

Wang et al. [35] demonstrated that when local descriptors are computed over dense trajectories, the performance improves considerably compared to when scene is calculate over spatio-temporal features [34].

Most approaches to person identification via motion focus on the problems of recognizing humans beings by observing gait through the movement of human legs and arms [48, 49, 50, 51]. Laptev et al. [52] assumed that similar patterns of motion contain similar events with consistent motion in the image sequences. They demonstrated that local-temporal image descriptors can be defined to carry important information from space-time events

for subsequent recognition.

2.3 Image Descriptors

In this section, we provide an introduction and a brief explanation of the image descriptors adopted in this research work.

2.3.1 Census Transform Histogram

As mentioned previously, the holistic descriptor known as CENTRIST (Census Transform Histogram) [53,54] is employed to extract features from video frames due to their properties for encoding structural information while suppressing detailed textural content. It models the distribution of local structures and geometrical information through spatial descriptors.

CENTRIST compares the intensity value of the center pixel with its eight neighbors, as shown in Equation 2.1. If the intensity of a neighbor pixel is lower than or equal to the intensity of the center pixel, value 1 is assigned to its cell; otherwise, value 0 is assigned. Then, all Census Transform values are concatenated and converted to an unsigned 8-bit integer. CENTRIST descriptor can be compared to obtaining histograms from the LBP descriptor [55] when using 8 neighbors with distance 1. In sequence, all values are combined into a histogram vector with 256 bins that represents the appearance frequency of the Census Transform. Finally this histogram is normalized in order to turn the comparison independent from the image size.

$$\begin{array}{|c|c|c|} \hline 32 & 64 & 96 \\ \hline 32 & 64 & 96 \\ \hline 32 & 32 & 96 \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|} \hline 1 & 1 & 0 \\ \hline 1 & & 0 \\ \hline 1 & 1 & 0 \\ \hline \end{array} \Rightarrow (11010110)_2 \Rightarrow (214)_{10} \qquad (2.1)$$

2.3.2 Histogram of Oriented Gradients

Histogram of Oriented Gradients [56], or simply HoG, is a well-known descriptor for human detection in images, such as in video scenes [27]. This technique counts occurrences of gradient orientation in parts of an image. Although the method is similar to other descriptors, such as Edge Orientation Histograms [57], Scale-Invariant Feature Transform (SIFT) [58], and Shape Contexts [59], it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

The HoG descriptor has some advantages: for instance, it operates in local cells and is invariant to geometric and photometric transformations, except for object orientation. Such changes would appear only in larger spatial regions. To calculate a HoG descriptor [2, 56], it is necessary to compute the horizontal and vertical gradients, since we need to generate a histogram of gradients. This is easily achieved by filtering the image with the following kernels $[-1, 0, 1]$ and $[-1, 0, 1]^T$, equivalent to applying the Sobel operator with kernel size 1. Subsequently, we can find the magnitude and direction of gradient using the following

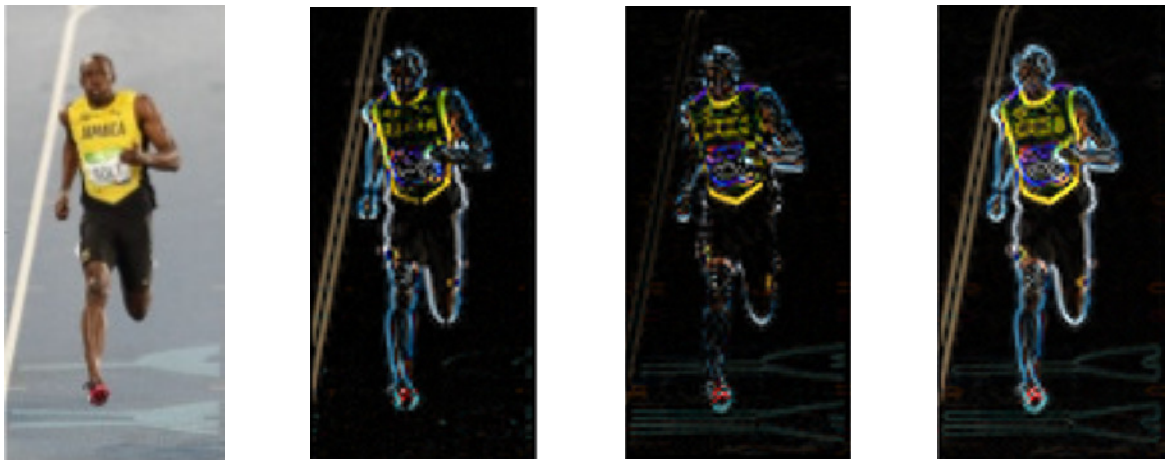
expression:

$$g = \sqrt{g_x^2 + g_y^2} \tag{2.2}$$

$$\theta = \arctan \frac{g_y}{g_x} \tag{2.3}$$

The x-gradient increases on vertical lines and the y-gradient increases on horizontal lines. The magnitude of the gradient raises whenever there is a sharp change in intensity. None of them fire when the region is smooth. The gradient image removes a lot of non-essential information (for instance, the constant colored background), but outlines highlighted.

At each pixel, the gradient has a magnitude and a direction. For color images, the gradients of the three channels are evaluated (as shown in Figure 2.2). The magnitude of the gradient in a pixel is the maximum of the magnitude of gradients of the three channels, whereas the direction is the angle corresponding to the maximum gradient.



(a) (b) (c) (d)

Figure 2.2: (a) Source image. (b) gradient extraction: absolute value of x-gradient. (c) gradient extraction: absolute value of y-gradient. (d) gradient extraction: magnitude of the gradient. Source: Images modified from Histogram of Oriented Gradients [2].

To calculate the Histogram of Gradients in $n \times n$ cells, the image is divided into $n \times n$ cells and a histogram of gradients is calculated for each $n \times n$ cell (Figure 2.3). One of the important reasons for using a feature descriptor to represent an image patch is that it provides a compact representation. For example, for 8×8 cells, an 8×8 image patch contains 8×8×3=192 pixel values. The gradient of this patch contains 2 values (magnitude and direction) per pixel, which adds up to 8×8×2=128 numbers.

In the end, these 128 numbers are represented using a 9-bin histogram, which can be stored as an array of 9 numbers. Not only is the representation more compact, the calculation of a histogram on a patch makes this representation more robust to noise. Individual gradients may have noise, but a histogram over an 8×8 patch makes the representation much less sensitive to noise [2].

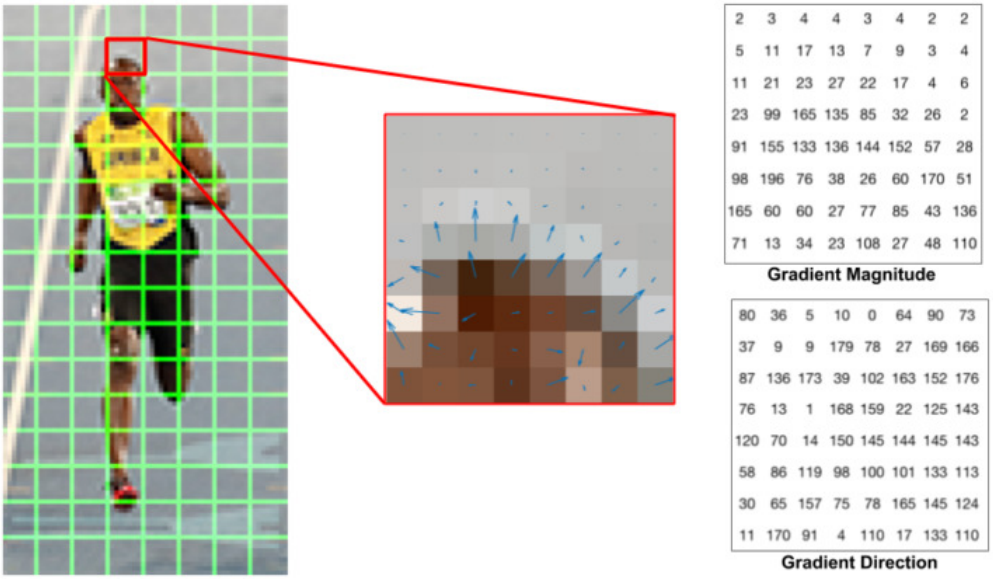


Figure 2.3: Left: source image. Center: RGB patch and gradients represented using arrows. Right: gradients in the same patch represented as numbers. Source: Images extracted from Histogram of Oriented Gradients [2].

In our work we make usage of the implementation provided by OpenCV library [60] (version 2.4.13) using the following parameters: Number of orientation bins = 8, Size (in pixels) of a cell = (16, 16), Number of cells in each block = (1, 1), Block normalization method = 'L2-Hys'.

2.3.3 Gabor Filter

A Gabor Filter [61] is a linear filter used for texture analysis, which means that it basically analyzes whether there is any specific frequency content in the image in specific directions in a region located around the point or region under analysis. Gabor filters can serve as proper band-pass filters for one-dimensional signals. A complex Gabor filter is defined as the product of a Gaussian kernel multiplied by a complex sinusoid [62].

Frequency and orientation representations of Gabor filters are similar to those of the human visual system and have been considered particularly suitable for texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. When a Gabor filter is applied to an image, it provides the highest response at edges and at points where texture changes.

Gabor filters with different frequencies and orientations in different directions have been used to localize and extract text-only regions from complex document images [63, 64], for facial expression recognition [65] and have also been widely used in pattern analysis applications [66]. For instance, the Gabor space is very useful in image processing applications such as optical character recognition, iris recognition (in the case of Daugman's algorithms [67]) and fingerprint recognition. For example, the effects of Gabor filter can be observed in the second image of Figure 2.4.

The Gabor Filter, expressed in Equation 2.4, has several parameters that affect the

(a) original image (b) after Gabor filtering

Figure 2.4: Images before and after applying Gabor filters. Source: Images extracted from Gabor Filter: A practical overview [3].

source image in different aspects

$$g(x, y; k, \sigma, \theta, \lambda, \gamma, \psi, ktype) = \exp\left(-\frac{x^2 + \gamma^2 y'^2}{2\sigma^2}\right) \exp\left(i\left(2\pi\left(\frac{x'}{\gamma}\right) + \psi\right)\right) \tag{2.4}$$

where

1. k: is the size of the Gabor kernel. If $k = a, b$, we then have a Gabor kernel of size $a \times b$ pixels. As with many other convolution kernels, k is preferably odd and the kernel is a square (just for the sake of uniformity).
2. σ: is the standard deviation of the Gaussian function used in the Gabor filter and controls the width of the Gaussian envelope used in the Gabor kernel.
3. θ: is the orientation of the normal to the parallel stripes of the Gabor function. This is perhaps one of the most important parameters of the Gabor filter. It decides what type of features the filter responds to. For instance, assigning theta a value of zero means that the filter is responsive only to horizontal features. Therefore, to obtain features at various angles in an image, we divide the interval between 0 and 180 into 16 equal parts and compute a Gabor kernel for each value of theta thus obtained. In our implementation, we chose exactly 16 splits, because that was the default value in the OpenCV [60] implementation. These parameter values could be modified to suit specific purposes.
4. λ: is the wavelength of the sinusoidal factor in Equation 2.4.
5. γ: is the spatial aspect ratio. Gamma controls the ellipticity of the Gaussian. When $\gamma = 1$, the Gaussian envelope is circular.
6. ψ: is the phase offset.
7. *ktype*: indicates the type and range of values that each pixel in the Gabor kernel can hold.

In our work, we employ the implementation provided by OpenCV library 2.4.13 [60] using the following parameters: $\sigma = 4.0$, $\theta = 16$, $k = 31$, $\lambda = 10$, $\psi = 0.5$, $\gamma = 0$ and *ktype*=cv2.CV_32F.

2.3.4 Histogram of Optical Flow

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene.

Sequences of ordered images, such as in Figure 2.5, allow the estimation of motion as either instantaneous image velocities or discrete image displacements [4]. The optical flow methods aim to calculate the motion between two video frames that are taken at times t and $t + \Delta t$ at every voxel position. These methods are called differential, as they are based on local Taylor series approximations of the image signal; that is, they use partial derivatives with respect to the spatial and temporal coordinates.

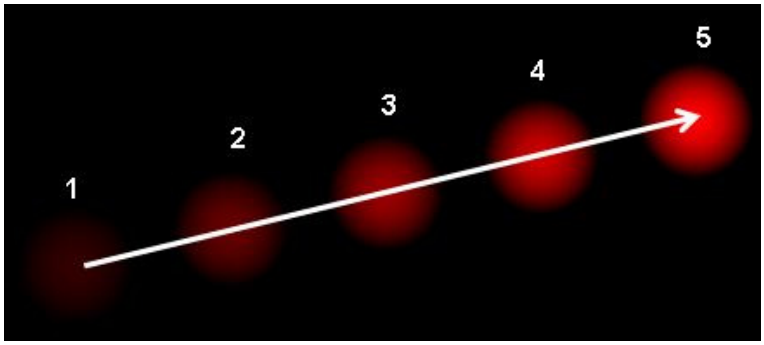


Figure 2.5: Ball moving in 5 consecutive frames. The arrow shows its displacement vector. Source: Images extracted from Beauchemin et al. [4].

Fleet and Weiss [68] provided a tutorial introduction to gradient-based optical flow. Barron et al. [69] provided a performance analysis of various optical flow techniques. It emphasizes the accuracy and density of measurements.

For a $2D + t$ dimensional case (3D or n-D cases are similar), a voxel at location (x, y, t) with intensity $I(x, y, t)$ will be moved by Δx, Δy and Δt between the two image frames, and the following brightness constancy constraint can be expressed as:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{2.5}$$

Assuming the movement is small, the image constraint at $I(x, y, t)$ with Taylor series can be developed to obtain:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \text{H.O.T.} \tag{2.6}$$

where $\Delta x, \Delta y$ are the x and y components of the velocity or optical flow of $I(x, y, t)$ and $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$ are the derivatives of the image at (x, y, t) in the corresponding directions, and H.O.T. are high order terms. I_x, I_y and I_t can be written for the following derivatives.

For processing dense optical flow, the method developed by Lucas and Kanade [70] has been widely used. By combining information from several nearby pixels, it is often possible to resolve the inherent ambiguity of the optical flow equation. It is also less sensitive to image noise than point-wise methods. On the other hand, since it is a purely local method,

it cannot provide flow information within uniform regions of the image.

$$\begin{bmatrix} u(x,y,t) \\ v(x,y,t) \end{bmatrix} = \begin{bmatrix} (\sum_i f_{x_i})^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & (\sum_i f_{y_i})^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix} \tag{2.7}$$

This method assumes that the displacement of the image contents between two nearby instants (frames) is small and approximately constant in a neighborhood of the point p under consideration. Thus, the optical flow equation can be assumed to hold for all pixels within a window centered at p. Thus, Lucas-Kanade method [70] takes a 3×3 patch around the point. All the 9 points have the same motion. We can find (f_x, f_y, f_t) for these 9 points. The problem is now reduced to solve 9 equations with two unknown variables that are over-determined. A better solution is obtained with least square fitting method. In Equation 2.7, it is the final solution that is a two equation-two unknown problem.

After evaluating the optical flow, a histogram is calculated from the output matrix. In this case, it is possible to use the magnitude (HOF), such as orientation (HOOF). Histogram of optical flow orientation, proposed by Dalal et al. [71], is introduced as a descriptor that encodes the moving information of each video frame.

The histogram of the optical flow has been used in action recognition [71, 72] and abnormal event detection in video streams [35].

2.3.5 Background Subtraction using Mixture of Gaussians

The background subtraction method using Mixture of Gaussians (MoG) has been widely used for foreground detection. KaewTraKulPong and Bowden [73] presented an adaptive mixture model of background scene for real-time tracking of moving objects. They used different equations in distinct phases and this allowed their system to learn faster and more accurately, as well as adapting effectively to changing environments. In addition, a shadow detection scheme was also introduced based on a computational color space that uses their background model.

In our work, we use the implementation of *BackgroundSubtractorMOG* [74] provided by OpenCV library version 2.4.13, using its default parameters: length of the history = 3, number of Gaussian mixtures = 5, background ratio = 0.001, noise strength sigma = 0.

2.4 Otsu's Method

Otsu's method [75] is a non-parametric and unsupervised method of automatic threshold selection for image segmentation. An optimal threshold is selected by the discriminant criterion to maximize the separability of the resulting classes in gray levels. The procedure is very simple, utilizing only the zero- and first-order cumulative moments of the gray-level histogram [75].

Otsu's method was cited by Sezgin et al. [76] to automatically perform the clustering-based image thresholding, or reduction of a gray-level image to a binary image. The algorithm assumes that the image contains two classes of pixels following a bi-modal his-

togram (foreground pixels and background pixels), then calculates the optimum threshold that separates the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal [75]. Consequently, Otsu's method is approximately a one-dimensional, discrete analog of Fisher's Discriminant Analysis [77]. Otsu's method is also directly related to the Jenks's optimization method [78].

In Otsu's method, an exhaustive search for the threshold that minimizes the intra-class variance (the variance within the class), defined as a weighted sum of variances of the two classes:

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{2.8}$$

Weights ω_0 and ω_1 are the probabilities of the two classes separated by a threshold t t, and σ_0^2 and σ_1^2 are variances of these two classes.

The class probability $\omega_{0,1}(t)$ is computed from the L bins of the histogram:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i) \qquad \omega_1(t) = \sum_{i=t}^{L-1} p(i) \tag{2.9}$$

Otsu showed that minimizing the intra-class variance is the same as maximizing inter-class variance [75]:

$$\begin{aligned} \sigma_b^2(t) &= \sigma^2 - \sigma_w^2(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \\ &= \omega_0(t)\omega_1(t)\left[\mu_0(t) - \mu_1(t)\right]^2 \end{aligned} \tag{2.10}$$

which is expressed in terms of class probabilities ω_0, ω_1 and class means μ_0, μ_1 while the class mean $\mu_{0,1,T}(t)\mu_{0,1,T}(t)$ is:

$$\mu_0(t) = \sum_{i=0}^{t-1} i\frac{p(i)}{\omega_0} \qquad \mu_1(t) = \sum_{i=t}^{L-1} i\frac{p(i)}{\omega_1} \qquad \mu_T = \sum_{i=0}^{L-1} ip(i) \tag{2.11}$$

The following relations can be easily verified as:

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T \qquad \omega_0 + \omega_1 = 1 \tag{2.12}$$

The class probabilities and class means can be computed iteratively. This idea yields an effective algorithm. To compute the histogram and probabilities of each intensity level, the next steps must be followed: (i) set up initial $\omega_i(0)$ and $\mu_i(0)$; (ii) step through all possible thresholds $t = 1, \ldots$, maximum intensity; (iii) update ω_i and μ_i; (iv) compute $\sigma_b^2(t)$; (v) the desired threshold corresponds to the maximum $\sigma_b^2(t)$. [79]

2.5 Classifiers

In our work, we evaluate several classifiers to achieve high accuracy rates. In the following subsections, we present some potential classification approaches to identify violent actions in video scenes.

2.5.1 SVM Classifier

Support vector machines (SVMs) [80] are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

An SVM model is a representation of the examples as points in a mapped space, so that examples can be categorically divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall [81].

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick [82], implicitly mapping their inputs into high-dimensional feature spaces.

More formally, an SVM constructs a hyperplane or set of hyperplanes in a high (or infinite)-dimensional space [80], which can be used for classification, regression and other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (known as functional margin), since, in general, the larger the margin, the lower the generalization error of the classifier [81].

In our work, we mainly use the implementation provided by Scikit-Learn library versions 0.17.1 and 0.18.2 [83]. During the experiments to refine our results, we apply grid search with the following variations of $\gamma = [10^{-2}, 10^{-1}, 1, 10, 10^2]$ and $C = [1, 10, 50, 10^2]$ for RBF kernel, and $C = [1, 10, 50, 10^2]$ for linear kernel.

2.5.2 Stochastic Gradient Descent

Stochastic gradient descent (SGD) learning is a regularized linear model classifier [84]. The gradient of the loss is estimated for each sample at a time and the model is updated along the way with a decreasing strength schedule (known as learning rate).

For best results using the default learning rate schedule, the data should have zero mean and unit variance. This implementation works with data represented as dense or sparse arrays of floating-point values for the features [84]. n-stochastic (or on-line) gradient descent, the true gradient of $Q(w)$ is approximated by a gradient in a single example:

$$w = w - \eta \nabla Q_i(w) \tag{2.13}$$

As the algorithm sweeps through the training set, it performs the update shown in Equation 2.13 for each training example. Several steps can be done in the training set until the algorithm converges. If this is done, the data can be shuffled for each pass to avoid cycles. The model loss parameter can be controlled and typical implementations can use an adaptive learning rate for the algorithm to converge. In most implementations, by default, it uses a linear support vector machine (SVM) for discriminative learning, such as the implementation *sklearn.svm.SVC* provided by Scikit-Learn library [83] and used in our work. Other parameters used in this implementation are: default loss function = 'hinge', penalty = l2, maximum number of passes over the training data = 1000, stopping

criterion = $10e^{-3}$, epsilon = 10.

2.5.3 Random Forests

Random forests [85] are a combination of tree predictors so that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The combination of the results makes the model more robust with respect to noise.

A random forest classifier consists of a collection of tree-structured classifiers $h(x, k), k = 1, ...,$ where k are independent identically distributed random vectors and each tree casts a unit vote for the most popular class in input x. The strength of the individual trees in the forest and their correlation directly impacts the final error [85].

In our work, we use the *sklearn.ensemble.RandomForestClassifier* implementation provided by Scikit-Learn library version 0.18.2 [83] using its default parameters: number of decision trees = 10, maximum depth = 2, function to measure the quality of a split = Entropy.

2.5.4 AdaBoost

The AdaBoost classifier, short for Adaptive Boosting, is a machine learning meta-algorithm formulated by Schapire and Freund [86]. It can be used in conjunction with many other types of learning algorithms to improve their performance. The output of the other learning algorithms (known as weak learners) is combined into a weighted sum that represents the final output of the boosted classifier.

AdaBoost is adaptive in the sense that subsequent weak learners that combines several minor results in favor of a global stronger result. Moreover, in some problems, it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner [86].

In the training step, AdaBoost [86] refers to a specific training method of a boosted classifier. A boost classifier is a classifier that takes the results from a weak learner as input and returns a value indicating the class of the object. For example, in the two class problem, the sign of the weak learner output identifies the predicted object class and the absolute value gives the confidence in that classification [87]. Similarly, the T-th classifier will be positive if the sample is believed to be in the positive class and negative otherwise. Finally, it is used voting criteria to determine which is more likely class.

In our work, we use the *sklearn.ensemble.AdaBoostClassifier* implementation, named AdaBoost SAMME [88], provided by Scikit-Learn library [83], as well as the following parameters: base estimator = Decision Tree using max_depth=1, number of estimators = 50, learning rate = 1.

2.5.5 Extreme Gradient Boosting

The Extreme Gradient Boosting (XGB) [89] is an optimized distributed gradient boosting system designed to be highly efficient, flexible and portable. It implements machine

learning algorithms under the Gradient Boosting framework. XGB also provides a parallel tree boosting. According to the author of the algorithm, what makes XGBoost a unique boosting approach is that it uses a more regularized model formalization to control over-fitting, which gives it better performance [89] and, consequently, helps to reduce overfitting.

In our work, we use the implementation called XGBoost Python package, provided by XGBoost library version 1.0.0 [90] using its default parameters: maximum depth = 3, learning rate = 0.1, number of estimators = 100, objective function = 'binary:logistic', and booster function = 'gbtree'.

2.6 Codebook Techniques

In this section, we present some techniques used in this work to obtain a more representative feature space from the data.

2.6.1 K-Means Classifier

K-means clustering [91] is a method of vector quantization [92], originally from signal processing field, popular in cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

In our work, we use the *sklearn.cluster.k_means* implementation provided by Scikit-Learn library version 0.16.2 [83] with the following parameters: method for initialization = 'k-means++' (selects initial cluster centers for K-means clustering in a smart way to speed up convergence), maximum number of iterations = 100.

2.6.2 Gaussian Mixture Models

A Gaussian Mixture Model (GMM) [93] is a probabilistic model that assumes that all data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. Mixture models can be thought of as generalizing K-means clustering to incorporate information about the covariance structure of the data, as well as the centers of the latent Gaussian [94].

To select the number of components in a classical Gaussian Mixture Model, the Bayesian Information Criterion (BIC) [95] can be used efficiently. In theory, it gathers the number of components only in the asymptotic regime (for instance, if much data is available and assuming that the data was actually generated independent and identically distributed from a mixture of Gaussian distribution) [94].

In our work, we use the *sklearn.mixture.GMM* implementation provided by Scikit-Learn library version 0.16.2 [83] with the following parameters: covariance type = 'full' (each component has its own general covariance matrix, parameters updated in the initialization process = 'wmc' (combination of 'w' for weights, 'm' for means, and 'c' for covariances), convergence threshold = $1e^{-3}$, number of iterations = 200.

2.6.3 Sparse Coding

The purpose of sparse coding [96] is to find a set of basis vectors ϕ_i such that we can represent an input vector $\mathbf{x}$ as a linear combination of these basis vectors:

$$\mathbf{x} = \sum_{i=1}^{k} a_i \mathbf{d}_i = a\mathbf{D} \tag{2.14}$$

It learns an over-complete set of basis vectors to represent input vectors $\mathbf{x} \in \mathbb{R}^n$ (that is, such that $k > n$). Thus the basis vectors are better able to capture structures and patterns inherent in the input data. In the process, the following steps are performed:

- Learning:
  - Given training data $\mathbf{X}$.
  - Learn dictionary $\mathbf{D}$ [97] and sparse code $\mathbf{a}$ [96].
- Encoding:
  - Given test data $\mathbf{x}$, dictionary $\mathbf{D}$.
  - Learn sparse code $\mathbf{a}$ [96].

To implement sparse coding technique [98], we use two methods provided by Scikit-Learn library version 0.18.2 [83]. The former, *sklearn.decomposition.MiniBatchDictionaryLearning* corresponds, to the implementation for "online dictionary learning for sparse coding" [97]. The latter, *sklearn.decomposition.sparse_encode*, is used to find a sparse array code [96] regarding the dictionary defined in the previous step.

2.7 Related Work

In this section, we initially introduce some approaches related to anomaly detection, action recognition, and abnormal crowd behavior. Then, we describe relevant results from the literature achieved in the datasets on which our work is focused.

2.7.1 Anomaly Detection

Li et al. [5] proposed a detector based on the video representation that accounts for both appearance and dynamics by using a set of mixture of dynamic textures (MDT) models. These models implement a center surround discriminant saliency detector that produces spatial saliency scores, and a model of normal behavior that is learned from training data and produces temporal saliency scores.

Spatial and temporal anomaly maps are then defined at multiple spatial scales, by considering the scores of these operators at progressively larger regions of support. The multi-scale scores act as potentials of a conditional random field that guarantees global consistency of the anomaly judgments [5].

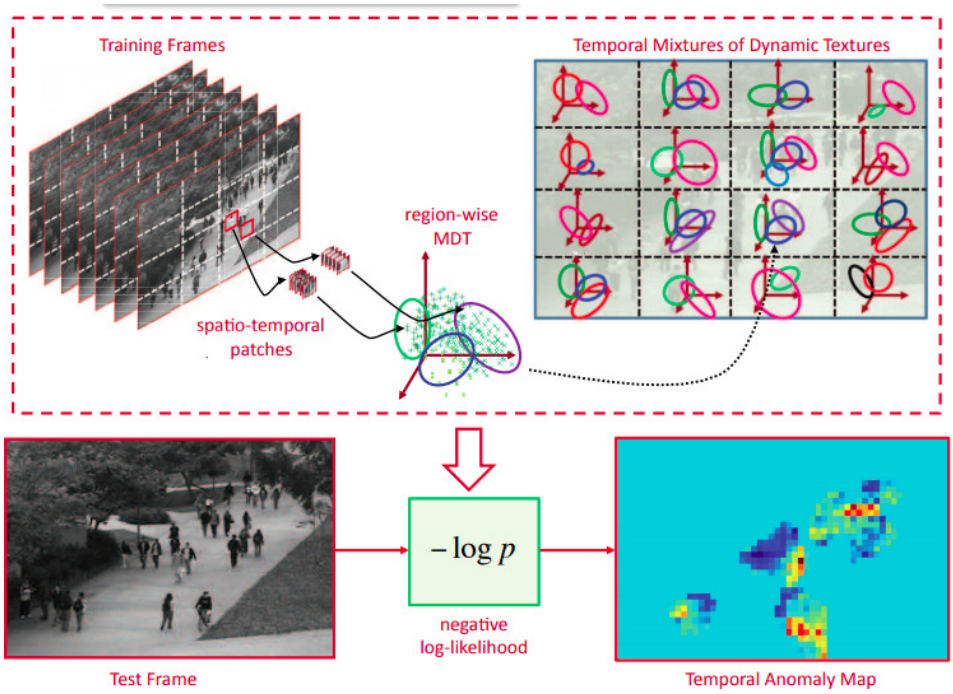


Figure 2.6: An MDT is learned per scene subregion, at training time. A temporal anomaly map is produced by measuring the negative log probability of each video patch under the MDT of the corresponding region. Source: Image extracted from Li et al. [5].

2.7.2 Abnormal Crowd Behavior

Cao et al. [99] employed a method of abnormal crowd behavior detection based on spatio-temporal LBP-weighted social force model. This algorithm integrates the time domain characteristics and regional information contained in the spectral characteristics of spatio-temporal LBP sequence into the social force model, becoming the crowd model more accurate.

Yin et al. [100] used the parameter of population density to make full use of the crowd density characteristics and dynamic characteristics. It was proposed a novel method by increasing the dimension of feature vector to increase the information content in order to improve the recognition accuracy. Crowd dynamic and crowd density information are combined to form a higher dimension of feature vectors, which is named as the crowd behavior feature vector to improve the robustness of the algorithm. The authors utilize Local Binary Pattern Co-Occurrence Matrix (LBPCM) for crowd density estimation to ensure the excellent accuracy. At the same time, they adopts high accuracy optical flow histograms of the orientation of interaction force to extract the crowd dynamic information (HOIF). Finally, an SVM is adopted to detect the abnormal events using the crowd behavior feature vector.

Sousa [101] proposed the CENTRIST3D descriptor, a spatio-temporal variation of the CENTRIST descriptor. The method created a histogram of spatio-temporal features from successive frames by extracting histograms of the volumetric Census Transform from a spatial representation using a modified spatial pyramid matching algorithm. The approach was evaluated on different datasets, focusing on detecting anomaly, violence and actions.

2.7.3 Spatio-Temporal / Audio Features

A method for violent scene identification was proposed by Nam et al. [102], where a spatio-temporal dynamic activity, an audio-visual flame detector, and a blood-detector were approached as feature descriptors for violence detection. In their work, they attempted to detect violence using audio and color matching criteria.

Giannakopoulos et al. [103] combined 7 audio features with 1 visual feature to detect violent videos on video sharing sites. Bilinski et al. [104] proposed an extension of the Improved Fisher Vectors (IFV) that allows representing a video using both local features and their spatio-temporal positions. Using the sliding window approach to violence detection, they reformulated the IFV and used the summed area table data structure to speed up the method. They showed that the proposed improvements make the violence recognition more accurate compared to the standard IFV, the IFV with spatio-temporal grid, and other state-of-the-art methods and make the violence detection significantly faster.

2.7.4 Spatio Temporal Interest Points (STIP)

Nievas et al. [14] applied two spatio-temporal descriptors, Space-Time Interest Points (STIP) and Motion scale-invariant feature transform (MoSIFT) in combination with bag-of-words to discriminate local image features and provide a more compact representation for the patterns.

The authors estimated the spatio-temporal extents of the detected events and computed their scale-invariant spatio-temporal descriptors. Using such descriptors, the method classifies events and constructs video representation in terms of labeled space-time points. In the context of human motion analysis, the method allows for detection of walking people in scenes with occlusions and dynamic backgrounds [14].

2.7.5 Violent Flows Descriptor (ViF)

Hassner et al. [13] proposed a representation, named Violent Flows (ViF) descriptor, for real-time crowd violence detection.

The authors described a novel approach to real-time detection of breaking violence in crowded scenes. The method considers statistics of how flow-vector magnitudes change over time. These statistics, collected for short frame sequences, are represented using the Violent Flows descriptor. ViF descriptors are then classified as either violent or non-violent using linear SVM. Magnitudes of the optical flow are used to model the frequencies of the ViF words as bag-of-features [13].

2.7.6 Oriented Violent Flows Descriptor (OViF)

Gao et al. [29] described a feature descriptor, called Oriented VIolent Flows (OViF), which explores information of motion magnitude changes in statistical motion orientations.

In their work, the combination of features using AdaBoost and linear SVM have achieved high accuracy rate on the Violent Flows benchmark.

2.7.7 Histogram of Tracklets (HOT)

Mousavi et al. [6] considered orientation and magnitude in a two-dimensional histogram to encode the motion patterns expected in each cuboid. The method classifies frames as normal and abnormal by using Latent Dirichlet Allocation (LDA) and Support Vector Machines (SVM).

The term tracklets is used to name compact spatio-temporal representations of moving rigid objects (Figure 2.7). They represent fragments of an entire trajectory of individual point movement, generated by frame-wise association between point localization results in the neighboring frames. They captured the evolution of patches that were originally introduced to describe human motions for action recognition proposals. In their work, the effectiveness of tracklets using HOT descriptors for abnormal behavior understanding is explored in crowded scenarios [6].

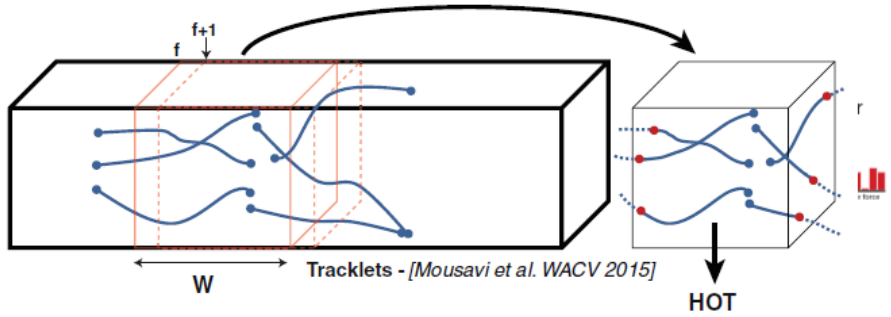


Figure 2.7: Tracklets represent fragments of an entire trajectory of individual point movement. Source: Images extracted from Mousavi et al. [6].

2.7.8 Substantial Derivative and Fluid Dynamics

Mohammadi et al. [7] presented a novel video descriptor based on an important concept in fluid mechanics that captures the rate of change of a fluid property as it travels through a velocity field.

Unlike standard approaches that only use temporal motion information, their descriptor explores the spatio-temporal characteristic of substantial derivative (SD). In particular, the spatial and temporal motion patterns are captured by convective and local accelerations, respectively [7].

After estimating the convective and local field from the optical flow, the standard bag-of-word procedure for each motion pattern separately is followed, and the two resulting histograms are concatenated to form the final descriptor. Figure 2.8 illustrates the derivative descriptor proposed by Mohammadi et al. [7], where the optical flow is extracted from the input picture (left) and used to calculate the local convective forces (center). Then, a standard bag-of-words method is executed separately for local and convective forces. Finally, local and convective forces are combined for the selected patches region and used as descriptors (right).

Wang et al. [8] proposed a modified algorithm based on the combination between a streakline model based on fluid dynamics and an abnormal behavior detection method

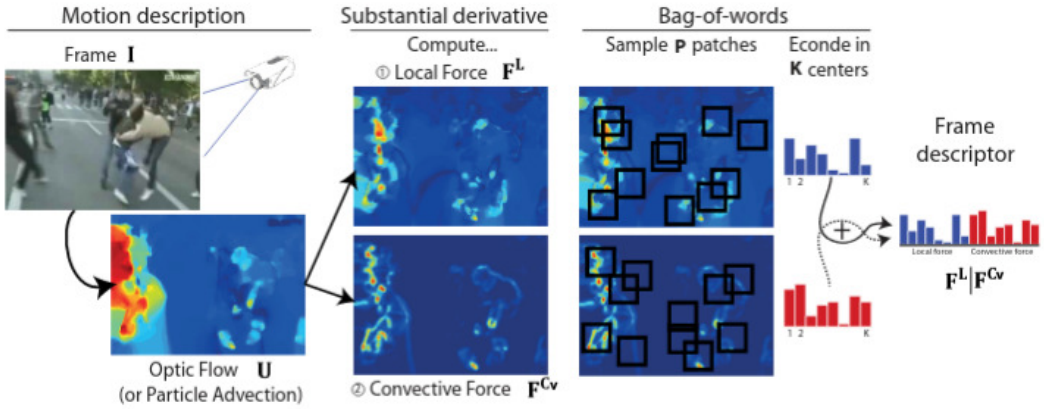


Figure 2.8: Summary of the social force approach to abnormal behavior detection in crowd videos. Source: Images extracted from Mohammadi et al. [7].

presented by Hassner et al. [13] in order to improve the recognition accuracy of abnormal crowd behavior. In Figure 2.9, we give an example of how the streaklines behave. To calculate them, it is assumed to have a particle position $(x_i(t), y_i(i))$ at time t. Then, the particle position through point p at any time instant t is computed, as described in Equation 2.15.

$$x_i^p(t+1) = x_i^p(t) + u(x_i^p(t), y_i^p(t), t) \tag{2.15}$$
$$y_i^p(t+1) = y_i^p(t) + v(x_i^p(t), y_i^p(t), t) \tag{2.16}$$

where u and v are optical flow field.

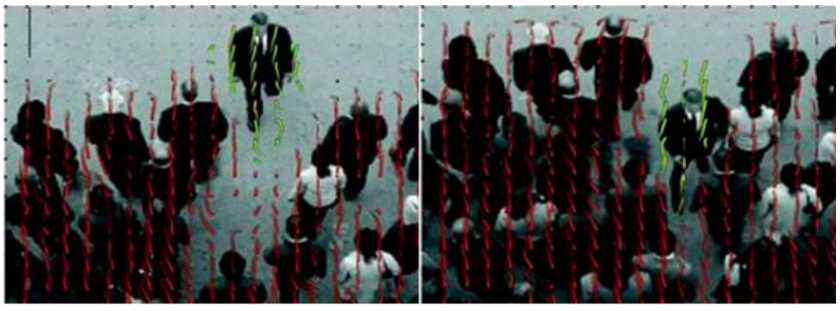


Figure 2.9: Streakline visualization. Source: Images extracted from Wang et al. [8].

2.7.9 Social Force (SF)

Mehran et al. [9] detected abnormal behavior in crowd videos using Social Force (SF) model. For their proposal, a grid of particles is placed over the image and space-time average of optical flow is used to evaluate the content.

By treating the moving particles as individuals, their interaction forces are estimated using the social force model. Figure 2.10 illustrates the social force descriptor proposed by Mehran et al. [9]. They incorporated a holistic approach to analyzing videos of crowds using the particle advection. The interaction force is mapped onto the image plane to obtain the

Force Flow for every pixel in each frame. They evaluated the change of interaction forces over time to determine the ongoing behavior of the crowd. Spatio-temporal volumes of Force Flow are randomly selected and used to model the normal behavior of the crowd in a bag-of-word approach [9].

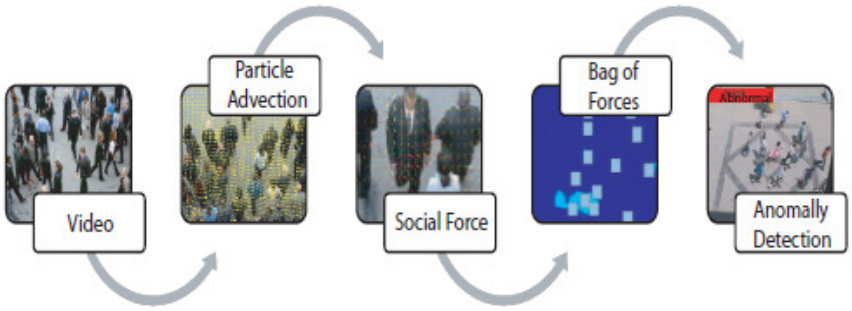


Figure 2.10: Summary of the social force approach to abnormal behavior detection in crowd videos. Source: Images extracted from Mehran et al. [9].

2.7.10 Holistic Features

Marsden et al. [10] presented an approach to detect anomaly in crowd behavior that uses a set of efficiently computed scene-level holistic features. Two features from the literature are combined: crowd collectiveness and crowd conflict, with two developed crowd features: mean motion speed and a new formulation of crowd density.

Figure 2.11 illustrates the descriptor proposed by Marsden et al. [10], where the crowd density is estimated through the percentage of cells occupied in the scene. The collectiveness is analyzed by tracklet positions and velocities found in the current frame to build a weighted adjacency matrix. The edge weights in each matrix column are summed and averaged. This mean value corresponds to the overall collectiveness level for the current frame. Conflict is another holistic crowd property that can be defined as the level of friction/interaction between neighboring points. In their work, it is calculated by summing the velocity correlation between each pair of neighboring points tracked in a given frame [10].

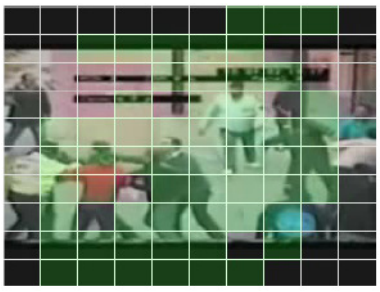


Figure 2.11: Crowd density calculation grid for a scene from the violent-flows dataset. Each green square corresponds to an occupied grid cell (crowd density in this frame = 57%). Source: Images extracted from Marsden et al. [10].

In their work, two strategies were used for recognition:

- when only normal training data is available, a GMM for outlier detection is used.
- when both normal and abnormal training data are available, an SVM for binary classification is used.

2.7.11 WLD-based Descriptors (WLD, MoWLD, IWLD)

Chen et al. [105] proposed a simple, but very powerful and robust local descriptor. It is inspired by Weber's Law, which is a psychological law and it consists of two components: differential excitation and orientation.

It states that the change of a stimulus (such as sound, lighting) that will be just noticeable is a constant ratio of the original stimulus. When the change is smaller than this constant ratio of the original stimulus, a human being would recognize it as a background noise rather than a valid signal.

Motivated by this point for a given pixel, the differential excitation component of the proposed Weber Local Descriptor (WLD) is computed based on the ratio between the two terms: one is the relative intensity differences of a current pixel against its neighbors.

WLD first computes the salient micro-patterns, such as differential excitation, and then builds statistics on these salient patterns along with the gradient orientation of the current point. Thus, Zhang et al. [106] used WLD approach to detect violence for video surveillance scenes.

Zhang et al. [107] proposed a novel Motion Weber Local Descriptor (MoWLD) based on the well-known WLD. They extended the WLD spatial descriptions by adding a temporal component to the appearance descriptor, which implicitly captures local motion information as well as low-level image appear information.

To eliminate redundant and irrelevant features, the non-parametric Kernel Density Estimation (KDE) is employed on the MoWLD descriptor. In order to obtain more discriminative features, it adopts the sparse coding and max pooling scheme to further process the selected MoWLDs.

Zhang et al. [11] described the Improved WLD (IWLD) and moIWLD. Two major improvements and a more effective algorithm for detecting violence from motion images were proposed. The first, the improved WLD (IWLD) to better depict low-level image appearance information and to extend the spatial IWLD descriptor by adding a temporal component to capture local motion information and hence form the MoIWLD.

The second one proposed a modified sparse-representation-based classification (SRC) model to both control the reconstruction error of coding coefficients and minimize the classification error [11]. Based on the proposed sparse model, a class-specific dictionary containing dictionary is learned by using class labels of training samples. With this dictionary, besides representation residual also the representation coefficients become discriminative. A classification scheme integrating the modified sparse model is developed to explore such discriminative information:

- it controls the reconstruction error of coding coefficients and minimizes the classification error.

- based on the proposed sparse model, a class-specific dictionary containing dictionary atoms corresponding to the class labels is learned by using class labels of training samples.
- with this learned dictionary, not only the representation residual, but also the representation coefficients become discriminative.

In Figure 2.12, it is shown how the method proposed by Zhang et al. [11] performs the preprocessing step before extracting features in order to build the sparse representation and to produce the discriminative model that classify video scene.

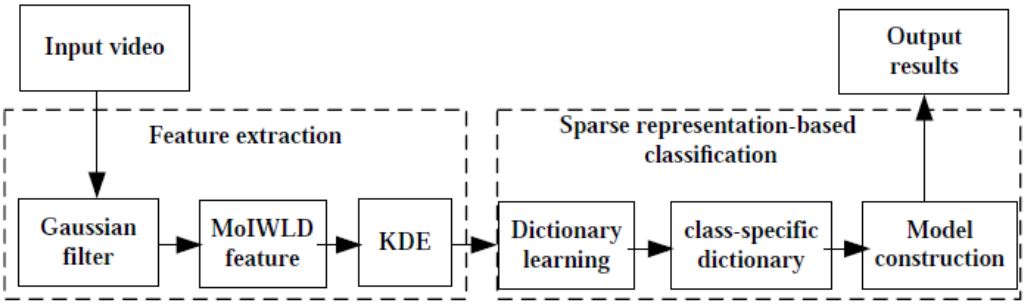


Figure 2.12: Method proposed by Zhang et al. [11].

2.7.12 Temporal Robust Features

Moreira et al. [108] explored a content description method for violence detection based on temporal robust features that grasp video sequences, automatically classifying violent videos. The method also generated promising results for fast and effective classification of other recognition tasks (for instance, pornography and other sensitive content).

When compared to more complex approaches to violence detection, their method shows similar classification quality, while being several times more efficient in terms of run-time and memory footprint. Their work employed a fast end-to-end Bag-of-Visual-Words (BoVW)-based framework for violence classification. They adapted Temporal Robust Features (TRoF) [109], a fast spatio-temporal interest point detector and descriptor, which is customized for sensitive content detection, such as violence [108].

2.7.13 Violence Detection in Surveillance Video using Low-Level Features

Zhou et al. [110] extracted low-level features from the motion regions after segmenting the motion regions. As, in general, it is very common that the torsos or legs in action videos are not visible due to the occlusion among people, the standard Histogram of Oriented Gradient (HoG) descriptor [56] cannot be considered effective for human detection. Aiming at this task, their work was proposed to represent the actions in videos using the features of Local Histogram of Oriented Gradient (LHoG).

To make full use of the temporal information, another descriptor (LHOF) is proposed to capture dynamic changes. LHoG and LHOF could independently describe parts of the person and extract meaningful information from partially occluded persons, which is suitable for violence detection. LHoG features are obtained from RGB images and LHOF features are captured from the motion magnitude images. Information from different modalities is complementary and expresses different characteristics of an action. An LHoG (or LHOF) descriptor is extracted from a "block", composed of "cells". The number and size of motion regions are different for different video clips, leading to different-length features.

Under the framework of BoW model, the extracted low-level features are represented as a fixed-length vector using a histogram which reflects the frequency of different words. The visual words in the BoW model are typically defined as the cluster centers which are obtained using K-means clustering method over the low-level features (LHoG or LHOF). The number of visual words could be set according to the practical application requirement. Intuitively, BoW approach collects the statistic information of the feature distributions. Thereon, the vectors with the same length could be further processed using a standard classifier.

Zhou et al. [110] evaluated that previous methods process features with BoW framework after fusing the features. For instance, MoWLD [107] is a long vector by directly combining HoG and HOF, followed by the BoW method. However, HoG and HOF features may not share the same class space, which will reduce the discriminative ability.

Unlike the previous early-fusion strategy, the authors applied a late-fusion for the extracted features. They argued that the class space of LHoG features is different from that of LHOF features, and based on these arguments, they used these features that are processed using the BoW model, resulting in two types of vectors with the same length. Then, the two types of vectors are combined before feeding into the classifier. Experimental results demonstrated that the late-fusion method outperformed the early-fusion method for the low-level features. In the classification stage, an SVM with a Radial Basis Function (RBF) kernel is used as the classifier to distinguish the violent video sequences.

Lohithashva et al. [111] proposed method that uses Local Binary Pattern (LBP) and GLCM (Gray Level Co-occurrence Matrix) as feature descriptors for the detection of a violent event. Prominent features are used with five different supervised classifiers and the results are fused when evaluating Hockey Fights and Violent in Crowds datasets. GLCM is a texture-based feature descriptor that can be used to extract spatial variation from the matrix. The GLCM texture features [112] provide 14 texture features measured from the probability matrix to extract the characteristics from the texture statistics of the frames. In their work, they used only four statistical properties: contrast, correlation, energy and homogeneity. Each element (x, y) in GLCM specifies the number of times that the pixel with value x occurred horizontally next to a pixel with value y. GLCM metrics were used to allow rotational invariance through a set of rotational parameter. Generally, 8 orientations separated $\pi/4$ radian aside, where N indicates the intensity value present in the frame. The properties (energy, correlation, contrast and homogeneity) are calculated using the normalized GLCM. The contrast property is used to measure local variations in GLCM [111].

2.7.14 Deep Learning Networks

Over the last decade, several approaches based on deep learning have been considerably increased in various research fields and yielding substantial gains in several benchmarks. The history of convolutional neural network design started with LeNet-style models [113], which were simple stacks of convolutions for feature extraction and max-pooling operations for spatial sub-sampling.

Since ImageNet competition [114], neural networks such AlexNet [115] have been successfully applied to a wide variety of computer vision tasks. Deep learning techniques have overcome traditional approaches in various visual detection and classification problems: object-detection [116], segmentation [117], human pose estimation [118, 119], video classification [120], object tracking [121], and super-resolution [122]. In the following paragraphs, we will provide a brief description of the techniques involved in the evolution of deep learning techniques.

The Inception deep convolutional architecture was introduced as GoogLeNet by Szegedy et al. [123], here referred to as Inception-v1. The idea behind the Inception module is to make this process easier and more efficient by explicitly factoring it into a series of operations that would independently analyze cross-channel correlations and spatial correlations [12]. More precisely, the typical Inception module first examines the cross-channel correlations via a set of 1×1 convolutions, mapping the input data into 3 or 4 separate spaces that are smaller than the original input space, and then maps all correlations in these smaller 3D spaces, through regular 3×3 or 5×5 convolutions [12].

Subsequently, the Inception architecture was refined in several ways. For instance, Inception-v2 introduced batch normalization [124], Inception-v3 received additional factorization ideas [125], and Inception-v4 used cheaper Inception blocks than the original Inception [126].

Each Inception block is followed by filter-expansion layer (1×1) convolution without activation) that is used for scaling up the dimensionality of the filter bank before the residual addition to match the depth of the input. This is necessary to compensate for the dimensionality reduction induced by the Inception block. Szegedy and Ioffef et al. [126] raised the question: "Are there any benefits to combining Inception [125] architectures with residual connections [127]". Thus, through Inception-v4, they provided empirical evidence that training with residual connections significantly speeds up training of the Inception networks. For ImageNet classification challenge, it achieved 3.08% for top-5 error in the test set.

In the work by Chollet al. [12], Xception modeled based on the fundamental building block of Inception-style models, which is the Inception module (shown in Figure 2.13). It considers a simplified version of an Inception module that uses only one convolution size (for instance, 3×3) and does not include an average pooling tower. After having reformulated it as a large 1×1 convolution followed by spatial convolutions that would operate on non-overlapping segments of the output channels (shown in Figure 2.14), a stronger hypothesis is assumed: cross-channel correlations and spatial correlations could be mapped completely separately [12].

An "extreme" version of an Inception module (shown in Figure 2.15) is based on

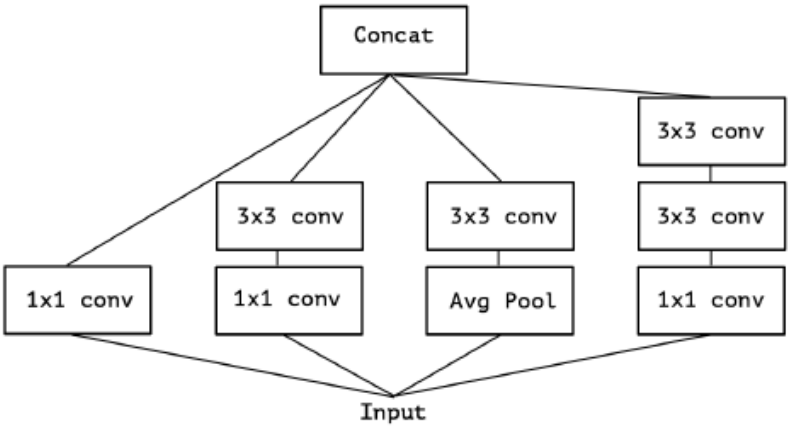


Figure 2.13: A canonical Inception module (Inception-v3). Source: Images extracted from Chollet et al. [12].

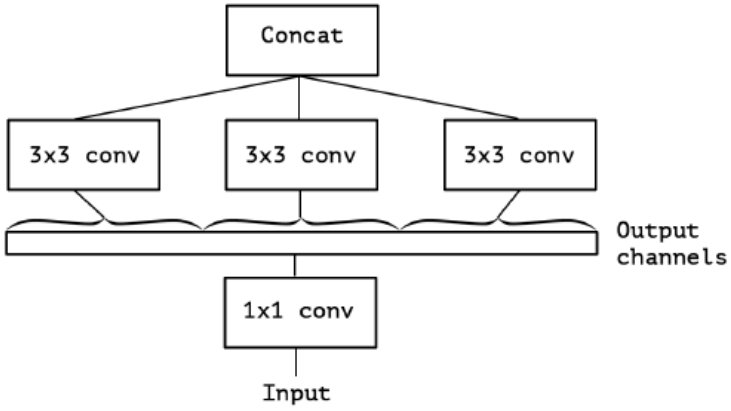


Figure 2.14: A strictly equivalent reformulation of the simplified Inception module. Source: Image extracted from Chollet et al. [12].

this stronger hypothesis, which would first use a 1×1 convolution to map cross-channel correlations and would then separately map the spatial correlations of every output channel. In short, the Xception architecture is a linear stack of depthwise separable convolution layers with residual connections. This makes the architecture very easy to define and modify, unlike architectures such as Inception-v2 or Inception-v3, which are much more complex to define [12].

Compared to Inception-v3, Xception shows little gain in classification performance on the ImageNet dataset and large gains on the JFT dataset. Figure 2.16 illustrates the results obtained by Chollet al. [12] that demonstrate the difference in performance between Xception and Inception-v3 using weights based on training with ImageNet dataset.

Although most of the deep learning approaches previously mentioned have gradually outperformed the benchmarks for several datasets, such as ImageNet, it is possible to observe that the results are highly dependent on the amount of data available for training. Furthermore, although the increase in model size and computational cost tends to translate into immediate quality gains for most tasks (as long as enough labeled data is provided for training), there is still a lack of data needed to train this model to achieve good results

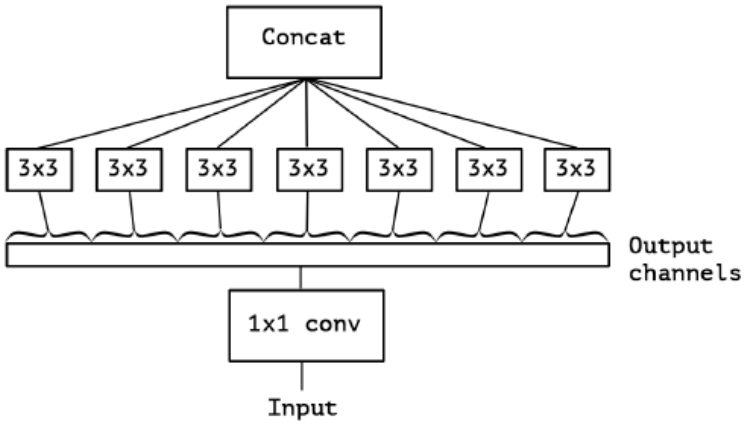


Figure 2.15: An "extreme" version of their Inception module, with one spatial convolution per output channel of the 1×1 convolution. Source: Image extracted from Chollet et al. [12].

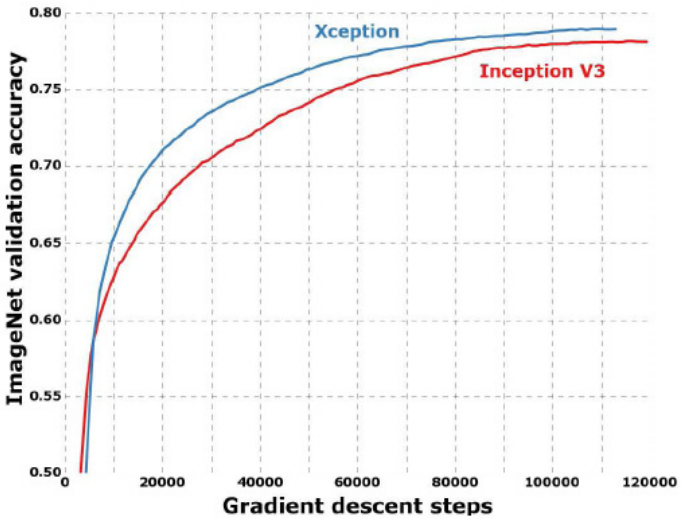


Figure 2.16: Accuracy comparison between Xception and Inception V3 on the ImageNet dataset, as the gradient descent steps evolve. Source: Image extracted from the work described by Chollet et al. [12].

in real-world scenarios. Therefore, computational efficiency and low parameter count are still essential factors for various use cases, such as mobile vision and big-data scenarios.

2.7.15 Deep Neural Network for Violent/Non-Violent Video Classification

Mondal et at. [128] proposed the construction of a frame-level descriptor by combining motion and shape features extracted from each frame and using a deep neural network architecture trained for video classification in violent and non-violent categories based on their video-level descriptor. They were inspired by the Weber Local Descriptor (WLD) [107] in relation to differential excitation and orientation, and so they restricted themselves to some salient interest points selected by Shi-Tomasi's corner point detection algorithm [129],

which finds the most appropriate interest points for tracking. Thus they reduced processing time and also unnecessary computation for most pixels in the frame belonging to the static background [128].

The optical flow for interest points is determined based on the Lucas-Kanade algorithm [70]. Then, they estimated the movement of interest points and extracted the magnitude and orientation of the movement. The frame-level descriptors of training videos are clustered into two groups using K-means clustering. The cluster for which majority descriptor corresponds to frames of violent sequences is the reference cluster for violent videos and the other one is the reference for non-violent videos. Vectors representing the center of the corresponding clusters are taken as the prototype vectors for violent and non-violent frame.

For classification, they proposed a deep neural network architecture with four fully connected hidden layers with 20, 15, 10 and 5 nodes and the input layer has the same number of nodes as the dimension of video-level descriptor. The neural network is trained using backpropagation algorithm [130] and uses stochastic gradient-based optimizer proposed by Kingma et al. [131].

2.7.16 Temporal Segment Networks for Action Recognition in Videos

Want et al. [132] presented a general and flexible video-level framework for learning action models in videos. This method, called temporal segment network (TSN), aims to model long-range temporal structures with a new segment-based sampling and aggregation module. This unique design allows TSN to efficiently learn action models using all action videos.

The authors adapted the model for action recognition in both trimmed and untrimmed videos with simple average pooling and multi-scale temporal window integration, respectively. They also studied a number of good practices that instantiate a temporal segment network framework given limited training samples.

Their approach obtained the state-the-of-art performance on four challenging action recognition benchmarks: HMDB51 [133](71.0%), UCF101 [134](94.9%), THUMOS14 [135](80.1%), and ActivityNet v1.2 [133](89.6%). Using the proposed RGB difference for motion models, their method has achieved competitive accuracy on UCF101 [134](91.0%) while running at 340 FPS. Furthermore, based on the temporal segment networks, it won the video classification track at the ActivityNet challenge in 2016 among 24 teams, which demonstrates the effectiveness of temporal segment network and the proposed good practices.

2.7.17 CNN + LSTM

To recognize a video as violent or non-violent, the network proposed by Ammar et al. [136] has the ability to encode localized spatial attributes and how they change over time. CNNs are capable of generating discriminating spatial features, however, for temporal encoding using Long Short Term Memory (LSTM), existing methods use the features extracted

from the fully connected layers. The output of the fully-connected layers represents the global descriptor of the entire image. Since existing methods do not encode the located spatial changes, they used methods that involve adding more data streams, such as optical flow images, resulting in increased computational complexity [136].

In the context of their work, the use of LSTM becomes relevant since it encodes the convolutional features of the CNN. In addition, the convolutional gates in the LSTM are trained to encode temporal changes from local regions. This allows the entire network to encode localized spatio-temporal characteristics.

Convolutional neural networks process information as separate data points. However, other types, such as Recurrent Neural Networks (RNN [137]) or LSTM networks [138], also have promising results by considering previous steps over time.

The main attraction of RNNs is to take in account a step back in time, in combination with the current input to determine how they respond to new data. Sequential information is preserved in the hidden state of the recurring network, which spans many steps as it falls forward, affecting each new input processing. This architecture can be very useful, as it is a sequence of events that happen over time. By taking advantage of sequential information, it can perform tasks that are impossible for feed-forward networks [136].

An LSTM network, instead of just using the information passed by one step in the past, learns through many steps of time, allowing them to link causes and effects that occur over an extended period of time [136]. This type of network can effectively use the temporal information of the video as input to interpret events that depend on a sequence of actions. In the case of violence detection, this type of neural network points to a hidden transitional state between a non-violent and a violent state, and the identification of this hidden state can be valuable information to predict whether there will be a violent scene before the violence action actually occurs.

Hanson et al. [139] proposed a Bidirectional Convolutional LSTM architecture, called Spatiotemporal Encoder, which includes bidirectional temporal encoding and elementwise max pooling. This approach takes advantage of temporal knowledge to decide whether the current frame is still in the previous state or moves to a different one.

2.7.18 Two-Stream CNN and 3D-CNN

Perez et al. [140] proposed a pipeline, in which they evaluate the impact of different feature extractors, using two-stream CNN, 3D CNN and a local interest point descriptor, as well as different classifiers, such as end-to-end CNN, LSTM and SVM. Their results confirmed how challenging the problem is and highlighted the importance of explicit motion information to improve performance. They tackled this problem by first proposing the CCTV-Fights dataset [140], a challenging dataset containing 1,000 videos of real fights, with more than 8 hours of annotated CCTV footage that contain a diverse range of characteristics: different nature, duration of the videos, number of videos, purpose, recording source, staged or not, etc. The first step of the pipeline of their model, the feature extraction, consists of using the RGB information from the frames to extract meaningful features for the task at hand. These features are meaningful if they are discriminative enough for a decision-making method to correctly classify that feature as coming from a violent action or not. Depending

on how these features are generated, they can be used to describe a single frame or a small snippet of the video. The feature extraction for the two-stream based solution [141] is performed by using a 2D-CNN architecture to generate two different models, one for the spatial stream of the videos (RGB data of Frames) and another for the temporal stream (Stack of Optical Flows). The authors aggregated this information at the end by average pooling the scores or by concatenating the features from the last fully-connected layer before feeding it to a classifier.

The 3D-CNN solution [142] consists of a convolutional neural network architecture that enables convolutional on three dimensions. This approach can be explored not only for the spatial correlation in a single frame, but also for the temporal correlation between a short sequence of frames. Instead of using optical flow information, it is applied only over a stack of sequential frames. For the local interest-points, the method used the local features detector and descriptor named Temporal Robust Features (TRoF) based on the approached proposed by Moreira et al. [108], a work related to fight localization [140]. The next step of their approach, frame/snippet prediction, denotes the moment when the classifier will determine whether the feature comes from a positive case (fight) or a negative case, according to what it has learned before with the training and split of the data. The predictions, represented by a confidence score, are produced in this step at frame or snippet level [140]. In the last step, predictions from the previous step are aggregated to produce well-defined temporal segments for the predicted fight instances. Here, some higher-level intuition about the continuity of an event can be used to smooth the punctual predictions from the frames/snippets and achieve more realistic segments than directly using the scores independently. The authors addressed a straightforward smoothing and aggregating strategy. Smoothing is a traditional mean filter applied to reduce the impact of noisy prediction scores by using the score information from the neighboring snippets. After smoothing the scores, continuous predictions that satisfy a pre-determined score threshold are merged to create the final segments [140].

Ullah et al. [143] proposed a triple-staged end-to-end deep learning violence detection framework. First, people are detected in the surveillance video stream using a lightweight CNN model to reduce and overcome the massive processing of useless frames. Then, a sequence of 16 frames with detected people is passed to 3D-CNN to extract spatio-temporal features and feed the softmax classifier. Their method outperformed most state-of-the-art methods in different datasets.

Unlike most existing violent video detection models ignore the fact that the audio-visual data in the same violent video may correspond semantically, Gu et al. [144] proposed a novel violent video detection model based on the semantic correspondence between audio-visual data of the same video. Based on the fact that deep neural networks are used to extract features from three different modalities (appearance, motion and audio), their work fused these multimodal features through shared subspace learning. Semantic correspondence is used to guide this process via multitask learning and semantic embedding learning. To assess the effectiveness of their model, they conducted experiments on several public datasets and a self-built dataset, named Violence Correspondence Detection, and achieved competitive results [144].

2.7.19 Literature Results

Hockey Fights dataset [14] was created specifically from National Hockey League matches and contains different view points of person-to-person violence. The benchmark contains 1000 movies and its content is approximately two hours long. Table 2.1 presents some results for the Hockey Fights dataset using methods available in the literature.

Table 2.1: Literature Results for the Hockey Fights dataset [14].

Method	Year	Accuracy (%)
SRC [145]	2009	94.40±1.07
HoG+BoW [14]	2011	91.00
HOF+BoW [14]	2011	88.60
MoSIFT+BoW [14]	2011	90.90
MoWLD+BoW [107]	2015	91.90
MoWLD+SparseCoding [107]	2015	93.70±1.68
MoWLD+KDE+SparseCoding [107]	2015	94.20±1.91
MoIWLD+BoW [11]	2015	91.80±1.03
RVD [106]	2015	92.10±1.01
AMDN [146]	2015	89.70±1.13
moIWLD+ZhangSRC [11]	2016	96.80±1.04
STIFV [104]	2016	93.40
Three streams+LSTM [147]	2016	93.90
Mondal et al. [128]	2017	94.92±1.59
ConvLSTM [148]	2017	97.10±0.55
FightNet [149]	2017	97.01
LHoG+LHOF+Bow [110]	2018	95.10±1.15
BiConvLSTM SpatialTemporalEnconder [139]	2018	97.90±0.37
BiConvLSTM SpatialEnconder [139]	2018	**98.10±0.58**
CNN+LSTM (Darknet19) [136]	2019	98.00±0.55
Ullah et al. [143]	2019	96.00
LBP+GLCM [111]	2020	91.51±1.51

Violence in Crowds dataset [13] is a real-world, video footage of crowd violence, along with standard benchmark protocols designed to test both violent/non-violent classification and violence outbreak detections. The dataset contains 246 videos. All videos were downloaded from YouTube. The shortest clip duration is 1.04 seconds, the longest clip is 6.52 seconds, and the average length of a video clip is 3.60 seconds. Table 2.2 presents some results for the Violence in Crowds dataset using different methods covered in the literature.

Table 2.2: Literature results for the Violence in Crowds dataset [13].

Method	Year	Accuracy (%)
HoG [33]	2008	57.43±0.37
HOF [33]	2008	58.53±0.32
HNF [33]	2008	56.52±0.33
LTP [150]	2009	61.53±0.17
SRC [145]	2009	89.38±0.13
MoSIFT+BoW [14]	2011	57.09±0.37
ViF (SVM) [13]	2012	81.30±0.21
Fluid Mechanic [8]	2014	92.30
AMDN [146]	2015	84.72±0.17
SD [7]	2015	85.43±0.21
HOT [6]	2015	82.30
RVD [106]	2015	82.79±0.19
MoWLD+BoW [107]	2015	82.56±0.19
MoWLD+SparseCoding [107]	2015	86.39±0.15
MoWLD+KDE+SparseCoding [107]	2015	89.78±0.13
Holistic Features (SVM) [10]	2016	85.53±0.17
Holistic Features (GMM) [10]	2016	65.00±0.15
OViF (SVM) [29]	2016	76.80±3.90
OViF (AdaBoost) [29]	2016	74.00±4.90
ViF+OViF (SVM) [29]	2016	86.00±1.41
ViF+OViF (AdaBoost+SVM) [29]	2016	88.00±2.45
ViF+OViF (AdaBoost) [29]	2016	82.40±3.58
MoIWLD+BOW [11]	2016	88.78±0.19
moIWLD+ZhangSRC [11]	2016	93.19±0.12
STIFV [104]	2016	96.40
CENTRIST3D [101]	2017	78.00
Mondal et al. [128]	2017	94.92±1.59
ConvLSTM [148]	2017	94.57±2.34
LHOG+LHOF+BoW [110]	2018	94.31±1.65
BiConvLSTM [139]	2018	94.54±4.13
BiConvLSTM SpatialTemporalEnconder [139]	2018	96.32±1.52
BiConvLSTM SpatialEnconder [139]	2018	93.87±2.58
CNN+LSTM (Darknet19) [136]	2019	92.19±0.12
Ullah et al. [143]	2019	**98.00**
3D-ResNet [151]	2020	94.54±4.13
LBP+GLCM [111]	2020	89.06±3.32
Semantic Correspondence [144]	2020	97.69

Chapter 3

Proposed Method

In this chapter, we describe the proposed methodologies we approached to detect violent events. Initially, our work focused on evaluating holistic approaches to discriminating violence. To do so, we start by evaluating CENTRIST and the combination between this descriptor and Histogram of Gradients (HoG) [27] through two well-known datasets used to detect crowd violence: Violence in Crowds and Hockey Fights.

First, we assess how the capture frame rate influences performance. Then, we evaluate which preprocessing methods can make CENTRIST more accurate. Following the influence of using three different scales (80%, 100%, 120%) to extract features, they were also compared. In addition, we also experiment deep features generated through pre-trained deep learning models for comparison purposes. Then, we assess approaches based on code generation to convert handcrafted descriptors into a more representative feature descriptor. For this perspective, we address feature vectors obtained using sparse coding and evaluate distances to centroids obtained from K-means clusters or GMM Gaussians. Finally, we explore the generation of descriptors using dense optical flow and combine them with CENTRIST.

After achieving the CENTRIST limit using holistic approaches, we evaluated other methods. In order to assess smaller parts of the scene, the sliding window was used with different block sizes to evaluate the detection performance using the same types of descriptors. Then, we refine our method using optical flow to discriminate less relevant blocks, comparing their local optical flow level with the global optical flow.

The main stages of our method are illustrated in Figure 3.1, which are detailed in the following sections. This pipeline was used to evaluate the proposed CENTRIST-based descriptors, as well as other handcrafted descriptors.

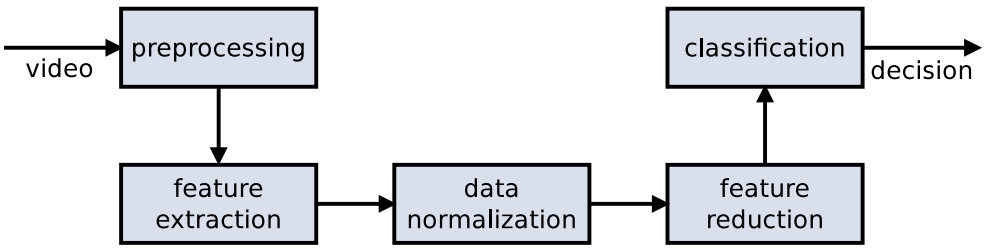


Figure 3.1: Main stages of the proposed methodology.

3.1 Preprocessing

In the first step of our pipeline, we read the videos and apply the preprocessing filters. For the video reading, we initially identify the frame rate at which each video footage was recorded. Then, the reading is performed through two approaches: reading all frames or using a fixed frame rate, which depends on the approaches adopted in the feature extraction and training steps.

Depending on the type of video descriptor, the image is converted to a specific image color format. For CENTRIST and other handcrafted descriptors used in our work, we convert the image into gray-scale color space. Each individual video frame is then preprocessed to make it more suitable for further processing. For methods based on deep features, presented in Section 3.3.2, we pass RGB channels as input to the deep models used to extract features.

Different preprocessing techniques are applied to evaluate their influence in the classification process, including the following: (i) histogram equalization to distribute pixel intensities to enlarge the contrast range in order to reduce illumination interference, (ii) Gaussian blur filter or Gabor filter are applied to reduce noise effect, and (iii) a background subtraction using Mixture of Gaussians (MoG) [73] in larger order to avoid objects not related to the people in the scene. Additionally, video frames are also evaluated at multiple scales, in which their dimensions are reduced or increased by a constant factor.

3.2 Feature Extraction

To evaluate the video recording, we consider not using all frames due to the relative similarity between consecutive frames. Thus, in our research, we compare the processing of all frames and the skipping of some intermediate frames, described as follows: (i) to skipping a fixed interval between the frames to be captured (10 or 20 frames), and (ii) specifying a frame processing rate (for instance, 1.0Hz or 2.0Hz), considering that the video recording frame rate is relevant.

In our work, we employ the descriptors to extract feature vectors from the video frames in two different strategies: (i) the descriptor is used to evaluate the entire content of each frame at a specific frame capture rate, as shown in Figure 3.2, and (ii) a grid with specific dimension is used to split each frame into blocks. A sliding window of specific size (64×64, 72×72, 96×96 and 128×128 pixels) is used to individually evaluate frames in stages at half the specified block size, as shown in Figure 3.3, in order to evaluate overlapping regions and avoid processing cut regions from actors that are set at the borders of the block.

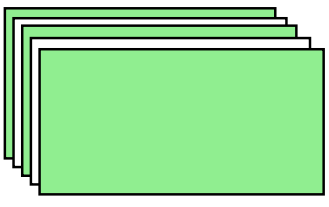


Figure 3.2: The figure depicts how the entire frame content is processed at specific frame rates.

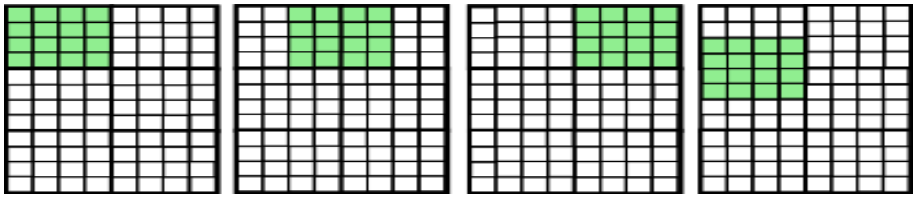


Figure 3.3: First four iterations for the sliding window using step equal to half of the block size (N). The first three steps move to right to the end of the row, then the sliding window is moved by $N/2$ pixels below the beginning of the next row.

In addition to the methods mentioned previously, we also explored a preprocessing approach using multi-resolution (Figure 3.4 by extracting features from the current frame with different resolutions (80%, 100% and 120%) [152].

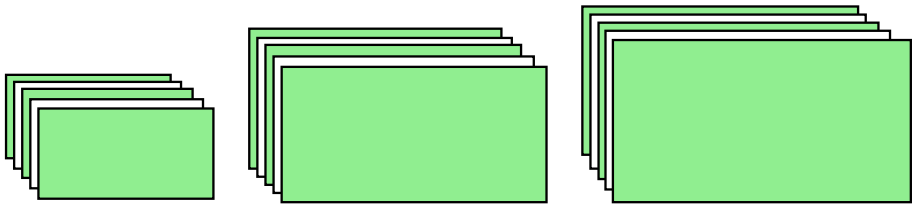


Figure 3.4: Strategy for constructing the feature descriptor using frames at multiple resolutions.

Additionally, two different approaches are assessed to refine the prediction step in order to consider a video as violent or not: (i) the classification is performed frame-by-frame and the prediction uses 50% as a decision threshold and (ii) looking for a threshold that discriminates more efficiently violence we calculate its value according to the value that maximizes accuracy in the training dataset.

3.3 Block Selection Criteria Based on Optical Flow Evaluation

Regarding the experiments that will shown in Section 4.7, we identify that the camera motion can be considered the major challenge for finding the regions of interest. After observing several pictures, it is possible to determine that most of the scene movement corresponds to camera motion. To deal with this problem, methods based on optical flow have been employed as a likely tool for selecting regions of interest in which the actors are performing the action in the scene.

To evaluate each region, we consider using two approaches based on the optical flow: (i) to check if the block presents some optical flow value greater than the global threshold value and (ii) to calculate a local measure value and compare it to a threshold obtained from the global measure value.

Three measures are used to analyze the optical flow of the entire frame and at patches of this frame in order to identify in which regions the motion is more relevant. The notation (`Avg`/`Median`/`Otsu`) will be used here and in Section 4.10.1 to identify which filtering approach was used in each experiment.

1. `Avg`: Average of optical flow.
2. `Median`: Median of optical flow.
3. `Otsu`: Otsu's method.

In our work we assess three approaches to evaluate the local optical flow in order to determine if the blocks can be considered relevant at the scene:

1. select blocks whose optical flow mean value (`Avg`) is greater than the threshold obtained through one of statistic measures (`Avg`/`Median`/`Otsu`).
2. select blocks whose optical flow median value is greater (`Median`) than the threshold obtained through one of the statistic measures (`Avg`/`Median`/`Otsu`).
3. selected blocks that contain at least one pixel (`Any`) whose optical flow value is greater than the threshold obtained through one of the statistic measures (`Avg`/`Median`/`Otsu`).

For instance, Figure 3.5 illustrates the dense optical flow for consecutive frames using 2Hz frame rate. Initially, an image from dense optical flow is obtained using OpenCV library to combine the orientation and magnitude of the optical flow.

To do so, the Farneback's method is used to obtain the orientation and and magnitude and the combination is done by a cartographic to polar operation followed by a conversion from HSV to BGR [153]. Then, the global threshold is calculated through Otsu's method and points whose value is below the threshold are removed. Finally, when moving the sliding window, we compare if there is any point in the block whose value is greater than the threshold. Thus, only blocks whose relative motion is greater than the global scene motion are evaluated.

3.3.1 Census Transform Histogram

As described in Section 2.3.1, Census Transform (CT) is a non-parametric local transform originally designed to match local patches [26, 54]. Similar to other non-parametric local transforms, which are based on intensity comparisons (for instance, ordinal measures [26, 154]), Census Transform is robust to illumination changes and gamma variation.

As mentioned by Wu and Rehg et al. [26], histogram of CT values for an image or an image region can be easily computed. We used the CENTRIST (Census Transform Histogram) as our visual descriptor with length 256. CENTRIST can be computed very efficiently. It involves only 16 operations to compute the CT value for a center pixel (8 comparisons and 8 additional operations to set bits to 0 or 1). The cost of computing CENTRIST is linear in the number of pixels of the region under consideration. Thus, the histogram is generated through the values computed for each pixel of the frame. Figure 3.6 illustrates a sample of the CENTRIST histogram processed over a violence scene region.

To evaluate images of different sizes or resolutions, a normalization is applied to make the descriptor independent of the image size. After generating the vector histogram, each term is divided by the number of elements presented in the histogram. Therefore, the resulting histogram will contain elements with values in the range between 0 and 100%.

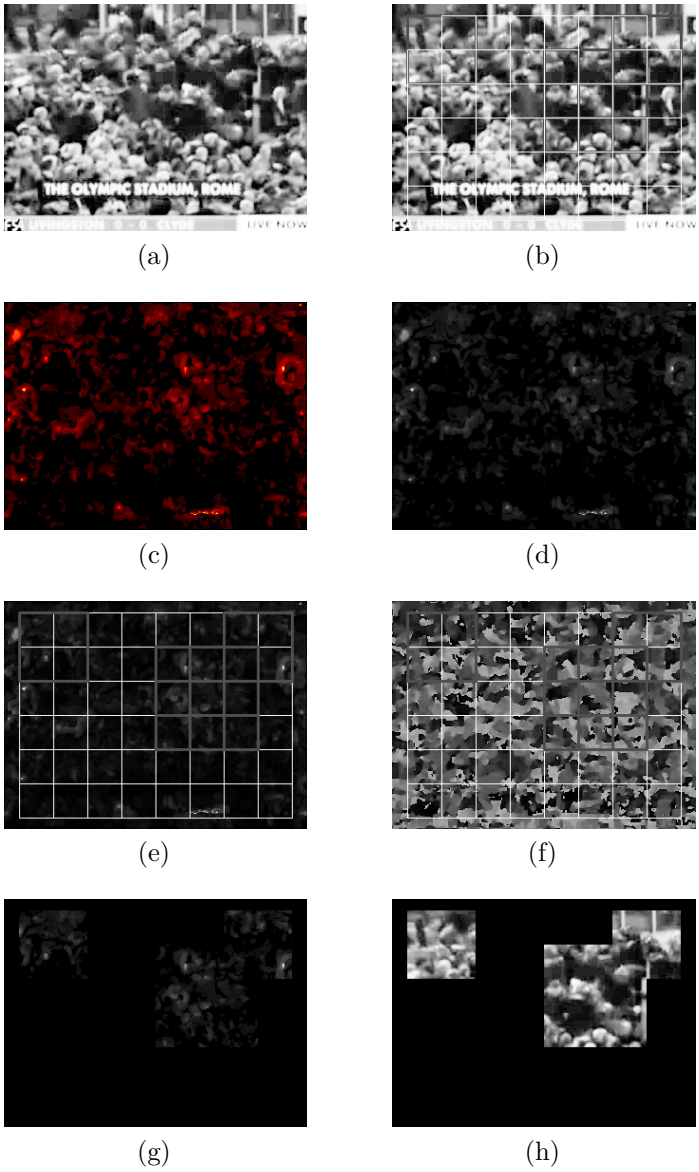


Figure 3.5: Images that illustrate each step of the process to filter the blocks using optical flow. In this example, images are extracted for a specific 2-frame sequence using frame rate of 2Hz to evaluate the dense optical flow. (a) image under evaluation; (b) grid with the blocks to be evaluated by the sliding window; (c) heat map for the dense optical flow obtained by combining the optical flow's magnitude and orientation; (d) pixels whose intensity is superior than the threshold obtained through the Otsu's method; (e) image obtained from the values of the optical flow orientation; (f) image obtained from the values of the optical flow magnitude; (g) blocks selected by evaluating that the average of the magnitude of the block is greater than the value obtained by the Otsu's method; (h) regions selected after block filtering.

Figure 3.7 shows how any variation in orientation or reflection impacts the distribution in the histogram over the CENTRIST descriptor. Depending on the type of image structure, the histogram will concentrate most of the values in a specific region. Thus, it is possible to observe that, although CENTRIST can be considered invariant to illumination, it is important to guarantee the dataset is diverse enough to turn the model robust to

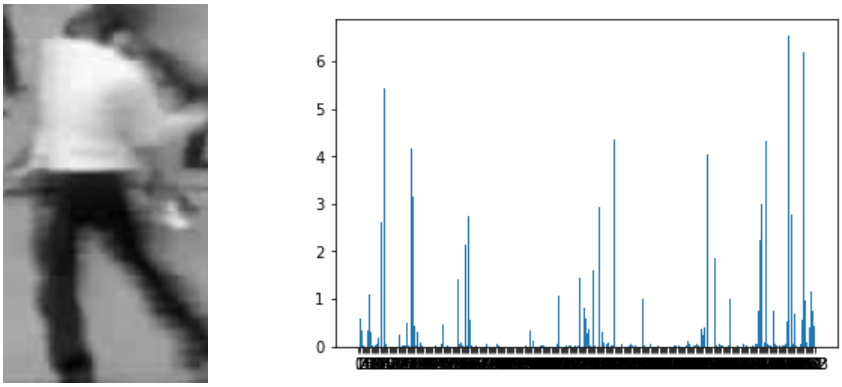


Figure 3.6: CENTRIST extracted over a region that contains a person who participates in a violent scene.

orientation and reflection variations.

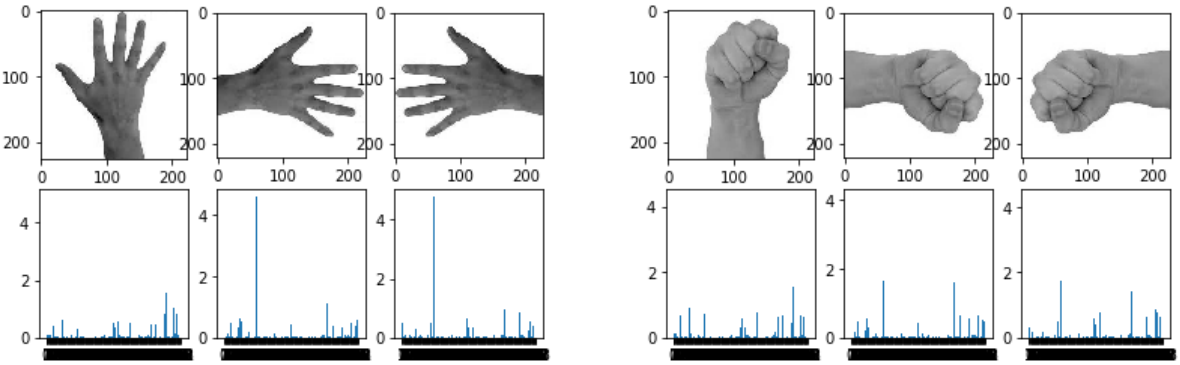


Figure 3.7: CENTRIST extracted from hand gestures A (left) and B (right) in different positions.

3.3.2 Feature Extraction using Deep Features

In addition to handcrafted features, deep learning models based on pre-trained networks, such as Inception-v3 and Xception, were also used to extract features from the video frames. In Figure 3.8, it is illustrated how the pipeline was modified to be executed using deep features. Unlike the process used for handcrafted descriptors (shown in Figure 3.1), after applying preprocessing filters on the frames, a deep learning model is used directly as descriptor to extract features that are passed to traditional machine learning models, for instance SVM, Adaboost, among others.

Figure 3.8: Main stages of the pipeline using deep features.

Due to reduced amount of data to train a deep learning model, we opted for using a pre-trained model to extract features and use traditional machine learning techniques to

detect violence. Therefore, as described in Section 2.7.14, we applied models previously trained on the ImageNet competition as basis for extracting features from each frame individually. To obtain the features, the values that are passed in general to the latest classification layer are used.

To work with pre-trained models, we use models compatible with the TensorFlow [155] and Keras [156] frameworks. Using Keras API, it is possible to directly load the network structure of Inception-v3 [125] and Xception [12] models and the respective weights obtained from the dataset used for the ImageNet competition.

3.3.3 Encoding Techniques

In addition to directly evaluating the features, Figure 3.9 illustrates the stages of our method when the encoding technique is applied to obtain a more representative code. The training and classification steps are applied after the feature vector is converted to the new encoding.

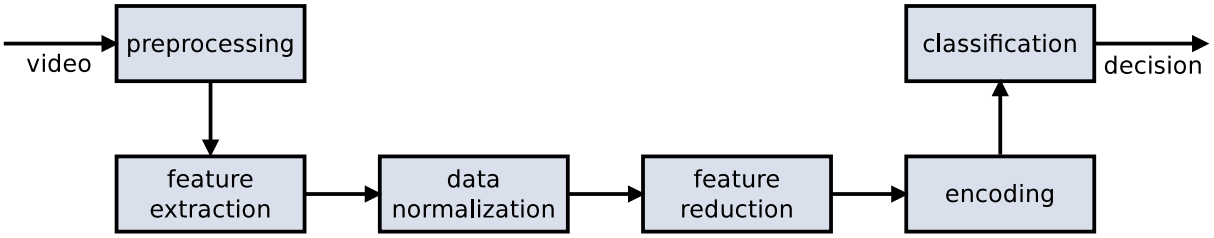


Figure 3.9: Main stages of the proposed methodology using the encoding technique.

For this method, we proposed three different techniques to encode the feature vectors from CENTRIST-based descriptors:

1. Euclidean-distance [157] from K-means clusters [91].
2. Mahalanobis-distance [158] from GMM [93].
3. sparse coding [96] (described in Section 2.6.3).

In our work, two methodologies based on K-means are adopted:

1. We will apply K-means to split our data into two specific clusters, where each one will represent one of the our two classes (violence and non-violence). Then, the prediction for an unknown sample will be returned, since the probability of the feature vector belongs to the cluster that contains samples of violence.
2. K-means will be used to group N clusters. Then, the distance to each cluster will be used as dictionary to train the model using a secondary classifier (for instance, an SVM). In this approach Euclidean distance [157] is used to measure the distance from the feature vector to each cluster.

Similarly to K-means, using GMM two methodologies are also adopted:

1. the training will be carried out using two Gaussian distributions, where each represents a specific class (violence and non-violence). Then, the distance to each Gaussian will be related to the class.
2. similarly to K-means, GMM will be used to generate N Gaussians. Then, the Mahalanobis distance [158] to each Gaussian is used as dictionary [159]. The resulting feature vector will be employed for training using a secondary classifier (for instance, an SVM).

To apply these techniques, we use the tools provided by the Scikit-Learn library [83].

3.4 Data Transformation

Once the descriptor was extracted, data scaling normalization was optionally applied in some experiments to transform the data with zero mean and unit variance. Principal Component Analysis (PCA) [160] technique can also be applied for dimensionality reduction. We evaluate the effectiveness of the normalization and dimensionality reduction of the features in our experiments.

Initially, PCA is applied using a percentage number of components of the used descriptor. After empirically testing different percentage values (between 25 and 50%) to process PCA, it was identified that 35% would obtain satisfactory results without demanding high computational time. Thus, most experiments were performed using 35% for the number of components. After identifying the best techniques, we approached the 99% of cumulative variance as a measure to define the number of components to be used in PCA.

3.5 Classification

In the training step, we evaluate the use of the Support Vector Machine (SVM) classifier [161] with Radial Basis Function (RBF) and linear kernels, as well as Random Forests (RF) [85] and Stochastic Gradient Descent (SGD) [162], and techniques based on boosting, such as Adaboost [88] and eXtreme Gradient Boosting (XGB) [90] classifiers.

After the training step, each video sequence from the test dataset is individually evaluated. In order to make the final decision, we use a simple vote strategy to evaluate the prediction for each video frame based on the proportion of violent frames. For our sliding window approach, considering in short video (3 to 5 seconds duration) that only a few frames are processed, we keep using this generic voting approach based on the proportion of violence results obtained from all frames. For the direct extraction approach, since the result for each frame corresponds exactly to the result of a feature, we can generalize our formulation as

$$\text{output} = \begin{cases} \text{violence}, & \text{if } VF \;/\; (VF + NVF) > T \\ \text{non-violence}, & \text{otherwise} \end{cases} \tag{3.1}$$

where VF represents the number of features considered violent, NVF represents the number of features considered non-violent, and T is a threshold.

3.6 Benchmark Protocol

In this section, we present the protocol for evaluating the Violence in Crowds and Hockey Fights datasets used in our experiments.

To assess the accuracy of the classification process, we employ the same protocol specified by Hassner et al. [13], that is, a k-fold cross-validation protocol. We split the video sequences into k sets, where k is equal to 5 sets for both datasets evaluated during our research. In each set, half of the videos depict the violent crowd behavior and half the non-violent behavior. In some cases, different videos originate from the same YouTube clip or the same scene. In such cases, these videos are all included in the same set (the sets are mutually scene exclusive).

For each test, four out five sets are used for training and the performance for the violence labeling is then assessed on the remaining set. After obtaining the results through the five different combinations of datasets, the final result is calculated by the prediction accuracy rate and its respective standard deviation.

Chapter 4

Experimental Results

In this chapter, we describe in chronological order the experiments carried out during this research work, supporting the decision we made according to the established premises.

The results for the Violence in Crowds and HockeyFights datasets were evaluated according to the 5-fold cross-validation protocol specified by Nievas et al. [14]. In this protocol, the dataset is split into 5 smaller sets, where a model is trained using 4 these folds and evaluated in the remaining fold. Finally, the mean and standard deviation are calculated from the five outcomes for possible variations and, therefore, these values are used as final measure to assess the method efficiency.

The first sections are divided as follows: specification of hardware used during our research, individual results for the Violence in Crowds dataset using CENTRIST, results from the combination between HoG+CENTRIST. Next, we experiment Deep Features to evaluate the violence detection and present our preliminary conclusions. After identifying the most promising approaches using the entire frame, we applied them to the Hockey Fights dataset to evaluate their efficiency in a different scenario.

We also examine alternative strategies for assessing the scene in order to improve the method efficiency, such as Gabor [61] as preprocessing filter and different coding approaches as training methods. Then, we apply the Sliding Window approach to evaluate blocks from the video scene instead of extracting descriptors from the entire scene at once. In addition, we attach information related to motion from descriptors obtained through the Optical Flow attributes. Next, we use the Optical Flow magnitude to filter out the blocks that may be most relevant across the scene. Finally, we analyze the most relevant results in each approach and determine which would be the most competitive. Finally, we conduct cross-dataset experiments to evaluate generalization aspects in Section 4.11.

4.1 Hardware and Software Specifications

During our research, we carried out a large number of experiments that demanded different types of resources as they progressed. Thus, we used different hardware specifications according to the availability and needs of the experiments. At beginning of the research, we started very simple configurations, since we were using relatively simple approaches. To execute the techniques that extract entire frames from the Violent in Crowd and Hockey

Fights datasets, only 8 GB of RAM memory was sufficient. Since we explored techniques that demanded more computational resources, we moved to a dedicated server to deal with CPU and memory requirements. Furthermore, when we explored deep learning models to generate deep features, as shown in Section 3.3.2, the use of a GPU (Graphic Processing Unit) was crucial to train the models with TensorFlow in a reasonable time. Although the specification gains have provided substantial improvements in execution speed, this change has not made it possible to compare training and prediction times.

The hardware configurations used in our experiments were:

1. Laptop Dell Inspiron 4200 with an Intel i5-4200 CPU 1.8 GHz, 12 GB RAM, NVidia GeForce 740M 2GB, Ubuntu 14.04 64-bit operating system.
2. Desktop computer with an Intel i7-2600 CPU 2.4GHz, 8 GB RAM, NVidia GeForce 1050Ti 4GB, Ubuntu 14.04 64-bit operating system.
3. Server with an Intel Xeon E3-1200 CPU 3.3 GHz, 32 GB RAM, Ubuntu 14.04 64-bit operating system.
4. Laptop Dell G3 with an Intel i7-9750H CPU 2.60 GHz, 12 GB RAM, NVidia GeForce 1660Ti with Max-Q 6GB, Ubuntu 14.04 64-bit operating system.

The software versions required by the platform to run our experiments were:

1. CUDA: 7.5
2. OpenCV: 2.4.13.7
3. Python 2.7
4. wxPython: 2.8.10.1
5. ffmpeg
6. OpenJDK-7

4.2 Results Using CENTRIST

At the beginning of the research, we focused our efforts on identifying the potential of CENTRIST as a descriptor for discriminating violence scenes. Then, using the Violence in Crowds dataset, we performed experiments using the entire frame and different input parameters.

Initially the video capture frequency was analyzed to determine how many frames could be ignored due to the great similarity between consecutive frames. Thus, our first approach was to define the number of frames that could be discarded without reducing effectiveness. However, since each video was recorded using a specific frame rate, we also evaluate frame selection based on the frame rate for each video, rather than skipping a fixed number of consecutive frames for all videos.

In Table 4.1, we report our results using the CENTRIST descriptor, skipping either 10 or 20 frames and using different frame rates (1 Hz, 2 Hz). These tests were performed using three different classifiers: SVM (Support Vector Machine) with both linear and RBF kernels, AdaBoost, and Random Forest. The features were extracted by applying only Histogram Equalization [163] as preprocessing filter.

Table 4.1: Accuracy for Violence in Crowds dataset [13] using different frame rates.

Method	10	20	2.0 Hz	1.0 Hz
CENTRIST (AdaBoost)	83.71±4	81.68±5	84.55±4	84.48±4
HoG+CENTRIST (AdaBoost)	84.93±4	85.38±4	87.85±2	88.61±1
CENTRIST (SVM-RBF)	82.90±3	82.06±2	85.71±3	84.48±4
HoG+CENTRIST (SVM-RBF)	87.78±2	86.18±1	87.81±2	88.61±1
CENTRIST (SVM-Linear)	81.72±3	84.60±3	86.93±3	82.93±1
HoG+CENTRIST (SVM-Linear)	84.58±3	82.96±4	**88.61±2**	85.75±3
CENTRIST (RandomForest)	86.16±4	82.50±4	82.91±5	82.93±2
HoG+CENTRIST (RandomForest)	85.00±4	80.11±6	85.40±4	83.33±2

From Table 4.1, we can observe that the use of fixed values to skip frames generates less effective results than skipping frames in the temporal interval.

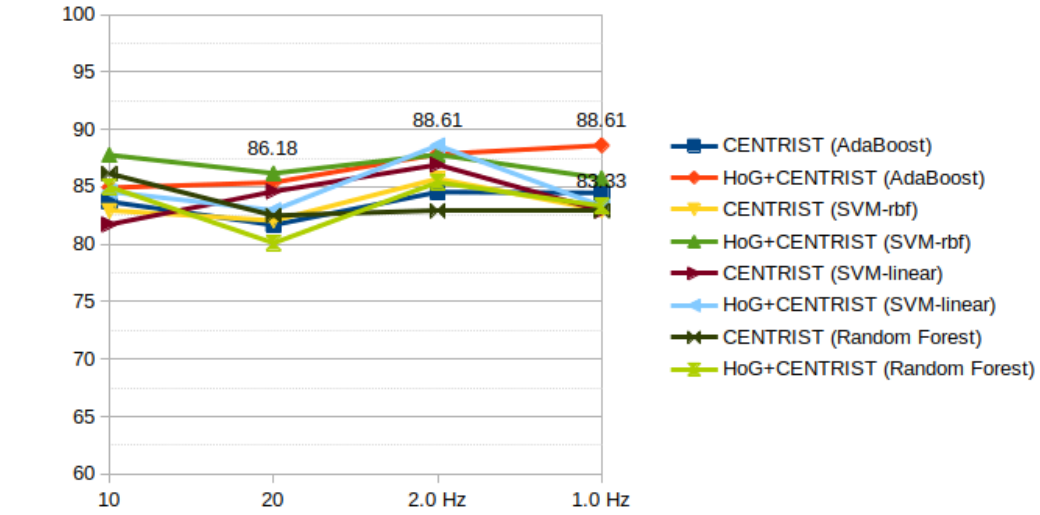
To assess the influence of contrast in addressing the problems related to low scene illumination, we apply histogram equalization to the video frames and verify how it affects the descriptor performance. Although the Census Transform is described as robust to illumination and gamma variations, by increasing the illumination contrast, we aim to facilitate visual inspection and enable the comparison of results between descriptors that are not considered robust against illumination issues. In Table 4.2, results of the experiments are presented using histogram equalization before extracting the descriptors.

Table 4.2: Accuracy for Violence in Crowds dataset [13] using different frame rates after applying histogram equalization.

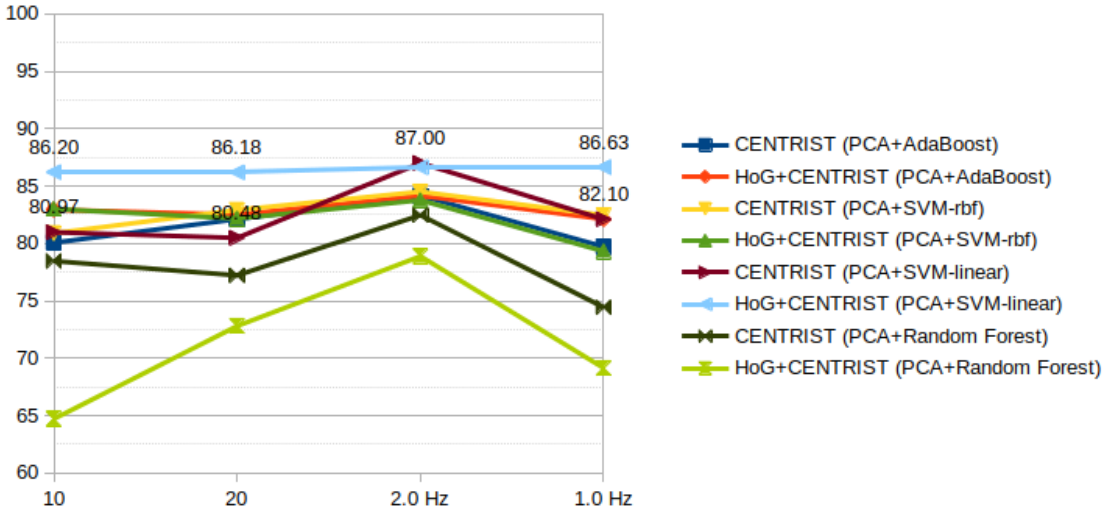
Method	10	20	2.0 Hz	1.0 Hz
CENTRIST (AdaBoost)	81.25±7	82.53±8	84.51±3	82.10±2
HoG+CENTRIST (AdaBoost)	81.28±7	86.60±2	84.18±3	**88.21±3**
CENTRIST (SVM-RBF)	82.88±2	82.88±4	85.75±2	85.31±2
HoG+CENTRIST (SVM-RBF)	83.00±5	82.16±5	83.76±2	79.31±6
CENTRIST (SVM-Linear)	81.71±2	80.51±3	86.60±4	84.60±3
HoG+CENTRIST (SVM-Linear)	86.20±3	85.78±4	87.03±4	86.63±3
CENTRIST (Random-Forest)	83.75±3	85.75±5	84.51±3	84.60±2
HoG+CENTRIST (Random-Forest)	86.60±2	81.28±7	85.83±4	82.15±3
CENTRIST (PCA+AdaBoost)	80.06±4	82.11±2	84.15±7	79.71±3
HoG+CENTRIST (PCA+AdaBoost)	82.93±5	82.51±3	84.15±2	82.11±6
CENTRIST (PCA+SVM-RBF)	80.88±5	82.91±5	84.50±3	82.46±3
HoG+CENTRIST (PCA+SVM-RBF)	83.00±5	82.16±5	83.76±2	79.31±6
CENTRIST (PCA+SVM-Linear)	80.97±4	80.48±1	**87.00±2**	82.10±4
HoG+CENTRIST (PCA+SVM-Linear)	86.20±3	86.18±2	86.63±5	86.63±3
CENTRIST (PCA+Random-Forest)	78.46±3	77.21±1	82.46±3	74.43±3
HoG+CENTRIST (PCA+Random-Forest)	64.70±6	72.80±3	78.88±5	69.13±2

Summarizing the information shown in Tables 4.1 and 4.2 in a line graphic, we can identify the influence of the capture frame rate for the results. According to the results illustrated in Figure 4.1, it is possible to observe that the use of adaptive capture frame

rate based on the video recording frame rate is shown to be more effective than using a fixed rate to skip consecutive frames. For instance, we obtained higher results for SVM as classifier using 2 Hz for frame rate and using 1 Hz for AdaBoost. Furthermore, we observe that, using 1 or 2 Hz, the standard deviation is considerably lower for most classifiers. Thus, when comparing the results mentioned previously, in which the histogram equalization (HEq) was applied, it is possible to notice that, when contrast is normalized, we obtain statistically similar results, but approximately inferior to the results evaluated without this preprocessing step.



(a)



(b)

Figure 4.1: (a) Accuracy for Violence in Crowds dataset [13] at different frame rates; (b) Accuracy for Violence Crowds dataset using PCA [13] at different frame rates.

From Table 4.3, it is possible to notice that, for some combinations, we achieved results that overcome the baseline [13] and other methods available in the literature (shown in Table 2.2). For instance, the CENTRIST descriptor combined with PCA and SVM techniques was able to obtain an accuracy of 86.16±2.8%. Even though the concepts involved in the process are relatively simple, the result was superior to other works. On

Table 4.3: Results for Violence in Crowds dataset [13] using Multiscale and Background Subtraction (MoG).

Method	Accuracy (%)
CENTRIST (SVM)	85.75±5.57
CENTRIST Multiscale (SVM)	84.90±5.40
CENTRIST (PCA+SVM)	86.16±2.80
CENTRIST Multiscale (PCA+SVM)	85.81±2.64
CENTRIST (MoG, SVM)	82.51±2.72
CENTRIST Multiscale (HEq+MoG, SVM)	81.70±2.66
CENTRIST (MoG, PCA+SVM)	83.73±2.21
CENTRIST (HEq+MoG, PCA+SVM)	82.88±2.98
CENTRIST (SGD)	**87.80±1.84**
CENTRIST (MoG, SGD)	83.35±1.30
CENTRIST Multiscale (HEq+MoG, SGD)	82.96±2.50
CENTRIST (MoG, PCA+SGD)	83.30±2.55

the other hand, we can also identify that the multi-scale approach was slightly inferior to its single-scaled version, achieving 85.81%±2.6% of accuracy at maximum, with scales 0.8, 1.0, and 1.2.

Then, we assess the influence of applying the Gaussian filter [164] to each frame before extracting features to reduce the effect of noise and illumination. Multi-scaling was also explored, as mentioned in Section 3.2. Two extra scales (80% and 120%) associated with the standard resolution (100%) in the training step were evaluated.

In Figure 4.2, the results using different preprocessing approaches are compared through CENTRIST and HoG+CENTRIST using SVM. We can observe that, for most preprocessing approaches used, performance using raw image is still among the best options. For CENTRIST, although this descriptor is considered robust for illumination issues, accuracy using histogram equalization was the only result that overcomes the accuracy using raw input. For HoG+CENTRIST, the result using histogram equalization shows to be slightly worse than using the raw image. In addition, trying to reduce noise was not effective using Gaussian blur filter or Gabor filter for both descriptors. When looking at the multiscale approaches, despite expectations when augmenting our dataset, the results did not seem to be promising.

4.2.1 Results using HoG+CENTRIST

As mentioned in Section 2.3.2, HoG [2] was used in combination with CENTRIST to add more descriptive information to our descriptor. As shown in Table 4.4, when applying the background subtraction, the combination of CENTRIST and HoG descriptors reached the best accuracy rate (87.45±2.7%) when using only dimensionality reduction with PCA, and SVM with RBF kernel as classifier for the Violence in Crowds dataset.

However, in experiments without using background subtraction, the results are still representative (86.96±3.8%), as the use of background subtraction was not sufficiently

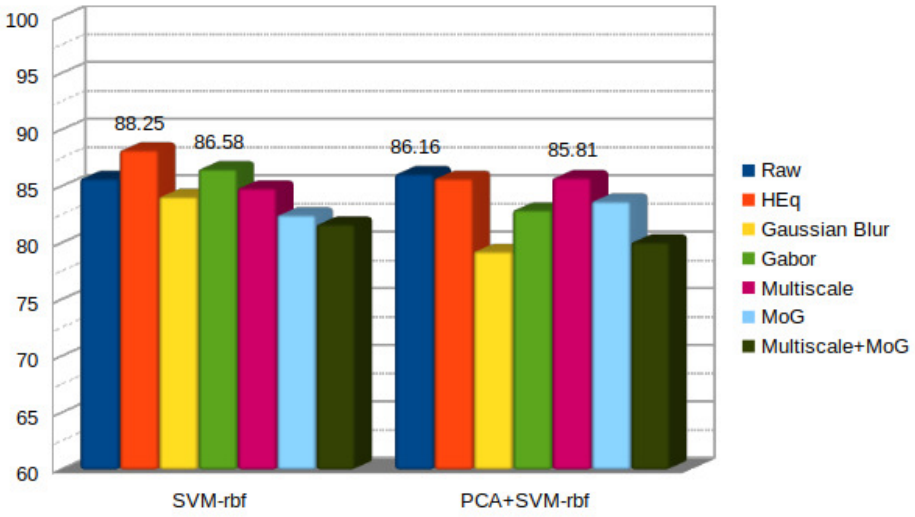


Figure 4.2: Accuracy for Violence in Crowds dataset [13] at different 2 Hz using different preprocessing approaches.

meaningful. After visually analyzing the dense optical flow from several video scenes, it is possible to state that regions of interest can be highly affected by camera motion. However, the use of background subtraction using MoG is not as relevant to the results in terms of accuracy.

In addition, we also conducted experiments with different classifiers. The combination of HoG and CENTRIST descriptors was evaluated using SGD (Stochastic Gradient Descent), AdaBoost and Random Forests that produced, respectively, the following accuracy rates: 85.81%, 86.61% and 86.20%, which are results similar to the formers obtained using SVM classifier.

Table 4.4: Results for Violence in Crowds dataset [13] with HoG+CENTRIST using different classifiers.

Method	Accuracy (%)
HoG+CENTRIST (SVM-RBF)	86.96±3.1
HoG+CENTRIST (PCA+SVM-RBF)	86.61±3.6
HoG+CENTRIST (AdaBoost)	86.61±4.1
HoG+CENTRIST (SGD)	85.81±5.8
HoG+CENTRIST (Random Forest)	86.20±6.0
HoG+CENTRIST (HEq, SVM-RBF)	86.96±3.8
HoG+CENTRIST (MoG, SVM-RBF)	87.40±1.4
HoG+CENTRIST (HEq+MoG, SVM-RBF)	**87.45±2.7**
HoG+CENTRIST (HEq+MoG, SGD)	85.00±4.3

In Figure 4.3, we have a comparison using different classifiers with CENTRIST and HoG+CENTRIST. When analyzing such information, we can identify that the SVM obtained satisfactory results for both descriptors. Although SGD achieved the best result using CENTRIST, the same was not true using HoG+CENTRIST. For the combination, the results considerably improved for all descriptors. Furthermore, the accuracy is similar

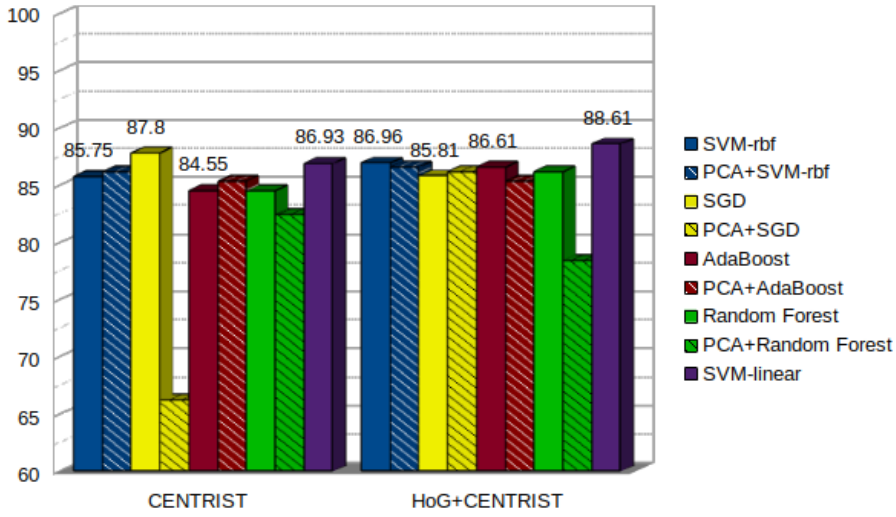


Figure 4.3: Accuracy for Violence in Crowds dataset [13] at different frame rates using different classifiers.

using all different classifiers. This fact shows us that the combination of these descriptors makes the performance more robust.

Another fact that we identified is related to the use of PCA. In general, the difference between the results obtained through PCA and SVM is practically insignificant to the results without PCA. While we cannot conclude whether or not using the PCA is better, we can point out that the feature length can be reduced using PCA without a significant lost of effectiveness.

Moreover, to ensure that the HoG+CENTRIST combination is not due to HoG individually, we also performed some experiments using HoG to identify its relevance compared to CENTRIST. As shown in Table 4.5, we can state that HoG individually is not responsible for obtaining sufficient information to discriminate violence. Thus, we can conclude that the HoG and CENTRIST feature vectors, when combined, are actually more effective than when evaluated individually. In addition, we can observe that, although both descriptors evaluate the histogram of characteristics of the image structure, CENTRIST seems to be considerably more representative than HoG for violent detection.

Table 4.5: Accuracy for Violence in Crowds dataset [13] using HoG.

Method	Accuracy (%)	Accuracy (%) using PCA
HoG (SVM-RBF)	**80.93±6.8**	77.71±8.3
HoG (RandomForest)	74.95±8.3	66.70±7.1
HoG (XGB)	76.46±2.6	78.55±9.5

4.2.2 Best Results for the Approach using Direct Frame Extraction

After evaluating the results using different classifiers, we concluded that SVM (using both RBF and linear kernels), SGD and AdaBoost could obtain promising results. In addition, considering that HoG+CENTRIST could be used effectively to find relevant features to discriminate the scene, we explore the descriptor in more details. In Section 3, we described more deeply which techniques were addressed.

In Table 4.6, we report our best results using the CENTRIST and HoG+CENTRIST descriptors for the Violence in Crowds dataset, after applying histogram equalization (HEq). From these results, it is possible to observe that, using 2 Hz, we reached the best results for HoG+CENTRIST compared to Tables 4.1 and 4.2.

Table 4.6: Accuracy for the dataset Violence in Crowds dataset [13] using different frame rates after applying histogram equalization.

Method	10	20	**2 Hz**	1 Hz
CENTRIST (HEq, AdaBoost)	81.25±7	82.53±8	84.51±3	82.10±2
CENTRIST (HEq, SVM-RBF)	82.88±2	82.88±4	85.75±2	85.31±2
CENTRIST (HEq, SVM-Linear)	81.71±2	80.51±3	86.60±4	84.60±3
HoG+CENTRIST (HEq, AdaBoost)	86.60±2	81.28±7	88.21±3	84.18±3
HoG+CENTRIST (HEq, SVM-RBF)	88.21±2	87.45±3	**88.25±3**	87.40±1
HoG+CENTRIST (HEq, SVM-Linear)	86.20±3	85.78±4	87.03±4	86.63±3

When comparing the results obtained with the techniques described previously, such as Gaussian blur or background subtraction, no representative result can be highlighted. Although these techniques have been shown to be effective in the literature, we noticed that, for Violence in Crowds dataset, this is not true. By analyzing this dataset, we found that the low quality of the images presents high influence when the smoothing filter is used. Moreover, when background subtraction using MoG is applied, we notice that, due to camera motion, the quality of the focused regions is not appropriate for the identification of violence. In general, instead of identifying the movement of the actors, we can see the camera motion is considerably present.

4.2.3 Results using Deep Features

In this section, we present results using approaches based on deep features mentioned in Section 3.3.2. To compare the handcrafted approaches, we evaluate techniques based on pre-trained deep learning models to extract features from the scene. At the beginning of the work, we used the SqueezeNet model [165] to evaluate the Violent in Crowds dataset and, after training approximately 100 epochs, we obtained very low results compared to the results obtained using CENTRIST (shown in Table 4.1). Thus, using only the Violence in Crowds dataset, we assume that the amount of data was not enough to train a model for deep learning and choose to employ a pre-trained model using a larger dataset to extract features that would be passed to a traditional classification model.

In our work, we explored SVM using only two kernel types (linear and RBF) and AdaBoost, because here we intend to determine whether this approach would be more promising than the use of handcrafted descriptors. In Table 4.7, we have few results from experiments using the two different deep learning models: Inception-v3 [125] and Xception [12], pre-trained using the ImageNet dataset, to extract features that have been eventually used as input to traditional machine learning models.

Table 4.7: Accuracy for dataset [13] using different frame rates.

Method	Accuracy (%) (1.0 Hz)	Accuracy (%) (2.0 Hz)
Inception-v3 (SVM-RBF)	**78.88±4.0**	76.43±3.6
Inception-v3 (SVM-linear)	65.11±5.4	63.48±5
Xception (SVM-RBF)	68.33±2.8	66.25±3.3
Xception (AdaBoost)	68.35±4.4	70.75±2.5

When evaluating different frame rates using Inception-v3, we found that applying all frames to extract features could not make the model more robust. Although using all frames for training with Inception-v3 allowed 76.83±6.1% for lower frame rates, it was possible to obtain better results. For instance, when using 1.0 and 2.0Hz for video capture, we obtained similar or superior results, as shown in Table 4.7. After identifying this fact, to save time, we chose not to evaluate all the frames in the following experiments, using deep learning models. Therefore, when comparing the results of Inception-v3 and Xception, although Xception was slightly superior to Inception-v3 on the ImageNet dataset (shown in Section 2.7.14), the results show the opposite effectiveness on the Violence in Crowds dataset. While deep features using Xception achieved accuracy rate ranging from 68 to 70%, Inception-v3 reached 78.88% when using SVM-RBF for training.

Nevertheless, when comparing results from deep features (Table 4.7) with previous results using CENTRIST and HoG (Table 4.4 and Table 4.3), in fact, it is easily noticeable that the results using deep features are inferior to the results gathered through the handcrafted methods used previously. After obtaining poor results by skipping some consecutive frames, we decided to evaluate all frames to confirm that the result was really related to the number of processed frames. Then, using Inception-v3 to extract features from all frames and SVM-RBF for training, we obtained an accuracy rate of 76.83±6.1%. Thus, we conjecture that our dataset was not representative enough to make a model based on deep learning robust.

4.3 Using Coding Techniques

Using methods based on the CENTRIST descriptor, instead of deep learning models, we seek a more representative input feature vector for training and classification. As an alternative, we include an intermediate coding layer between the descriptor extraction and the classifiers. For this, we pass the descriptors through a clustering method, such as K-means, Gaussian Mixture Models, or Sparse-Coding.

During the development of the work, we identified that the *eXtreme Gradient Boosting* (XGB) [166] classifier obtained relevant results in several pattern recognition tasks. In the following sections, results using XGB are also presented, confirming its robustness in comparison to the other classifiers evaluated in our experiments.

We now present results for the three approaches used to encode feature vectors into an alternative coding structure. Initially, in Tables 4.8 and 4.9, we obtained results using Euclidean distance over N clusters obtained using K-means and, respectively, SVM and XGB. In Table 4.10, we have results for Mahalanobis distance [158] over N Gaussians obtained through GMM [159]. Finally, in Table 4.11, we applied Sparse Coding [167] using a different number of components to find the most effective representation.

Table 4.8: Results for Violence in Crowds dataset [13] using K-means at 2.0Hz.

Method	Accuracy (%)
CENTRIST (HEq, K-means-12, SVM-RBF)	79.26±3.7
CENTRIST (HEq, K-means-32, SVM-RBF)	80.86±2.3
CENTRIST (HEq, K-means-64, SVM-RBF)	78.43±5.4
CENTRIST (HEq, K-means-128, SVM-RBF)	80.36±3.9
CENTRIST (HEq, K-means-256, SVM-RBF)	81.70±4.6
CENTRIST (HEq, K-means-512, SVM-RBF)	82.10±4.3
CENTRIST (HEq, K-means-12, PCA+SVM-RBF)	80.48±4.4
CENTRIST (HEq, K-means-32, PCA+SVM-RBF)	78.46±5.3
CENTRIST (HEq, K-means-64, PCA+SVM-RBF)	78.46±5.3
CENTRIST (HEq, K-means-128, PCA+SVM-RBF)	80.48±3.5
CENTRIST (HEq, K-means-256, PCA+SVM-RBF)	82.10±4.3
CENTRIST (HEq, K-means-512, PCA+SVM-RBF)	82.10±4.3
HoG+CENTRIST (HEq, K-means-12, SVM-RBF)	**82.95±4.0**
HoG+CENTRIST (HEq, K-means-24, SVM-RBF)	82.90±5.6
HoG+CENTRIST (HEq, K-means-32, SVM-RBF)	75.48±9.4
HoG+CENTRIST (HEq, K-means-64, SVM-RBF)	68.21±7.2
HoG+CENTRIST (HEq, K-means-256, SVM-RBF)	58.55±3.0
HoG+CENTRIST (HEq, K-means-512, SVM-RBF)	56.11±1.0
HoG+CENTRIST (HEq, K-means-12, PCA+SVM-RBF)	75.48±9.4
HoG+CENTRIST (HEq, K-means-24, PCA+SVM-RBF)	64.53±6.5
HoG+CENTRIST (HEq, K-means-32, PCA+SVM-RBF)	68.56±9.8
HoG+CENTRIST (HEq, K-means-64, PCA+SVM-RBF)	61.70±6.4
HoG+CENTRIST (HEq, K-means-128, PCA+SVM-RBF)	56.43±4.2
HoG+CENTRIST (HEq, K-means-256, PCA+SVM-RBF)	53.60±3.4

Tables 4.8 and 4.9 provide results for K-means for different numbers of clusters using, respectively, SVM and XGB for classification and PCA for dimensionality reduction. From the results of SVM or XGB using CENTRIST, it is possible to observe that, using XGB, the accuracy rate increases as the number of clusters is incremented until reaching a threshold. On the other hand, using SVM, the rate accuracy is not generally improved and, in some cases, can get worse.

Although XGB has obtained results that outperform SVM (85.00% versus 82.10%), the best accuracy is achieved when the number of clusters reaches 256. For SVM, the

Table 4.9: Results using XGB for Violence in Crowds dataset [13] using K-means at 2.0Hz.

Method	Accuracy (%)
CENTRIST (HEq, K-means-12, XGB)	80.11±3.2
CENTRIST (HEq, K-means-32, XGB)	80.95±2.9
CENTRIST (HEq, K-means-64, XGB)	83.33±0.7
CENTRIST (HEq, K-means-256, XGB)	**85.00±3.5**
CENTRIST (HEq, K-means-512, XGB)	83.76±4.1
CENTRIST (HEq, K-means-12, PCA+XGB)	80.93±3.5
CENTRIST (HEq, K-means-32, PCA+XGB)	83.35±3.8
CENTRIST (HEq, K-means-64, PCA+XGB)	81.73±4.3
CENTRIST (HEq, K-means-256, PCA+XGB)	84.18±4.3
CENTRIST (HEq, K-means-512, PCA+XGB)	82.93±3.4

accuracy improves as we increase the number of clusters until reaching 82.10% when using (512). In addition, when analyzing the impact of applying PCA for preprocessing, we can identify that the results in general are lower for both XGB and SVM.

However, when the CENTRIST+HoG combination is used (shown in Table 4.8), the opposite behavior can be observed. The accuracy rate decreases as the number of clusters increases. Thus, our best result was obtained using 12 clusters, respectively with SVM and PCA+SVM: 82.95% and 75.48%.

To compare K-means, we apply GMM as an alternative to encode the feature vector. Instead of using Euclidean distance, we use Mahalanobis distance since we look for a more appropriate distance measure for the Gaussian space [159]. In Table 4.10, we present few results for GMM due to the high computational cost to process each experiment. In general, it requires 5 times more time than using K-means. As in our first experiments, applying background subtraction in the preprocessing step obtained a considerably low result of 56.10%, we chose not to perform other tests using GMM as a preprocessing approach.

Table 4.10: Results for Violence in Crowds dataset [13] using GMM at 2.0Hz.

Method	Accuracy (%)
CENTRIST (HEq, GMM-32, AdaBoost)	82.46±3.0
CENTRIST (HEq, GMM-64, AdaBoost)	**84.10±4.9**
CENTRIST (HEq, GMM-32, SVM-RBF)	81.65±4.8
CENTRIST (HEq, GMM-64, SVM-RBF)	81.65±4.8

Table 4.10 reports results for two different classifiers. Using AdaBoost classifier, the result improves by increasing the number of Gaussians (from 82% using 32 Gaussians to 84% using 64 Gaussians), while using SVM-RBF we obtain the same result (81.65%) for both number of Gaussians (81.65%).

Comparing the results shown in Tables 4.8 and 4.10, we can observe that, using 64 clusters (or Gaussians for GMM) with SVM-RBF, we obtained 78.43% and 81.65%, respectively, however, for 32 clusters (or Gaussians), we obtained 81.73% for K-means and

81.65% for GMM.

Table 4.11 presents results using features encoded through Sparse Coding 2.6.3. When analyzing the Sparse Coding for each approach, we can observe that, for CENTRIST, such as for HoG+CENTRIST, the accuracy rate varies approximately when the number of components is increased from 16 to 512. For CENTRIST, for instance, the results range from 79.65 to 82.96 and from 77.21 to 81.63 when using PCA previously. In addition, for AdaBoost, we obtained results between 83.68 and 85.36. For XGB, the results range from 83.67 to 84.96% without PCA and from 82.03 to 85.33% using PCA.

For HoG+CENTRIST using SVM, the results appear to be lower for a few components and to be improved by increasing this number. For instance, using 12 components, we obtained 56.06%, whereas we reached 82.91% using 128 components. The same occurs when applying PCA previously: from 56.06 to 80.90%. For XGB, the results are still stable using a different number of components (from 32 to 512 components), however, this approach achieves the best result for 64 components: 87.43%.

In addition to the experiments mentioned previously, we try to analyze regions of interest by also evaluating background subtraction through MoG as preprocessing filter, in order to remove the background from the scene and restrict the feature extraction to only parts of the scene where the movement is present.

In Tables 4.12 and 4.13, we compare the results found in the previous experiments for coding and the equivalent of applying MoG for background subtraction before the feature extraction. However, after conducting several experiments, we were able to easily identify that this approach presents poor results in almost all methods that use coding techniques.

4.4 Using Gabor Filter

In relation to previous works that used Gabor filter and obtained expressive results in image recognition (Gdyczynski et al. [168] and Gangwar et al. [169]), we carried out experiments using Gabor as low-pass preprocessing filter to preserve more relevant details.

Table 4.14 presents results in which Gabor before processing CENTRIST and HoG+CENTRIST descriptors for non-coding and coding techniques. Comparing SVM and XGB classifiers using Gabor preprocessing, we observed that CENTRIST and HoG+CENTRIST using Gabor obtained results approximately close to both classifiers. For CENTRIST using SVM, we obtained 86.58% (Table 4.14) against 85.71% (Table 4.2) and, for HoG+CENTRIST and SVM, we obtained 87.78% (Table 4.14) against 87.81% (Table 4.2) without Gabor.

In addition, when evaluating coding techniques such as K-means and Sparse Coding, we obtained 81.68% (shown in Table 4.11) for SVM over K-means using 24 clusters processed with HoG+CENTRIST, against 82.90% and 82.95% using, respectively, 24 and 12 clusters (shown in Table 4.11). For CENTRIST, we obtained 81.68% (Table 4.2) against 71.56% (Table 4.14) using 24 components.

Comparing sparse coding method, for Gabor using 512 components and SVM-RBF, we obtained 82.56% and 83.78% (shown in Table 4.14) for CENTRIST and HoG+CENTRIST, respectively. Furthermore, without Gabor, we obtained 82.91% for HoG+CENTRIST

Table 4.11: Results for Violence in Crowds dataset [13] using Sparse Coding.

Method	Accuracy (%)
CENTRIST (HEq, SparseCod-16, SVM-RBF)	82.48±5.3
CENTRIST (HEq, SparseCod-32, SVM-RBF)	81.73±4.1
CENTRIST (HEq, SparseCod-64, SVM-RBF)	79.65±2.9
CENTRIST (HEq, SparseCod-128, SVM-RBF)	80.86±2.3
CENTRIST (HEq, SparseCod-512, SVM-RBF)	82.96±5.6
CENTRIST (HEq, SparseCod-16, XGB)	83.71±2.4
CENTRIST (HEq, SparseCod-32, XGB)	84.96±2.9
CENTRIST (HEq, SparseCod-64, XGB)	83.75±2.1
CENTRIST (HEq, SparseCod-128, XGB)	83.68±3.1
CENTRIST (HEq, SparseCod-512, XGB)	84.53±2.8
CENTRIST (HEq, SparseCod-16, PCA+SVM-RBF)	77.21±5.7
CENTRIST (HEq, SparseCod-32, PCA+SVM-RBF)	80.80±5.8
CENTRIST (HEq, SparseCod-64, PCA+SVM-RBF)	81.30±1.9
CENTRIST (HEq, SparseCod-128, PCA+SVM-RBF)	81.26±4.4
CENTRIST (HEq, SparseCod-512, PCA+SVM-RBF)	81.63±5.0
CENTRIST (HEq, SparseCod-16, PCA+XGB)	82.03±5.8
CENTRIST (HEq, SparseCod-32, PCA+XGB)	83.33±2.3
CENTRIST (HEq, SparseCod-64, PCA+XGB)	84.95±2.7
CENTRIST (HEq, SparseCod-128, PCA+XGB	84.10±2.9
CENTRIST (HEq, SparseCod-512, PCA+XGB)	**85.33±2.5**
HoG+CENTRIST (HEq, SparseCod-32, AdaBoost)	84.13±2.4
HoG+CENTRIST (HEq, SparseCod-64, AdaBoost)	85.36±3.2
HoG+CENTRIST (HEq, SparseCod-128, AdaBoost)	83.68±5.4
HoG+CENTRIST (HEq, SparseCod-32, SVM-RBF)	56.06±3.7
HoG+CENTRIST (HEq, SparseCod-64, SVM-RBF)	67.03±4.5
HoG+CENTRIST (HEq, SparseCod-128, SVM-RBF)	82.91±8.2
HoG+CENTRIST (HEq, SparseCod-32, XGB)	86.18±2.9
HoG+CENTRIST (HEq, SparseCod-64, XGB)	**87.43±3.4**
HoG+CENTRIST (HEq, SparseCod-128, XGB)	86.60±3.5
HoG+CENTRIST (HEq, SparseCod-512, XGB)	85.38±2.2
HoG+CENTRIST (HEq, SparseCod-32, PCA+SVM-RBF)	56.06±3.7
HoG+CENTRIST (HEq, SparseCod-64, PCA+SVM-RBF)	66.25±4.7
HoG+CENTRIST (HEq, SparseCod-128, PCA+SVM-RBF)	80.90±5.6
HoG+CENTRIST (HEq, SparseCod-128, PCA+XGB)	84.55±3.9
HoG+CENTRIST (HEq, SparseCod-512, PCA+XGB)	83.31±1.6

using 128 components.

4.5 Tests Performed on Hockey Fights Dataset

After obtaining considerable results for the Violence in Crowds dataset, we intend to confirm our results over a secondary crowd-based dataset involving violence. Thus, using the Hockey Fights dataset [14], we performed experiments using CENTRIST-based descriptors,

Table 4.12: Results for Violence in Crowds dataset [13] for K-means and Sparse Coding using Histogram Equalization, compared to applying or not Background Subtraction (Part 1).

Method	Accuracy (%)	Accuracy (%) with MoG
CENTRIST (HEq, K-means-512, SVM-RBF)	82.10±4.3	78.85±3.3
CENTRIST (HEq, K-means-256, PCA+SVM-RBF)	82.10±4.3	77.66±2.6
CENTRIST (HEq, K-means-512, PCA+SVM-RBF)	82.10±4.3	78.06±3.0
CENTRIST (HEq, K-means-12, XGB)	80.11±3.2	79.65±3.5
CENTRIST (HEq, K-means-32, XGB)	80.95±2.9	78.03±5.0
CENTRIST (HEq, K-means-64, XGB)	83.33±0.7	78.81±5.1
CENTRIST (HEq, K-means-256, XGB)	85.00±3.5	80.46±4.2
CENTRIST (HEq, K-means-512, XGB)	83.76±4.1	79.63±3.1
CENTRIST (HEq, K-means-12, PCA+XGB)	80.93±3.5	77.97±4.4
CENTRIST (HEq, K-means-32, PCA+XGB)	83.35±3.8	78.43±4.6
CENTRIST (HEq, K-means-64, PCA+XGB)	81.73±4.3	80.86±2.9
CENTRIST (HEq, K-means-256, PCA+XGB)	84.18±4.3	81.65±3.2
CENTRIST (HEq, K-means-512, PCA+XGB)	82.93±3.4	78.88±4.0
CENTRIST (HEq, SparseCod-64, SVM-RBF)	79.65±2.9	77.63±1.9
CENTRIST (HEq, SparseCod-128, SVM-RBF)	80.86±2.3	77.58±6.3
CENTRIST (HEq, SparseCod-512, SVM-RBF)	82.96±5.6	79.23±2.6
CENTRIST (HEq, SparseCod-64, XGB)	83.75±2.1	81.30±4.9
CENTRIST (HEq, SparseCod-128, XGB)	83.68±3.1	**82.13±3.1**
CENTRIST (HEq, SparseCod-512, XGB)	84.53±2.8	80.48±2.7
CENTRIST (HEq, SparseCod-64, PCA+SVM-RBF)	81.30±1.9	80.10±5.3
CENTRIST (HEq, SparseCod-128, PCA+SVM-RBF)	81.26±4.4	79.61±4.4
CENTRIST (HEq, SparseCod-512, PCA+SVM-RBF)	81.63±5.4	79.68±4.6
CENTRIST (HEq, SparseCod-64, PCA+XGB)	84.95±2.7	79.75±4.3
CENTRIST (HEq, SparseCod-128, PCA+XGB	84.10±2.9	75.55±5.5
CENTRIST (HEq, SparseCod-512, PCA+XGB)	**85.33±2.5**	78.91±5.4

such as when using the Violence in Crowds dataset, and achieved promising results, as shown in Table 4.16. Our method obtained an accuracy rate of 90.19% when extracting features over the full frame content and using only PCA and SVM. When background subtraction is previously applied to specific regions of the frame, the resulting accuracy was 90.89%. By visually inspecting some videos in both datasets, we identified some differences that may have contributed to the background subtraction being more effective for the Hockey Fights dataset:

1. it contains better resolution and less dark images. Consequently, the Mixture of Gaussians can discriminate the actor region with more quality. Even by visual inspection, it is hard to notice the actors in the Violent in Crowds dataset.
2. there are fewer actors on the scene in most of the videos in the Hockey Fights dataset. Consequently, the background subtraction is less likely to consider many actors together as the background itself.

Table 4.13: Results for Violence in Crowds dataset [13] for K-means and Sparse Coding using Histogram Equalization, compared to applying or not Background Subtraction (Part 2).

Method	Accuracy (%)	Accuracy (%) with MoG
HoG+CENTRIST (HEq, K-means-12, SVM-RBF)	82.95±4.0	75.28±5.3
HoG+CENTRIST (HEq, K-means-24, SVM-RBF)	82.90±5.6	75.23±4.9
HoG+CENTRIST (HEq, K-means-32, SVM-RBF)	75.48±9.4	75.30±5.4
HoG+CENTRIST (HEq, K-means-64, SVM-RBF)	68.21±7.2	71.98±5.7
HoG+CENTRIST (HEq, K-means-64, SVM-RBF)	68.21±7.2	75.25±3.8
HoG+CENTRIST (HEq, K-means-12, PCA+SVM-RBF)	75.48±9.4	47.93±6.7
HoG+CENTRIST (HEq, K-means-24, PCA+SVM-RBF)	64.53±6.5	59.00±5.5
HoG+CENTRIST (HEq, K-means-32, PCA+SVM-RBF)	68.56±9.8	58.55±3.3
HoG+CENTRIST (HEq, K-means-64, PCA+SVM-RBF)	61.70±6.4	54.90±3.1
HoG+CENTRIST (HEq, SparseCod-32, AdaBoost)	84.13±2.4	80.90±3.7
HoG+CENTRIST (HEq, SparseCod-64, AdaBoost)	85.36±3.2	80.58±5.0
HoG+CENTRIST (HEq, SparseCod-128, AdaBoost)	83.68±5.4	80.50±3.8
HoG+CENTRIST (HEq, SparseCod-32, SVM-RBF)	56.06±3.7	59.68±4.0
HoG+CENTRIST (HEq, SparseCod-64, SVM-RBF)	67.03±4.5	71.55±7.7
HoG+CENTRIST (HEq, SparseCod-128, SVM-RBF)	82.91±8.2	**82.10±4.8**
HoG+CENTRIST (HEq, SparseCod-32, XGB)	86.18±2.9	79.70±1.4
HoG+CENTRIST (HEq, SparseCod-64, XGB)	**87.43±3.4**	81.21±5.0

When applying the combination of HoG+CENTRIST, we slightly improved the result using SVM: 90.99%. In addition, we also conducted experiments with two other classifiers, Random Forests and AdaBoost, whose results achieved 90.60% and 92.29%, respectively. The latter was our best accuracy rate for the Hockey Fights dataset, using the entire frame to extract the descriptors.

Similarly to our results for Violence in Crowds dataset, our method reached results very close to those available in the literature at the time these experiments were performed: 94.40% with SRC [145] and 94.20% with MoWLD [107]. Some time later, however, MoIWLD, a method developed by Zhang et al. [11] achieved 96.80%, which demonstrated that our approach would need to be refined.

4.6 Automatic Threshold Selection as Voting Criteria

After having observed individually the results of several video recordings, we identified that the confidence in the decision in the majority of the evaluations varies between 40% and 70%. For this reason, we also decided to evaluate the threshold that would maximize the hit rate. To do this after performing the training stage using the training set, we evaluate the confidence in each frame that is considered violent in the training set.

Then, we found the threshold that splits the classes more efficiently, evaluating the hit

Table 4.14: Results for Violence in Crowds dataset [14] by adding Gabor as preprocessing filter.

Method	Accuracy (%)
CENTRIST (HEq+Gabor, SVM-RBF)	86.58±3.8
CENTRIST (HEq+Gabor, PCA+SVM-RBF)	82.96±5.3
CENTRIST (HEq+Gabor, XGB)	**86.58±4.5**
CENTRIST (HEq+Gabor, PCA+XGB)	79.28±3.8
HoG+CENTRIST (HEq+Gabor, SVM-RBF)	87.78±4.3
HoG+CENTRIST (HEq+Gabor, PCA+SVM-RBF)	75.21±3.8
HoG+CENTRIST (HEq+Gabor, XGB)	84.61±3.5
HoG+CENTRIST (HEq+Gabor, PCA+XGB)	79.76±4.8
CENTRIST (HEq+Gabor, K-means-32, SVM-RBF)	78.00±6.4
CENTRIST (HEq+Gabor, K-means-64, SVM-RBF)	81.26±5.4
CENTRIST (HEq+Gabor, K-means-512, SVM-RBF)	**81.68±3.5**
HoG+CENTRIST (HEq+Gabor, K-means-32, SVM-RBF)	71.56±2.8
HoG+CENTRIST (HEq+Gabor, K-means-64, SVM-RBF)	66.6±4.9
HoG+CENTRIST (HEq+Gabor, K-means-512, SVM-RBF)	58.1±3.8
CENTRIST (HEq+Gabor, PCA+K-means-32,SVM-RBF)	76.75±5.9
CENTRIST (HEq+Gabor, PCA+K-means-64,SVM-RBF)	78.38±7.8
CENTRIST (HEq+Gabor, PCA+K-means-512,SVM-RBF)	80.85±5.0
HoG+CENTRIST (HEq+Gabor, PCA+K-means-32, SVM-RBF)	56.88±4.2
HoG+CENTRIST (HEq+Gabor, PCA+K-means-64, SVM-RBF)	54.06±2.9
HoG+CENTRIST (HEq+Gabor, PCA+K-means-512, SVM-RBF)	52.06±3.2
CENTRIST (HEq+Gabor, SparseCod-32, SVM-RBF)	83.66±4.1
CENTRIST (HEq+Gabor, SparseCod-64, SVM-RBF)	78.88±3.4
CENTRIST (HEq+Gabor, SparseCod-512, SVM-RBF)	82.56±2.4
HoG+CENTRIST (HEq+Gabor, SparseCod-32,SVM-RBF)	59.73±4.8
HoG+CENTRIST (HEq+Gabor, SparseCod-64,SVM-RBF)	69.50±5.4
HoG+CENTRIST (HEq+Gabor, SparseCod-512,SVM-RBF)	83.78±5.5
CENTRIST (HEq+Gabor, PCA+SparseCod-32, SVM-RBF)	84.06±5.9
CENTRIST (HEq+Gabor, PCA+SparseCod-64, SVM-RBF)	80.43±6.0
CENTRIST (HEq+Gabor, PCA+SparseCod-512, SVM-RBF)	75.26±4.5
HoG+CENTRIST (HEq+Gabor, PCA+SparseCod-32, SVM-RBF)	60.13±6.3
HoG+CENTRIST (HEq+Gabor, PCA+SparseCod-64, SVM-RBF)	68.68±6.1
HoG+CENTRIST (HEq+Gabor, PCA+SparseCod-512, SVM-RBF)	**84.61±4.7**

Table 4.15: Results for Violence in Crowds dataset [13] using different preprocessing filters.

Method	Raw (%)	HEq (%)	MoG (%)	HEq+Gabor (%)
HoG+CENTRIST (SVM-RBF)	88.25±3.1	86.96±3.8	87.40±1.4	87.78±4.3

rate using all confidence values between 0.05 and 0.95, varying with step 0.05. We use this threshold to evaluate features obtained in the test fold. Finally, for each result, we use the voting criteria to determine which class the video footage will belong to.

Table 4.17 presents the results for extraction in the entire image using automatic

Table 4.16: Results for Hockey Fights dataset [14].

Method	Accuracy (%)
CENTRIST (HEq+MoG, PCA+SVM-RBF)	91.19±2.3
CENTRIST (MoG, PCA+SVM-RBF)	90.89±1.8
CENTRIST (Blur+MoG, SVM-RBF)	91.19±2.3
CENTRIST (Blur+MoG, PCA+SVM-RBF)	90.39±2.9
CENTRIST (Blur, PCA+SVM-RBF)	90.69±2.5
CENTRIST (Blur+HEq, PCA+SVM-RBF)	90.09±2.2
CENTRIST (Blur+HEq, SVM-RBF)	89.89±2.4
CENTRIST (Blur+HEq+MoG, SVM-RBF)	89.89±2.4
CENTRIST (Blur+HEq+MoG, PCA+SVM-RBF)	89.89±1.7
HoG+CENTRIST (Blur+HEq, SVM-RBF)	80.40±4.4
HoG+CENTRIST (HEq, PCA+SVM-RBF)	90.29±2.2
HoG+CENTRIST (HEq, SVM-RBF)	90.99±2.7
HoG+CENTRIST (HEq, AdaBoost)	**92.29±3.7**
HoG+CENTRIST (HEq+MoG, AdaBoost)	**92.29±3.7**
HoG+CENTRIST (HEq+MoG, Random Forest)	90.60±3.0

Table 4.17: Results for the dataset Violence in Crowds dataset [14] using automatic threshold selection.

Method	Accuracy (%)
HoG+CENTRIST (HEq, K-means-12, PCA+SVM-RBF, AutoThresh-0.5)	68.16±10.3
HoG+CENTRIST (HEq, K-means-32, PCA+SVM-RBF, AutoThresh-0.5)	68.56±9.8
HoG+CENTRIST (HEq, AdaBoost, AutoThresh-0.56)	88.25±5.6
HoG+CENTRIST (HEq, K-means-32, AdaBoost, AutoThresh-0.435)	**87.43±4.9**
HoG+CENTRIST (HEq, K-means-64, AdaBoost, AutoThresh-0.57)	86.63±3.8
HoG+CENTRIST (HEq, XGB, AutoThresh-0.5)	83.28±6.1
HoG+CENTRIST (HEq, K-means-12,XGB, AutoThresh-0.5)	87.01±4.6
HoG+CENTRIST (HEq, K-means-32,XGB, AutoThresh-0.54)	85.38±4.9
HoG+CENTRIST (HEq, K-means-64,XGB, AutoThresh-0.5)	86.61±5.1

selection for voting threshold. The AutoThresh-X parameter indicates which threshold was identified by our intermediate step as confidence X to maximize the hit rate in the training set.

In Table 4.17, we can observe that, in most results, the threshold 0.5, or a very close value, was identified as the best confidence to discriminate the classes. Therefore, we assume that we could continue to use 0.5 as threshold value instead of trying to determine it for each experiment. In any case, this evidence does not mean that we could not use the automatic threshold selection criteria to refine models whose results are already promising using 0.5 as a threshold value.

4.7 Block Evaluation using Sliding Window

As mentioned in Section 3.2, feature extraction was also performed using block evaluation. To do this, we initially tackled Sliding Window approach using blocks with 128×128 pixels to extract features using CENTRIST and HoG+CENTRIST. Then, we evaluate different dimensions of blocks to identify which one would improve accuracy.

In Table 4.18, we can easily observe that the results using Sliding Window outperform the results based on the holistic method. In addition, we also applied a coding technique before the classification step to find how the Sliding Window approach would behave. However, we noticed that, using Sparse Coding technique, the efficiency was reduced from 89.86% to 87.41%.

Table 4.18: Results for Violence in Crowds dataset [13] using Sliding Window.

Method	Accuracy (%)
CENTRIST (HEq, blocks-64, SVM-RBF)	89.85±3.3
CENTRIST (HEq, blocks-96, SVM-RBF)	89.88±5.0
HoG+CENTRIST (HEq, blocks-64, SVM-RBF)	91.05±1.6
HoG+CENTRIST (HEq, blocks-64, PCA+SVM-RBF)	**91.46±1.4**
HoG+CENTRIST (HEq, blocks-72, SVM-RBF)	90.26±1.8
HoG+CENTRIST (HEq, blocks-96, SVM-RBF)	89.86±2.7
HoG+CENTRIST (HEq, blocks-96, PCA+SVM-RBF)	89.03±1.5
HoG+CENTRIST (HEq, blocks-96, SparseCod-64, SVM-RBF)	87.41±2.3
HoG+CENTRIST (HEq, blocks-96, PCA+SparseCod-64, SVM-RBF)	89.00±3.1

4.7.1 Results using Sliding Window for Hockey Fights Dataset

For the Sliding Window approach, according to the results reported in Table 4.19, the CENTRIST and HoG+CENTRIST methods reached 91.69% and 92.79%, respectively, on the Hockey Fights dataset. The latter was approximately superior to the result of the evaluation of entire frames (92.29%, as shown in Table 4.16).

In addition to AdaBoost, we used XGB, identified during the development of our research, as an alternative boosting classifier. Through these experiments, we confirmed that the results using XGB generally slightly outperformed AdaBoost. Moreover, for the Hockey Fights dataset, unlike the results for the Violent in Crowds dataset, we can see that the best results were obtained using a larger block size instead of 64×64 pixels. This fact allows us to assume that not always reducing the block size will improve the accuracy. After performing a visual inspection of the datasets, we can identify the actors' height are generally different and could probably have been cut when using smaller block sizes.

4.8 Comparing Results to HoG Descriptor

To identify that the efficiency of our combination was not obtained only due to the HoG property, we also performed experiments using the HoG descriptor individually to extract

Table 4.19: Results for Hockey Fights dataset [14] using Sliding Window.

Method	Accuracy (%)
CENTRIST (HEq, blocks-64, AdaBoost)	90.79±1.9
CENTRIST (HEq, blocks-64, SVM-RBF)	90.99±2.4
CENTRIST (HEq, blocks-64, XGB)	91.19±2.3
CENTRIST (HEq, blocks-96, AdaBoost)	89.69±1.7
CENTRIST (HEq, blocks-96, SVM-RBF)	91.69±2.9
CENTRIST (HEq, blocks-96, XGB)	91.29±2.0
HoG+CENTRIST (HEq, blocks-64, AdaBoost)	90.79±1.8
HoG+CENTRIST (HEq, blocks-64, SVM-RBF)	90.99±2.4
HoG+CENTRIST (HEq, blocks-64, XGB)	91.59±2.7
HoG+CENTRIST (HEq, blocks-96, AdaBoost)	90.19±1.7
HoG+CENTRIST (HEq, blocks-96, SVM-RBF)	**92.79±3.0**
HoG+CENTRIST (HEq, blocks-96, XGB)	91.29±2.0

features from the video frames. For this, we use both approaches to analyze frames: entire content and Sliding Window over blocks.

Table 4.20: Results for Violence in Crowds dataset [14] for HoG descriptor using Sliding Window.

Method	Accuracy (%)
HoG (HEq, SVM-RBF)	78.46±5.7
HoG (HEq, K-means-64+SVM-RBF)	68.28±8.6
HoG (HEq, SparseCod-64+SVM-RBF)	69.11±8.6
HoG (HEq+MoG, SVM-RBF)	77.25±3.7
HoG (HEq+MoG, K-means-64+SVM-RBF)	63.51±8.4
HoG (HEq+MoG, SparseCod-64+SVM-RBF)	74.83±7.5
HoG (HEq, PCA+SVM-RBF)	78.08±4.5
HoG (HEq, PCA+K-means-64+SVM-RBF)	64.20±7.4
HoG (HEq, PCA+SparseCod+SVM-RBF)	78.08±4.5
HoG (HEq, XGB)	76.05±4.6
HoG (HEq, K-means-64+XGB)	80.48±7.2
HoG (HEq, SparseCod-64+XGB)	**80.48±3.5**
HoG (HEq, PCA+XGB)	78.43±7.5
HoG (HEq, PCA+K-means-64+XGB)	73.16±3.4
HoG (HEq, PCA+SparseCod-64+XGB)	78.01±5.1

Table 4.20 presents results for the HoG descriptor. In summary, we compare the results of experiments using the HoG descriptor only to the results provided by the CENTRIST descriptor 4.6. Then, we can observe that the effectiveness of HoG is approximately less when applied individually. It is important to mention that the coding techniques using eXtreme Gradient Boosting (78.46±5.7%) obtained results slightly superior to the results achieved using only SVM (**80.48±3.5**%).

The HoG descriptor was also applied individually using Sliding Window followed by

Table 4.21: Results for Violence in Crowds dataset [14] for HoG using grid without and with automatic threshold selection.

Method	Accuracy (%)
HoG (HEq, blocks-96, SVM-RBF)	**82.15±4.6**
HoG (HEq, blocks-64, SVM-RBF)	80.91±3.1
HoG (HEq, blocks-96, K-means-64+SVM-RBF)	80.48±3.0
HoG (HEq, blocks-64, K-means-64+SVM-RBF)	79.68±3.7
HoG (HEq, blocks-96, SparseCod-64+SVM-RBF)	81.76±6.2
HoG (HEq, blocks-64, SparseCod-64+SVM-RBF)	78.55±6.5
HoG (HEq, blocks-96, XGB)	78.10±8.6
HoG (HEq, blocks-64, XGB)	75.65±4.3
HoG (HEq, blocks-96, K-means-64+XGB)	76.08±5.3
HoG (HEq, blocks-64, K-means-64+XGB)	72.78±2.7
HoG (HEq, blocks-96, SparseCod-64+XGB)	73.23±9.0
HoG (HEq, blocks-64, SparseCod-64+XGB)	72.03±5.2
HoG (HEq, blocks-96, PCA+XGB)	77.31±6.2
HoG (HEq, blocks-64, PCA+XGB)	76.48±6.4
HoG (HEq, blocks-96, PCA+K-means-64+XGB)	76.91±6.6
HoG (HEq, blocks-64, PCA+K-means-64+XGB)	73.58±3.2
HoG (HEq, blocks-96, PCA+SparseCod-64+XGB)	73.61±6.6
HoG (HEq, blocks-64, PCA+SparseCod-64+XGB)	72.41±4.6
HoG (HEq, blocks-96, XGB, AutoThresh-0.665)	82.08±2.2
HoG (HEq, blocks-64, SVM-RBF, AutoThresh-0.74)	82.05±3.3
HoG (HEq, blocks-96, PCA+XGB, AutoThresh-0.72)	**83.75±4.8**
HoG (HEq, blocks-64, SVM-RBF, PCA+AutoThresh-0.77)	80.50±6.0

coding techniques. Table 4.21 shows results using only feature extraction. Although the extraction approach using the entire frame seems to have improved effectiveness, they do not outperform the results of experiments using only the CENTRIST descriptor (Table 4.6).

Another point to be highlighted is that, when the automatic threshold selection is performed, for XGB using Sliding Window approach with a block size of 96×96, the accuracy improved from 78% to 82% and from 77% to 83% using PCA. However, for SVM-RBF, observing the standard deviation, the effectiveness seems more stable near 88% when using HoG+CENTRIST for all different frame rates.

4.9 Results using Optical Flow-based Descriptors

In this section, we show results related to techniques that explore optical flow through the transition between frames to evaluate when violence occurs. Results are presented for two different approaches: the first is based on the Histogram of the Dense Optical Flow (HoDenseOF) and the second is based on the application of the CENTRIST descriptor on the image generated from the Dense Optical Flow (CENTRISTofDenseOF). In the following, we also combine the mentioned methods with the CENTRIST and CENTRIST+HoG

descriptors.

Table 4.22: Results for Violence in Crowds dataset [14] using descriptors based on Optical Flow.

Method	Accuracy (%)
HoDenseOF (HEq, XGB)	65.05±4.5
HoDenseOF (HEq, SVM-RBF)	82.15±4.6
HoDenseOF (HEq, SVM-linear)	63.06±7.1
CENTRISTofDenseOF (HEq, XGB)	82.15±4.6
CENTRISTofDenseOF (HEq, SVM-RBF)	82.15±4.6
CENTRISTofDenseOF (HEq, SVM-linear)	82.15±4.6
HoDenseOF+CENTRIST (HEq, XGB)	83.31±4.7
HoDenseOF+CENTRIST (HEq, SVM-RBF)	85.41±3.2
HoDenseOF+HoG+CENTRIST (HEq, SVM-RBF)	**89.06±3.6**
HoDenseOF+HoG+CENTRIST (HEq, SVM-RBF)	86.60±3.0
CENTRISTOfDenseOF+HoG+CENTRIST (HEq, XGB)	85.40±3.8
CENTRISTOfDenseOF+HoG+CENTRIST (HEq, SVM-RBF)	87.86±4.0

By comparing the results shown in Table 4.22, it is possible to identify that the CENTRIST of the Dense Optical Flow, by itself, obtains results certainly better than using Histogram of the Dense Optical Flow. Moreover, by comparing them to the results of Table 4.6, the combination of Histogram of Dense Optical Flow with CENTRIST does not seem to aggregate information to CENTRIST. Furthermore, when combining Histogram of Dense Optical Flow with CENTRIST and HoG, the results appear to be inferior to using only CENTRIST+HoG.

In Table 4.18, we present results of experiments in which we applied Sliding Window and used the combination of Histogram and CENTRIST of the Optical Flow. Table 4.23 presents results for both combinations. For HoDenseOF+HoG+CENTRIST and CENTRISTOfDenseOF+HoG+CENTRIST, we obtained 89.83% and 89.85%, respectively, against 90.26% from HoG+CENTRIST using the same 72×72 block size.

Table 4.23: Results for Violence in Crowds dataset [14] using grid with descriptors based on Optical Flow.

Method	Accuracy (%)
HoDenseOF+HoG+CENTRIST (HEq, blocks-72, SVM-RBF)	89.06±3.6
HoDenseOF+HoG+CENTRIST (HEq, blocks-96, SVM-RBF)	89.83±1.2
CENTRISTOfDenseOF+HoG+CENTRIST (HEq,blocks-72,SVM-RBF)	89.85±2.1
CENTRISTOfDenseOF+HoG+CENTRIST (HEq,blocks-96,SVM-RBF)	**89.85±1.1**

4.10 Results using Block Selection Criteria based on Optical Flow Evaluation

As previous results related Background Subtraction using MoG showed substantial decline compared to the standard Sliding Window approach, we noticed that the background and the camera motion significantly impact training and, consequently, our final results. Based on this fact, we addressed how we could identify the most relevant parts of the scene and focused our efforts on evaluating the difference between the background motion and the motion produced by the actors in the scene.

Taking this into account, we raise the assumption that whenever the camera moves in one direction and considering that no actor is moving in the scene, all the pixels in the scene move in the same direction and speed. Consequently, if there is movement in the scene, the background pixels should keep moving according to the camera motion and only the movement of the actors could be discriminated in comparison to the rest of the scene. Therefore, we could identify regions of interest by evaluating the movement in specific regions of the scene in comparison to the global motion. To do so, we compare the regions of the scene whose average optical flow is sufficiently different compared to the global average optical flow.

In this section, we explore methods to analyze the relevance of the blocks selected by the Sliding Window approach. In the following subsection, experimental results are presented to filter the least relevant blocks for the scene.

4.10.1 Results for Violence in Crowds Dataset using Sliding Window using Block Filtering

In order to restrict more relevant regions of interest, we compare the values obtained between the global optical flow and the local optical flow for each block of the Sliding Window.

Table 4.24 contains the results for the Sliding Window using optical flow and different approaches to evaluate the optical flow in order to obtain a threshold value to filter the most relevant blocks according to the criteria used. In addition to the Otsu's method, we also use *mean* and *median* to evaluate a threshold value, since they can be considered simpler than our previous approach. In this table, the following legend: local block measure $>$ global measure, where local block measure is the value obtained through a statistical method applied to evaluate each block region, whereas global measure is the threshold value obtained using a method to assess the entire frame region. Thus, the block will be passed to the next pipeline layer, if the local measure is equal to or greater than the global measure.

In these experiments, we evaluated different approaches that have already been used in Section 3.3. These functions showed similar results when evaluating the scene. In general, we can identity that, compared to the global threshold, using values greater than the threshold is more efficient than calculating local metrics. This statement was true for all global metrics. Furthermore, when evaluating the best local metric, the mean local optical flow obtained good results compared to the local median. In general, when visually

inspecting the video recordings, in the majority of the videos, actors are arranged over much of the scene. Thus, when methods were used to remove many blocks at once, we significantly reduced the amount of relevant descriptors produced for training.

Table 4.24: Results for Violence in Crowds dataset [14] using Sliding Window with filters based on statistics from the Optical flow and Otsu's method.

Method	Accuracy (%)
HoG (HEq, blocks-96, SVM-RBF, Otsu)	82.15±4.6
CENTRIST (HEq, blocks-96, SVM-RBF, Avg>Median)	89.45±3.8
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Avg>Median)	88.63±4.1
CENTRIST (HEq, blocks-96, SVM-RBF, Any>Median)	87.35±4.1
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Any>Median)	87.35±4.1
CENTRIST (HEq, blocks-96, SVM-RBF, Avg>Avg)	89.03±2.9
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Avg>Avg)	89.03±2.9
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Median>Otsu)	62.16±3.5
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Avg>Otsu)	89.03±2.9
HoG+CENTRIST (HEq, blocks-96, SVM-RBF, Any>Otsu)	90.23±2.4
HoG+CENTRIST (HEq, blocks-64, PCA+SVM-RBF, Any>Otsu)	91.45±1.5
HoG+CENTRIST (HEq, blocks-64, SVM-RBF, Any>Otsu)	**91.48±2.2**

4.11 Experiments using Cross Dataset

In order to evaluate the effectiveness of our proposed methods in a more challenging scenario, we executed cross-dataset experiments using both Violence in Crowds and Hockey Fights datasets.

Results when training with the Violence in Crowds dataset and evaluating on the Hockey Fight dataset are presented in Table 4.25. It is possible to observe that there is a small difference between applying blocks of 64×64 or 96×96 and applying block filtering. In summary, we obtained an accuracy close to 51%, as well as precision and recall rates of 76.92% and 4.00%, respectively.

Table 4.25: Cross-dataset results for HoG+CENTRIST and Sliding Window on Hockey Fights dataset using Violence in Crowds as training set.

Method	Accuracy	Precision	Recall
HEq, blocks-96	51.00	77.78	2.80
HEq, blocks-96+OptFlowDenseFilter (Any>Avg)	51.00	77.78	2.80
HEq, blocks-96+OptFlowDenseFilter (Any>Otsu)	51.00	75.00	3.00
HEq, blocks-96+OptFlowMagnitFilter (Any>Avg)	51.00	72.97	5.40
HEq, blocks-64	**51.70**	72.97	5.40
HEq, blocks-64+OptFlowDenseFilter (Any>Avg)	51.40	76.92	4.00
HEq, blocks-64+OptFlowDenseFilter (Any>Otsu)	51.40	76.92	4.00
HEq, blocks-64+OptFlowMagnitFilter (Any>Avg)	51.40	76.92	4.00

On the other hand, Table 4.26 presents the results when training on the Hockey Fights dataset and evaluating on the Violence in Crowds dataset.

Table 4.26: Cross-dataset results for HoG+CENTRIST and Sliding Window on Violence in Crowds dataset using Hockey Fights as training set.

Method	Accuracy	Precision	Recall
HEq, blocks-96	55.28	52.79	100.00
HEq, blocks-96+OptFlowMagnitFilter (Any>Avg)	55.28	52.79	100.00
HEq, blocks-96+OptFlowDenseFilter (Any>Otsu)	55.69	53.02	100.00
HEq, blocks-96+OptFlowDenseFilter (Any>Avg)	55.69	53.02	100.00
HEq, blocks-64	55.69	53.02	100.00
HEq, blocks-64+OptFlowMagnitFilter (Any>Avg)	**57.72**	54.19	100.00
HEq, blocks-64+OptFlowDenseFilter (Any>Otsu)	58.13	54.42	100.00
HEq, blocks-64+OptFlowDenseFilter (Any>Avg)	58.13	54.42	100.00

Comparing the results in Tables 4.25 and 4.26, we observe that, even tough applying the same technique, the dataset used for training directly impacts the violence detection. This can be evidenced by observing the difference between the recall rates when training with each dataset. When using Hockey Fights for training, the obtained recall rate is almost 100%, allowing us to conclude that the false positive rate is extremely high.

In Tables 4.27 and 4.28, we report cross-dataset results for Violence in Crowds and Hockey Fights by classifying each frame individually and adopting voting criteria through results of all descriptors from the same frame.

Table 4.27: Cross-dataset results for HoG+CENTRIST, Sliding Window and Classification per Frame on Hockey Fights dataset using Violence in Crowds as training set.

Method	Accuracy	Precision	Recall
HEq, blocks-96	52.51	77.96	7.22
HEq, blocks-96+OptFlowDenseFilter (Any>Avg)	52.21	85.71	5.57
HEq, blocks-96+OptFlowDenseFilter (Any>Otsu)	52.20	85.71	5.57
HEq, blocks-96+OptFlowMagnitFilter (Any>Avg)	52.21	85.71	5.57
HEq, blocks-64	**52.71**	74.35	8.52
HEq, blocks-64+OptFlowDenseFilter (Any>Avg)	52.14	80.91	5.91
HEq, blocks-64+OptFlowDenseFilter (Any>Otsu)	52.21	81.82	5.97
HEq, blocks-64+OptFlowMagnitFilter (Any>Avg)	52.14	80.91	5.91

In Table 4.27, we have results using Sliding Window approach with blocks of 64×64 and 96×96 pixels. In addition, techniques using block filtering were also applied. In general even having obtained precision considerably higher than expected (77.96%), the recall rate is extremely low (7.22%). When comparing only the Sliding Window, we noticed that the precision worsens when the blocks size is reduced. A similar behavior can be observed when applying block filtering, although precision has increased (85.71%) and recall has decreased (5.57%).

When evaluating the reverse cross-dataset experiment, we observe that the opposite behavior occurs. Although the results related to recall are expressive (98%), the precision

Table 4.28: Cross-dataset results for HoG+CENTRIST, Sliding Window and Classification per Frame on Violence in Crowds dataset using Hockey Fights as training set.

Method	Accuracy	Precision	Recall
HEq, blocks-96	62.16	60.80	98.54
HEq, blocks-96+OptFlowMagnitFilter (Any>Avg)	63.69	62.34	98.35
HEq, blocks-96+OptFlowDenseFilter (Any>Otsu)	63.64	62.33	98.25
HEq, HEq,blocks-96+OptFlowDenseFilter (Any>Avg)	63.69	62.34	98.35
HEq, blocks-64	64.77	62.41	99.27
HEq, blocks-64+OptFlowMagnitFilter (Any>Avg)	**66.01**	63.76	99.28
HEq, blocks-64+OptFlowDenseFilter (Any>Otsu)	65.97	63.73	99.28
HEq, blocks-64+OptFlowDenseFilter (Any>Avg)	**66.01**	63.76	99.28

is relatively low (62%). Comparing results using blocks with 64×64 and 96×96 pixels, it is possible to notice that both precision and recall rates were slightly improved for 62.41% and 99.27%, respectively. After filtering blocks using sliding window approach, we notice a small improvement for precision (63.76%) and recall (99.28%) using HoG+CENTRIST.

From the cross-dataset experiments, we identified a sensibility of our method using the CENTRIST descriptor. The dataset used for training makes our model highly sensitive to scenario variations. We obtained high precision and low recall when using the Violent Flow dataset as training set and Hockey Fights dataset as test set. On the other hand, we obtained lower precision and higher recall when using the Hockey Fights dataset as training set. Therefore, in addition to motion occurrence, the actions used for training are very important to evaluate the real-world scenarios.

4.12 Final Considerations

In Tables 4.29 and 4.30, we summarize the best results for each approach based on the CENTRIST descriptor. By comparing the results obtained via CENTRIST and HoG+CENTRIST, we can observe that they reached results that overcome most traditional handcrafted techniques for the Violence in Crowds dataset (shown in Table 2.2) and for the Hockey Fights dataset (shown in Table 2.1).

It is possible to see that the sliding window approach achieved competitive results for violence detection on both datasets, especially when optical flow block filtering was added to the pipeline. When applying PCA to reduce the dimensionality of HoG+CENTRIST, we achieved 91.46%. When applying block filtering using Otsu's method to obtain the global threshold to filter blocks, we obtained 91.45±1.5 and 91.48±2.2 with and without PCA, respectively. The latter result was the highest we obtained in this work for the Violent in Crowds dataset [13].

In Table 4.30, we have the best results for the Hockey Fights dataset [14]. Although most of the techniques related to this work have focused on the Violent in Crowds dataset [13], we can observe that the combination of CENTRIST and HoG achieved considerably high results, even without requiring the sliding window approach. Using different classifiers, we obtained an accuracy rate greater than 92%. Then, using the sliding

Table 4.29: Accuracy for the Violence in Crowds [13] and Hockey Fights [14] datasets.

Method	Violence in Crowds (%)
Baseline - ViF (SVM) [13]	81.30±0.21
3D-ResNet [151]	94.54±4.1
LHoG+LHOF+Bow [110]	94.60±1.7
BiConvLSTM SpatialTemporalEnconder [139]	96.32±1.5
Semantic Correspondence [144]	97.69
Ullah et al. [143]	**98.00**
CENTRIST (SVM)	85.75±5.6
CENTRIST (PCA+SVM)	86.16±2.8
CENTRIST (SGD)	87.80±1.8
CENTRIST (MoG, PCA+SVM)	83.73±2.2
CENTRIST (MoG, PCA+SGD)	83.30±2.5
CENTRIST (MoG, SVM)	82.51±2.7
CENTRIST (MoG, SGD)	83.35±1.3
CENTRIST Multiscale (SVM)	84.90±5.4
CENTRIST Multiscale (PCA+SVM)	85.81±2.6
CENTRIST Multiscale (MoG, SVM)	81.70±2.6
CENTRIST Multiscale (MoG, SGD)	82.96±2.5
HoG+CENTRIST (SVM)	86.96±3.1
HoG+CENTRIST (AdaBoost)	86.61±4.1
HoG+CENTRIST (SGD)	85.81±5.8
HoG+CENTRIST (Random Forest)	86.20±6.0
HoG+CENTRIST (MoG, SVM)	87.40±1.4
HoG+CENTRIST (MoG, SGD)	85.00±4.3
HoG+CENTRIST (MoG, AdaBoost)	80.16±4.7
HoG+CENTRIST (MoG, Random Forest)	74.85±3.8
CENTRIST (blocks-96, SVM)	89.88±5.0
CENTRIST (blocks-64, SVM)	89.85±3.3
CENTRIST (blocks-64, PCA+Adaboost)	87.76±2.4
HoG+CENTRIST (blocks-96, SVM)	89.86±2.7
HoG+CENTRIST (blocks-96, SVM, Any>Otsu)	90.23±2.4
HoG+CENTRIST (blocks-72, SVM)	90.26±1.8
HoG+CENTRIST (blocks-64, SVM)	91.05±1.6
HoG+CENTRIST (blocks-64, SVM, Any>Otsu)	**91.48±2.2**
HoG+CENTRIST (blocks-64, PCA+SVM)	91.46±1.4
HoG+CENTRIST (blocks-64, PCA+SVM, Any>Otsu)	91.45±1.5

window approach, we slightly improved our best result for the Hockey Fights dataset [14] to 92.79% using 96×96 block size with sliding window. This shows that the block size cannot be reduced indefinitely to improve the performance of our model. When visually comparing the differences between videos from the Violent in Crowds and Hockey Fights datasets, we can notice that the actors in the Hockey Fights dataset are considerably larger and well defined compared to the actors in the Violent in Crowds dataset. Therefore, our

main assumption is that small block sizes are not efficient for cropping regions that are considerably smaller than the actor height. It could only crop pieces of the image that do not contain any relevant information. Furthermore, against the behavior shown for Violent in Crowds dataset, even though it improved most of the results for the sliding window, the approach using block filtering based on optical flow did not outperform the best result for our secondary dataset.

Table 4.30: Accuracy for the Hockey Fights [14] dataset.

Method	Hockey Fights (%)
ConvLSTM [148]	97.10±0.5
FightNet [149]	97.01
BiConvLSTM SpatialTemporalEnconder [139]	97.90±0.3
CNN+LSTM (Darknet19) [136]	98.00±0.5
BiConvLSTM SpatialEnconder [139]	**98.10±0.5**
CENTRIST (SVM)	89.99±1.7
CENTRIST (PCA+SVM)	88.69±1.8
CENTRIST (SGD)	88.49±7.1
CENTRIST(MoG, PCA+SVM)	91.19±2.3
CENTRIST (MoG, PCA+SVM)	91.19±2.3
CENTRIST (MoG, SGD)	90.69±3.6
CENTRIST Multiscale (PCA+SVM)	88.19±3.8
CENTRIST Multiscale (SVM)	89.59±3.8
CENTRIST Multiscale (MoG, SGD)	89.39±2.1
HoG+CENTRIST (SVM)	90.99±2.8
HoG+CENTRIST (AdaBoost)	92.29±3.7
HoG+CENTRIST (Random Forest)	89.50±2.7
HoG+CENTRIST (MoG, SVM)	90.99±2.7
HoG+CENTRIST (MoG, AdaBoost)	92.29±3.7
HoG+CENTRIST (MoG, Random Forest)	90.60±3.0
CENTRIST (blocks-96, SVM)	91.69±2.9
CENTRIST (blocks-64, SVM)	90.69±3.1
CENTRIST (blocks-64, PCA+Adaboost)	89.19±2.1
HoG+CENTRIST (blocks-96, SVM)	**92.79±3.0**
HoG+CENTRIST (blocks-72, SVM)	91.99±2.75
HoG+CENTRIST (blocks-96, SVM, Any>Otsu)	85.59±3.4
HoG+CENTRIST (blocks-64, SVM)	90.99±2.4

Chapter 5

Conclusions and Future Work

In this work, we presented a method for detecting violence events in video scenes based on the CENTRIST descriptor. Several experiments were carried out on two benchmarks to demonstrate the effectiveness of our method.

Although the proposed approach has not overcome the best results reported in the literature, the accuracy rates are very competitive, using a conceptually simple technique capable of capturing discriminative characteristics for violence classification in video scenes. In addition, its combination with preprocessing strategies, such as histogram equalization and background subtraction, provides improvements to the descriptor.

Some relevant experimental results obtained with our method are summarized as follows:

(a) Violence in Crowds dataset:

- entire frame approach: CENTRIST (86.96%) and HOG+CENTRIST (87.45%).
- block-based approach: CENTRIST (89.88%) and HOG+CENTRIST (91.46%).
- block-based approach using optical flow filter (Any>Otsu): HOG+CENTRIST (90.23%) using 96×96 and HOG+CENTRIST (91.48%) using 64×64.

(b) Hockey Fights dataset:

- entire frame approach: CENTRIST (90.69%) and HOG+CENTRIST (92.29%)
- block-based approach: CENTRIST (91.69%) and HOG+CENTRIST (92.79%)

From these results, it is possible to draw some conclusions related to the experiments conducted using the entire video frame as input to the feature extraction stage:

- CENTRIST has its performance increased when combined with the HOG descriptor.
- Background subtraction slightly improved the results when evaluating the entire frame and when using sliding window for Hockey Fights dataset.

Additionally, as mentioned in Section 3.3, we identified by visual inspection the background interference and motion when identifying the action in the scene. By using pre-trained models to evaluate different datasets, as shown in the experiments using the

cross datasets, described in Section 4.11, we can conclude that the CENTRIST-based descriptors are still sensitive to different types of actions. When evaluating the Hockey Fights dataset, we can notice that having different action patterns on the scene directly implies the precision result. Furthermore, by analyzing video recording (or movies) whose content is not fully related to crowded scenes, it is possible to notice that identifying regions of interest is still a problem when evaluating the scene.

Overall, although the combination CENTRIST+HoG is useful for characterizing crowd scenes, it is important to highlight that when the object detection is relatively small in the image, we cannot accurately detect the abnormality in the scene. In addition, as reported by additional experiments using cross datasets (shown in Section 4.11, the precision is highly affected due to the characteristics of the dataset). Thus we can state that using CENTRIST or HoG for training through Violence in Crowds dataset would not be so effective for evaluating another datasets, such as Hockey Fights. This assumption allows us to conclude that our proposed descriptor would not have enough effect to discriminate between normal and unknown abnormal actions based on a limited dataset.

Moreover, although we tried to discriminate regions of interest in the image, we were unable to effectively identify these regions through optical flow only. In many cases, especially when the camera is being moved too far, it is not possible to discriminate between actors and background. In these cases, the descriptors are extracted from samples of part of the background and used incorrectly in the training process. Consequently, even using different features or training methods, their performance has been considerably affected.

In conclusion, we can answer the research questions formulated in Section 1.4:

- Are holistic techniques effective to detect violence, especially in crowd scenes?
  Answer: Holistic features, such as HoG, CENTRIST and HoF, presented reasonable results when evaluating the Violence in Crowds and Hockey Fights datasets, even when extracted directly from the entire frame. Thus, we conclude that the application of the holistic feature can be effective in discriminating crowd scenes.

- How does CENTRIST descriptor perform to detect violence?
  Answer: CENTRIST has obtained results that overcomes the traditional handcraft descriptors such as HoG and HOF to evaluate crowed scenes. In Table 4.6 we observe the CENTRIST outperforms traditional feature descriptors, and can be enhanced when combined to descriptors with complementary characteristics.

- Which approaches using CENTRIST could improve the effectiveness?
  Answer: Using small portions of the scene proved to be more effective than evaluating the entire video frame. In general, when the camera motion is less intense, the method is more robust. Using block filtering with less restrictive approaches (for instance, Local(Any)>Global(Otsu) and Local(Any)>Global(mean)) can remove less relevant blocks.

- Can the combination between CENTRIST and other descriptor improve violence detection?

Answer: Regarding the experiments reported in Section 4.2, using various frame capture rates and different classifiers, it is possible to observe in Table 4.1 that the results combining HoG and CENTRIST surpassed most results using only CENTRIST and this fact is more evident when comparing CENTRIST+HoG to results using only HoG. In Table 4.5 we can clearly see that the results using HoG only are considerably lower than using CENTRIST+HoG.

In conclusion, although our method obtained competitive results for crowded scenes when evaluating a specific dataset, we have identified some weaknesses that are relevant when considering applying it to real-world scenes. Among them we can highlight:

- the cross-dataset validation showed that our method is not suitable for generalizing different types of violence scenes.
- the sliding window approach depends on the actor dimensions used to train the model; therefore, the block size cannot be reduced too much.
- the optical flow block filtering was unable to detect relevant interest regions due to camera motion.

Based on these statements, new descriptors and representations must be investigated to make the training step more robust and generalist to identify abnormal actions and violent behavior, as well as more invariant to motion influence. In addition, multiscale could be used in combination with the sliding window approach in order to make the model more robust when smaller blocks are used. Deep learning networks have potential to explore such events when large amount of data is available.

Bibliography

[1] M. Jain, H. Jegou, and P. Bouthemy, "Better Exploiting Motion for Better Action Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 2555–2562.

[2] M. Satya, "Histogram of Oriented Gradients," 2016, http://www.learnopencv.com/histogram-of-oriented-gradients.

[3] M. Krishna, "Gabor Filter: A Practical Overview," 2014, https://cvtuts.wordpress.com/2014/04/27/gabor-filters-a-practical-overview.

[4] S. S. Beauchemin and J. L. Barron, "The Computation of Optical Flow," *ACM Computing Surveys*, vol. 27, no. 3, pp. 433–466, 1995.

[5] W. Li, V. Mahadevan, and N. Vasconcelos, "Anomaly Detection and Localization in Crowded Scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 18–32, 2014.

[6] H. Mousavi, S. Mohammadi, A. Perina, R. Chellali, and V. Murino, "Analyzing Tracklets for the Detection of Abnormal Crowd Behavior," in *IEEE Winter Conference on Applications of Computer Vision*. Waikoloa Beach, HI, USA: IEEE, 2015, pp. 148–155.

[7] S. Mohammadi, H. Kiani, A. Perina, and V. Murino, "Violence Detection in Crowded Scenes using Substantial Derivative," in *IEEE International Conference on Advanced Video and Signal Based Surveillance*. Colorado Springs, CO, USA: IEEE, 2015, pp. 1–6.

[8] X. Wang, M. Gao, X. He, X. Wu, and Y. Li, "An Abnormal Crowd Behavior Detection Algorithm Based on Fluid Mechanics," *Journal of Computers*, vol. 9, no. 5, pp. 1144–1149, 2014.

[9] R. Mehran, A. Oyama, and M. Shah, "Abnormal Crowd Behavior Detection using Social Force Model," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 935–942.

[10] M. Marsden, K. McGuinness, S. Little, and N. E. O'Connor, "Holistic Features for Real-time Crowd Behaviour Anomaly Detection," in *IEEE International Conference on Image Processing*. Phoenix, AZ, USA: IEEE, 2016, pp. 918–922.

[11] T. Zhang, W. Jia, X. He, and J. Yang, "Discriminative Dictionary Learning with Motion Weber Local Descriptor for Violence Detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2016.

[12] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," *arXiv preprint*, pp. 1610–02 357, 2017.

[13] T. Hassner, Y. Itcher, and O. Kliper-Gross, "Violent Flows: Real-time Detection of Violent Crowd Behavior," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. Providence, RI, USA: IEEE, 2012, pp. 1–6.

[14] E. B. Nievas, O. D. Suarez, G. B. García, and R. Sukthankar, "Violence Detection in Video using Computer Vision Techniques," in *International Conference on Computer Analysis of Images and Patterns*. Springer, 2011, pp. 332–339.

[15] M. F. Alcantara, T. P. Moreira, and H. Pedrini, "Real-time Action Recognition based on Cumulative Motion Shapes," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, Italy, May 2014, pp. 2941–2945.

[16] J. Cai, X. Tang, and G. Feng, "Learning Pose Dictionary for Human Action Recognition," in *International Conference on Pattern Recognition*, vol. 1, Stockholm, Sweden, Aug. 2014, pp. 381–386.

[17] V. Kantorov and I. Laptev, "Efficient Feature Extraction, Encoding and Classification for Action Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, Jun. 2014.

[18] B. Liang and L. Zheng, "3D Motion Trail Model based Pyramid Histograms of Oriented Gradient for Action Recognition," in *International Conference on Pattern Recognition*, vol. 1, Stockholm, Sweden, Aug. 2014, pp. 1952–1957.

[19] M. Alcantara, H. Pedrini, and Y. Cao, "Human Action Classification based on Silhouette Indexed Interest Points for Multiple Domains," *International Journal of Image and Graphics*, vol. 17, no. 3, pp. 1 750 018_1–1 750 018_27, Jul. 2017.

[20] H. Tacon, A. Brito, H. Chaves, M. Vieira, S. Villela, H. Maia, D. Concha, and H. Pedrini, "Multi-Stream Architecture with Symmetric Extended Visual Rhythms for Deep Learning Human Action Recognition," in *15th International Conference on Computer Vision Theory and Applications (VISAPP)*, Valletta, Malta, Feb. 2020, pp. 351–358.

[21] A. Santos, H. Maia, M. Souza, M. Vieira, and H. Pedrini, "Fuzzy Fusion for Two-Stream Action Recognition," in *15th International Conference on Computer Vision Theory and Applications (VISAPP)*, Valletta, Malta, Feb. 2020, pp. 117–123.

[22] H. Maia, M. Souza, A. Santos, H. P. H. Tacon, A. Brito, H. Chaves, M. Vieira, and S. Villela, "Learnable Visual Rhythms Based on the Stacking of Convolutional

Neural Networks for Action Recognition," in *18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Boca Raton-FL, USA, Dec. 2019, pp. 1794–1799.

[23] H. Tacon, A. Brito, H. Chaves, M. Vieira, S. Villela, H. Maia, D. Concha, and H. Pedrini, "Human Action Recognition Using Convolutional Neural Networks with Symmetric Time Extension of Visual Rhythms," in *19th International Conference on Computational Science and its Applications (ICCSA)*, Saint Petersburg, Russia, Jul. 2019, pp. 351–366.

[24] A. Santos and H. Pedrini, "Spatio-Temporal Video Autoencoder for Human Action Recognition," in *14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*.

[25] D. Concha, H. Maia, H. Pedrini, H. Tacon, A. Brito, H. Chaves, and M. Vieira, "Multi-Stream Convolutional Neural Networks for Action Recognition in Video Sequences Based on Adaptive Visual Rhythms," in *17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando-FL, USA, Dec. 2018, pp. 473–480.

[26] J. Wu and J. M. Rehg, "CENTRIST: A Visual Descriptor for Scene Categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1489–1501, 2011.

[27] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 886–893.

[28] F. Souza and H. Pedrini, "Detection of Violent Events in Video Sequences based on Census Transform Histogram," in *30th Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2017, pp. 323–329.

[29] Y. Gao, H. Liu, X. Sun, C. Wang, and Y. Liu, "Violence Detection using Oriented Violent Flows," *Image and Vision Computing*, vol. 48, pp. 37–41, 2016.

[30] K. Lloyd, D. Marshall, S. C. Moore, and P. L. Rosin, "Detecting Violent Crowds using Temporal Analysis of GLCM Texture," *arXiv preprint 1605.05106*, 2016.

[31] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[32] G. Piriou, P. Bouthemy, and J.-F. Yao, "Recognition of Dynamic Video Contents with Global Probabilistic Models of Visual Motion," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3417–3430, 2006.

[33] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning Realistic Human Actions from Movies," in *IEEE Conference on Computer Vision and Pattern Recognition*. Anchorage, AK, USA: IEEE, 2008, pp. 1–8.

[34] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid, "Evaluation of Local Spatio-Temporal Features for Action Recognition," in *British Machine Vision Conference*. BMVA Press, 2009, pp. 124–1.

[35] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action Recognition by Dense Trajectories," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 3169–3176.

[36] I. Laptev, "On Space-Time Interest Points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, Sep. 2005.

[37] G. Willems, T. Tuytelaars, and L. Van Gool, "An Efficient Dense and Scale-Invariant Spatio-Temporal Interest Point Detector," in *European Conference on Computer Vision*. Springer, 2008, pp. 650–663.

[38] A. Hervieu, P. Bouthemy, and J.-P. Le Cadre, "A Statistical Video Content Recognition Method using Invariant Features on Object Trajectories," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1533–1543, 2008.

[39] P. Matikainen, M. Hebert, and R. Sukthankar, "Trajectons: Action Recognition through the Motion Analysis of Tracked Features," in *IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, 2009, pp. 514–521.

[40] R. Messing, C. Pal, and H. Kautz, "Activity Recognition using the Velocity Histories of Tracked Keypoints," in *12th International Conference on Computer Vision*. IEEE, 2009, pp. 104–111.

[41] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li, "Hierarchical Spatio-Temporal Context Modeling for Action Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 2004–2011.

[42] T. Brox and J. Malik, "Object Segmentation by Long Term Analysis of Point Trajectories," in *European Conference on Computer Vision*. Springer, 2010, pp. 282–295.

[43] S. Wu, O. Oreifej, and M. Shah, "Action Recognition in Videos Acquired by a Moving Camera using Motion Decomposition of Lagrangian Particle Trajectories," in *IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 1419–1426.

[44] A. Gaidon, Z. Harchaoui, and C. Schmid, "Recognizing Activities with Cluster-Trees of Tracklets," in *British Machine Vision Conference*. BMVA Press, 2012, pp. 30–1.

[45] J. Sivic and A. Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," in *Ninth International Conference on Computer Vision*. IEEE, 2003, p. 1470.

[46] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf, "Motion Interchange Patterns for Action Recognition in Unconstrained Videos," in *European Conference on Computer Vision*. Springer, 2012, pp. 256–269.

[47] H. Uemura, S. Ishikawa, and K. Mikolajczyk, "Feature Tracking and Motion Compensation for Action Recognition." in *British Machine Vision Conference*. BMVA Press, 2008, pp. 1–10.

[48] J. Little and J. Boyd, "Recognizing People by their Gait: The Shape of Motion," *Videre: Journal of Computer Vision Research*, vol. 1, no. 2, pp. 1–32, 1998.

[49] L. Wang, T. Tan, W. Hu, and H. Ning, "Automatic Gait Recognition based on Statistical Shape Analysis," *IEEE Transactions on Image Processing*, vol. 12, no. 9, pp. 1120–1131, 2003.

[50] J. P. Foster, M. S. Nixon, and A. Prügel-Bennett, "Automatic Gait Recognition using Area-based Metrics," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2489–2497, 2003.

[51] J. P. Gupta, D. Polytool, N. Singh, and V. B. Aemwal, "Analysis of Gait Pattern to Recognize the Human Activities," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 2, no. 7, pp. 7–16, 2014.

[52] I. Laptev and P. Pérez, "Retrieving Actions in Movies," in *IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[53] J. Wu and J. M. Rehg, "Centrist: A Visual Descriptor for Scene Categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1489–1501, 2010.

[54] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence," in *European Conference on Computer Vision*. Springer, 1994, pp. 151–158.

[55] Y. Cao, S. Pranata, and H. Nishimura, "Local Binary Pattern Features for Pedestrian Detection at Night/Dark Environment," in *IEEE 18th International Conference on Image Processing*. IEEE, 2011, pp. 2053–2056.

[56] R. K. McConnell, "Method of and Apparatus for Pattern Recognition," Jan. 1986, US Patent 4,567,610.

[57] D. Marr and E. Hildreth, "Theory of Edge Detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, pp. 187–217, 1980.

[58] D. G. Lowe, "Object Recognition from Local Scale-Invariant Features," in *Seventh IEEE International Conference on Computer Vision*, vol. 2. IEEE, 1999, pp. 1150–1157.

[59] S. Belongie, G. Mori, and J. Malik, "Matching with Shape Contexts," in *Statistics and Analysis of Shapes*. Springer, 2006, pp. 81–105.

[60] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[61] I. Fogel and D. Sagi, "Gabor Filters as Texture Discriminator," *Biological Cybernetics*, vol. 61, no. 2, pp. 103–113, 1989.

[62] "Gabor Filter," 2020, https://en.wikipedia.org/wiki/Gabor_filter.

[63] Strouthopoulos, C and Papamarkos, N and Chamzas, C, "Identification of Text-only Areas in Mixed-type Documents," *Engineering Applications of Artificial Intelligence*, vol. 10, no. 4, pp. 387–401, 1997.

[64] Pati, Peeta Basa and Raju, S. Sabari and Pati, Nishikanta and Ramakrishnan, A.G., "Gabor Filters for Document Analysis in Indian Bilingual Documents," in *International Conference on Intelligent Sensing and Information Processing*. IEEE, 2004, pp. 123–126.

[65] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding Facial Expressions with Gabor Wavelets," in *Third IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE, 1998, pp. 200–205.

[66] Raja, K Bommanna and Madheswaran, M and Thyagarajah, K, "Texture Pattern Analysis of Kidney Tissues for Disorder Identification and Classification using Dominant Gabor Wavelet," *Machine Vision and Applications*, vol. 21, no. 3, pp. 287–300, 2010.

[67] J. G. Daugman, "Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles," *Vision Research*, vol. 20, no. 10, pp. 847–856, 1980.

[68] D. Fleet and Y. Weiss, "Optical Flow Estimation," in *Handbook of Mathematical Models in Computer Vision*. Springer, 2006, pp. 237–257.

[69] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of Optical Flow Techniques," *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, 1994.

[70] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," vol. 2, Vancouver, BC, Canada, Aug. 1981, pp. 674–679.

[71] N. Dalal, B. Triggs, and C. Schmid, "Human Detection using Oriented Histograms of Flow and Appearance," in *European Conference on Computer Vision*. Springer, 2006, pp. 428–441.

[72] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of Oriented Optical Flow and Binet-cauchy Kernels on Nonlinear Dynamical Systems for the Recognition of Human Actions," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1932–1939.

[73] P. KaewTraKulPong and R. Bowden, "An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection," in *Video-based Surveillance Systems*. Springer, 2002, pp. 135–144.

[74] "BackgroundSubtractorMOG Class Reference," 2019, https://docs.opencv.org/ref/2.4/db/dcf/classcv_1_1BackgroundSubtractorMOG.html.

[75] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[76] M. Sezgin, "Survey over Image Thresholding Techniques and Quantitative Performance Evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–168, 2004.

[77] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.-R. Mullers, "Fisher Discriminant Analysis with Kernels," in *IEEE Signal Processing Society Workshop - Neural Networks for Signal Processing*. Ieee, 1999, pp. 41–48.

[78] G. F. Jenks, "The data model concept in statistical mapping," *International yearbook of cartography*, vol. 7, pp. 186–190, 1967.

[79] "Otsu's Method," 2020, https://en.wikipedia.org/wiki/Otsu%27s_method.

[80] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Fifth Annual Workshop on Computational Learning Theory*. ACM, 1992, pp. 144–152.

[81] "Support-Vector Machine," 2020, https://en.wikipedia.org/wiki/Support-vector_machine.

[82] A. Vedaldi and A. Zisserman, "Sparse Kernel Approximations for Efficient Classification and Detection," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2320–2327.

[83] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[84] N. Cesa-Bianchi, "Analysis of Two Gradient-based Algorithms for On-line Regression," in *Tenth Annual Conference on Computational Learning Theory*. ACM, 1997, pp. 163–170.

[85] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[86] R. E. Schapire and Y. Freund, *Boosting: Foundations and Algorithms*. MIT Press, 2012.

[87] H. Drucker and C. Cortes, "Boosting Decision Trees," in *8th International Conference on Neural Information Processing Systems*. Denver, Colorado: MIT Press, 1995, pp. 479–485.

[88] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-Class AdaBoost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[89] T. Chen and C. Guestrin, "Xgboost: A Scalable Tree Boosting System," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794.

[90] "XGBoost 1.0.0," 2020, https://xgboost.readthedocs.io/en/latest/python/index.html.

[91] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," in *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14. Oakland, CA, USA., 1967, pp. 281–297.

[92] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means Clustering Algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[93] M. West and M. D. Escobar, *Hierarchical Priors and Mixture Models, with Application in Regression and Density Estimation*. Institute of Statistics and Decision Sciences, Duke University, 1993.

[94] "Gaussian Mixture Models," 2019, https://scikit-learn.org/stable/modules/mixture.html.

[95] S. S. Chen and P. S. Gopalakrishnan, "Clustering via the Bayesian Information Criterion with Applications in Speech Recognition," in *IEEE International Conference on Acoustics, Speech and Signal*, vol. 2. IEEE, 1998, pp. 645–648.

[96] P. Bofill and M. Zibulevsky, "Underdetermined Blind Source Separation using Sparse Representations," *Signal Processing*, vol. 81, no. 11, pp. 2353–2362, 2001.

[97] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online Dictionary Learning for Sparse Coding," in *26th Annual International Conference on Machine Learning*, 2009, pp. 689–696.

[98] "Sparse Coding with a Precomputed Dictionary," 2020, https://scikit-learn.org/stable/auto_examples/decomposition/plot_sparse_coding.html.

[99] Y. Cao, H. Yang, and C. Li, "Abnormal Crowd Behavior Detection Based on LBP-weighted Social Force Model [J]," *Video Engineering*, vol. 21, p. 043, 2012.

[100] Y. Yin, Q. Liu, and S. Mao, "Global Anomaly Crowd Behavior Detection Using Crowd Behavior Feature Vector," *International Journal of Smart Home*, vol. 9, no. 12, pp. 149–160, 2015.

[101] E. L. M. Sousa, "CENTRIST3D: A Spatio-Temporal Descriptor for Abnormality Detection in Crowd Videos," Master's thesis, Instituto de Computação, Universidade Estadual de Campinas, Campinas-SP, 2017.

[102] J. Nam, M. Alghoniemy, and A. H. Tewfik, "Audio-visual Content-based Violent Scene Characterization," in *IEEE International Conference on Image Processing*, vol. 1. Chicago, IL, USA: IEEE, 1998, pp. 353–357.

[103] T. Giannakopoulos, A. Pikrakis, and S. Theodoridis, "A Multimodal Approach to Violence Detection in Video Sharing Sites," in *20th IEEE International Conference on Pattern Recognition*. Istanbul, Turkey: IEEE, 2010, pp. 3244–3247.

[104] P. Bilinski and F. Bremond, "Human Violence Recognition and Detection in Surveillance Videos," in *13th IEEE International Conference on Advanced Video and Signal Based Surveillance*. IEEE, 2016, pp. 30–36.

[105] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao, "WLD: A Robust Local Image Descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1705–1720, 2010.

[106] T. Zhang, Z. Yang, W. Jia, B. Yang, J. Yang, and X. He, "A New Method for Violence Detection in Surveillance Scenes," *Multimedia Tools and Applications*, pp. 1–23, 2015.

[107] T. Zhang, W. Jia, B. Yang, J. Yang, X. He, and Z. Zheng, "MoWLD: A Robust Motion Image Descriptor for Violence Detection," *Multimedia Tools and Applications*, pp. 1–20, 2015.

[108] D. Moreira, S. Avila, M. Perez, D. Moraes, V. Testoni, E. Valle, S. Goldenstein, and A. Rocha, "Temporal Robust Features for Violence Detection," in *IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2017, pp. 391–399.

[109] D. Moreira, S. Avila, M. Perez, D. Moraes, V. Testoni, E. Valle, S. Goldenstein, and Rocha, "Pornography Classification: The Hidden Clues in Video Spacetime," *Forensic Science International*, vol. 268, pp. 46–61, 2016.

[110] P. Zhou, Q. Ding, H. Luo, and X. Hou, "Violence Detection in Surveillance Video using Low-level Features," *PLOS One*, vol. 13, no. 10, p. e0203668, 2018.

[111] B. Lohithashva, V. M. Aradhya, and D. Guru, "Violent Video Event Detection Based on Integrated LBP and GLCM Texture Features," *International Information and Engineering Technology Association*, pp. 179–187, 2020.

[112] M. Partio, B. Cramariuc, M. Gabbouj, and A. Visa, "Rock Texture Retrieval using Gray Level Co-occurrence Matrix," in *5th Nordic Signal Processing Symposium*, vol. 75. Citeseer, 2002.

[113] Y. LeCun, L. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, and P. Simard, "Learning Algorithms for Classification: A Comparison on Handwritten Digit Recognition," *Neural Networks: The Statistical Mechanics Perspective*, vol. 261, p. 276, 1995.

[114] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[115] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[116] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[117] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[118] A. Toshev and C. Szegedy, "Deeppose: Human Pose Estimation via Deep Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1653–1660.

[119] R. A. Güler, N. Neverova, and I. Kokkinos, "Densepose: Dense human pose estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7297–7306.

[120] L. Fei-Fei, A. Karpathy, T. Leung, S. Shetty, R. Sukthankar, and G. Toderici, "Large-Scale Video Classification with Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[121] N. Wang and D.-Y. Yeung, "Learning a Deep Compact Image Representation for Visual Tracking," in *Advances in Neural Information Processing Systems*, 2013, pp. 809–817.

[122] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a Deep Convolutional Network for Image Super-Resolution," in *European Conference on Computer Vision*. Springer, 2014, pp. 184–199.

[123] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 1–9.

[124] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv preprint arXiv:1502.03167*, 2015.

[125] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[126] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-RESNET and the Impact of Residual Connections on Learning," in *The Association for the Advancement of Artificial Intelligence*, vol. 4, 2017, p. 12.

[127] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[128] S. Mondal, S. Pal, S. K. Saha, and B. Chanda, "Violent/Non-Violent Video Classification based on Deep Neural Network," in *Ninth International Conference on Advances in Pattern Recognition*. IEEE, 2017, pp. 1–6.

[129] J. Shi, "Good Features to Track," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1994, pp. 593–600.

[130] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[131] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[132] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal Segment Networks for Action Recognition in Videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[133] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A Large Video Database for Human Motion Recognition," in *International Conference on Computer Vision*. IEEE, 2011, pp. 2556–2563.

[134] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild," *arXiv preprint arXiv:1212.0402*, 2012.

[135] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah, "The THUMOS Challenge on Action Recognition for Videos "In the Wild"," *Computer Vision and Image Understanding*, vol. 155, pp. 1–23, 2017.

[136] S. Ammar, M. Anjum, T. Rounak, M. Islam, and T. Islam, "Using Deep Learning Algorithms to Detect Violent Activities," Ph.D. dissertation, BRAC University, 2019.

[137] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent Neural Network based Language Model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[138] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[139] Hanson, Alex and Pnvr, Koutilya and Krishnagopal, Sanjukta and Davis, Larry, "Bidirectional Convolutional LSTM for the Detection of Violence in Videos," in *European Conference on Computer Vision*, 2018.

[140] M. Perez, A. C. Kot, and A. Rocha, "Detection of Real-world Fights in Surveillance Videos," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2019, pp. 2662–2666.

[141] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[142] A. Montes, A. Salvador, S. Pascual, and X. Giro-i Nieto, "Temporal Activity Detection in Untrimmed Videos with Recurrent Neural Networks," *arXiv preprint arXiv:1608.08128*, 2016.

[143] F. U. M. Ullah, A. Ullah, K. Muhammad, I. U. Haq, and S. W. Baik, "Violence Detection using Spatio-temporal Features with 3D Convolutional Neural Network," *Sensors*, vol. 19, no. 11, p. 2472, 2019.

[144] C. Gu, X. Wu, and S. Wang, "Violent Video Detection Based on Semantic Correspondence," *IEEE Access*, vol. 8, pp. 85 958–85 967, 2020.

[145] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust Face Recognition via Sparse Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[146] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning Deep Representations of Appearance and Motion for Anomalous Event Detection," *arXiv preprint 1510.01553*, 2015.

[147] Z. Dong, J. Qin, and Y. Wang, "Multi-Stream Deep Networks for Person to Person Violence Detection in Videos," in *Chinese Conference on Pattern Recognition*. Springer, 2016, pp. 517–531.

[148] S. Sudhakaran and O. Lanz, "Learning to Detect Violent Videos using Convolutional Long Short-Term Memory," in *14th IEEE International Conference on Advanced Video and Signal Based Surveillance*. IEEE, 2017, pp. 1–6.

[149] P. Zhou, Q. Ding, H. Luo, and X. Hou, "Violent Interaction Detection in Video based on Deep Learning," in *Journal of Physics: Conference Series*, vol. 844, no. 1. IOP Publishing, 2017, p. 012044.

[150] L. Yeffet and L. Wolf, "Local Trinary Patterns for Human Action Recognition," in *IEEE 12th International Conference on Computer Vision*. Kyoto, Japan: IEEE, 2009, pp. 492–497.

[151] K. Gkountakos, K. Ioannidis, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "A Crowd Analysis Framework for Detecting Violence Scenes," in *International Conference on Multimedia Retrieval*, 2020, pp. 276–280.

[152] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution Gray-scale and Rotation Invariant Texture Classification with Local Binary Patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[153] "OpenCV-Python Tutorials: Dense Optical Flow in OpenCV," 2013, https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_video/py_lucas_kanade/py_lucas_kanade.html.

[154] D. N. Bhat and S. K. Nayar, "Ordinal Measures for Image Correspondence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 4, pp. 415–423, 1998.

[155] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, and M. Isard, "Tensorflow: A System for Large-Scale Machine Learning," in *12th USENIX Symposium on Operating Systems Design and Implementation*, vol. 16, 2016, pp. 265–283.

[156] F. Chollet, "Keras: Deep Learning Library for Theano and Tensorflow," *https://keras.io/k*, vol. 7, no. 8, p. T1, 2015.

[157] P.-E. Danielsson, "Euclidean Distance Mapping," *Computer Graphics and Image Processing*, vol. 14, no. 3, pp. 227–248, 1980.

[158] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The Mahalanobis Distance," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000.

[159] D. Ververidis and C. Kotropoulos, "Gaussian Mixture Modeling by Exploiting the Mahalanobis Distance," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 2797–2811, 2008.

[160] I. Jolliffe, *Principal Component Analysis*. New York: Springer-Verlag, 2002.

[161] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge university press, 2000.

[162] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Proceedings in Computational Statistics*. Springer, 2010, pp. 177–186.

[163] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, "Adaptive Histogram Equalization and its Variations," *Computer Vision, Graphics, and Image Processing*, vol. 39, no. 3, pp. 355–368, 1987.

[164] K. Ito and K. Xiong, "Gaussian Filters for Nonlinear Filtering Problems," *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 910–927, 2000.

[165] Iandola, Forrest N and Han, Song and Moskewicz, Matthew W and Ashraf, Khalid and Dally, William J and Keutzer, Kurt, "SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and < 0.5 MB Model Size," *arXiv preprint arXiv:1602.07360*, 2016.

[166] T. Chen, T. He, M. Benesty, V. Khotilovich, and Y. Tang, "Xgboost: Extreme Gradient Boosting," *R package version 0.4-2*, pp. 1–4, 2015.

[167] C. Liu, Y. Yang, and Y. Chen, "Constructing Visual Vocabularies using Sparse Coding for Action Recognition," in *IEEE International Conference on Information Engineering and Computer Science*. IEEE, 2009, pp. 1–4.

[168] C. M. Gdyczynski, A. Manbachi, S. Hashemi, B. Lashkari, and R. S. C. Cobbold, "On Estimating the Directionality Distribution in Pedicle Trabecular Bone from micro-CT images," *Physiological Measurement*, vol. 35, no. 12, p. 2415, 2014.

[169] A. Gangwar and A. Joshi, "Local Gabor Rank Pattern (LGRP): A Novel Descriptor for Face Representation and Recognition," in *IEEE International Workshop on Information Forensics and Security*. IEEE, 2015, pp. 1–6.