UNIVERSIDADE BEIRA INTERIOR
Engenharia

Sistemas Não Cooperativos para Registo de Assiduidade em Ambiente de Sala de Aula

Alexandre Daniel Ramos Fonseca

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor Hugo Pedro Martins Carriço Proença
Co-orientador: Prof. Doutor Pedro Ricardo Morais Inácio

julho de 2021

Acknowledgements

As I reach the end of this dissertation, I would like to thank all those who have allowed me to get this far.

First of all, I would like to thank Professor Hugo Proença for all his availability throughout this work, for directing me to the best possible solutions and for his essential clarifications for the conclusion of this work, continuously transmitting to me the knowledge that allowed me to evolve a lot in my knowledge throughout this work. I would also like to express my gratitude to Professor Pedro Inácio for having helped me in the various start-up processes of this project and the constant motivation. My deepest thanks to both of them!

To all my family, namely my parents, I would like to express my deepest gratitude. Their support throughout this whole process was decisive to reach the end, not only in this dissertation but in all my path. Everything I am today I owe to them and, for that, all thanks are few. Thank you very much for always believing in me and for always being there.

To my lifelong friends, Joana Soares and Sofia Ascensão, I would like to thank for all the support throughout my academic path, leading me to have been able to get this far. Thank you for believing in me, even in times when I do not and for always having a friendly word to give, making my day better.

I would also like to thank my friends Ana Agostinho and Francisca Nina for always making me smile and always making me believe that I will succeed.

To Pedro Batista, my greatest thanks for listening to me, for the technical opinions on the work and for always being there.

Likewise, I would also like to thank my friends who have not been particularly mentioned here but who were also essential for all my development and their companionship and support. I would also like to thank all the members of Socialab, where I did most of this work, for the excellent environment created and for their total availability whenever I asked for help. A special thanks to all the volunteers who participated in this work and without whom it would not have been possible.

Last but not least, I would like to thank the Informatics Department of the University of Beira Interior for the financial support that made this project possible.

Resumo

Ao longo dos anos, o abandono escolar, o universitário em particular, tem sido sempre um tema em grande destaque. Com o avanço de várias áreas tecnológicas, assim como da Inteligência Artificial e da Aprendizagem Automática temos necessariamente de pensar em maneiras de ajudar a mitigar este problema. Se há fatores que não podemos controlar, como os económicos, há outros onde a nossa ação pode ser dirigida. Um dos fatores que permite avaliar o risco de abandono escolar é a assiduidade dos estudantes. Embora, naturalmente, estes dados possam ser analisados manualmente para fazer estas deteções, seria mais eficiente ter um sistema que fosse capaz de registar esta assiduidade, uma vez que a capacidade humana para os analisar é finita e muitas vezes apenas consegue inferir esta situação demasiado tarde. Naturalmente que um sistema que registe apenas assiduidade não irá ser capaz de dar uma resposta definitiva, mas será um primeiro passe importante. Deste modo, um sistema que seja capaz de conciliar a deteção de uma pessoa e da sua face enquanto é capaz de monitorizar de forma constante o sítio onde a pessoa está, para ser sempre capaz de a identificar mesmo que mude de sítio em conjunto com o seu reconhecimento facial, parecem ser fatores determinantes para levar a bom porto um sistema deste calibre. Este tipo de sistemas está, geralmente, muito relacionado com a qualidade dos dados e das suas anotações, pelo que é extraordinariamente importante ser capaz de recolher ou obter dados de qualidade que auxiliem na resolução dos diversos problemas apresentados.

Tendo em atenção aquilo que foi sendo descrito, o principal objetivo desta dissertação passa por tentar dar início à resolução do problema do abandono escolar, nomeadamente através do estudo, validação e teste de diversos métodos estado-da-arte no que diz respeito à área da deteção de objetos, nomeadamente de pessoas e faces, mas também de tracking. O mesmo trabalho terá de ser realizado nos métodos de reconhecimento facial, sendo capaz no final de poder indicar os melhores métodos estado-da-arte de cada tarefa. Como referido no parágrafo anterior, uma limitação grande a este tipo de tarefas é a respetiva qualidade dos dados, visto que nem sempre e possível encontrar um conjunto que se adeque perfeitamente ao nosso contexto. Assim, de modo a solucionar esta lacuna iremos também apresentar um dataset com cerca de 40,000 imagens, completamente anotado frame a frame e que acreditamos ser uma mais valia na resolução deste problema. Para além do referido, e de modo não só a dar uma resposta mais significativa e dirigida aos nossos dados em particular, mas também para que seja possível ter uma visão preliminar daquilo que poderá ser o funcionamento de uma das tarefas do sistema, iremos apresentar duas experiências com os nossos dados, na área da deteção. A primeira irá envolver o finetuning dos nossos dados, enquanto que a segunda levará a um treino iniciado de raiz, apresentando depois os seus resultados como uma prova da escolha acertada do método de estado da arte.

Palavras-chave

Análise de Imagens de Vídeo, Aprendizagem Automática, Biometria, Deteção de Faces, Deteção de Pessoas, Inteligência Artificial, Reconhecimento Facial, Tracking, Visão Computacional.

Resumo alargado

Ao longo dos anos, o abandono escolar, o universitário em particular, tem sido sempre um tema em grande destaque. Com o avanço de várias áreas tecnológicas, assim como da Inteligência Artificial e da Aprendizagem Automática temos necessariamente de pensar em maneiras de ajudar a mitigar este problema. Se há fatores que não podemos controlar, como os económicos, há outros onde a nossa ação pode ser dirigida. Um dos fatores que permite avaliar o risco de abandono escolar é a assiduidade dos estudantes. Embora, naturalmente, estes dados possam ser analisados manualmente para fazer estas deteções, seria mais eficiente ter um sistema que fosse capaz de registar esta assiduidade, uma vez que a capacidade humana para os analisar é finita e muitas vezes apenas consegue inferir esta situação demasiado tarde. Naturalmente que um sistema que registe apenas assiduidade não irá ser capaz de dar uma resposta definitiva, mas será um primeiro passe importante. Deste modo, um sistema que seja capaz de conciliar a deteção de uma pessoa e da sua face enquanto é capaz de monitorizar de forma constante o sítio onde a pessoa está, para ser sempre capaz de a identificar mesmo que mude de sítio em conjunto com o seu reconhecimento facial, parecem ser fatores determinantes para levar a bom porto um sistema deste calibre. Este tipo de sistemas está, geralmente, muito relacionado com a qualidade dos dados e das suas anotações, pelo que é extraordinariamente importante ser capaz de recolher ou obter dados de qualidade que auxiliem na resolução dos diversos problemas apresentados.

Esta dissertação tem como fim apresentar soluções de visão computacional em resposta ao desafio biométrico de detetar pessoas e fazer o seu registo de assiduidade num ambiente de sala de aula, em particular de forma não-cooperativa, isto é, sem ser exigida qualquer participação ativa no processo, a não ser a presença na sala de aula. Em concreto, pretende-se iniciar este projeto procedendo à instalação de um dispositivo de vídeo-vigilância dentro de uma sala de aula da UBI, que será responsável pela captura e transmissão dos dados para um dispositivo central de processamento. Após esta primeira fase do projeto, pretende-se desenvolver o estudo dos diversos métodos estado-da-arte nas tarefas mais relevantes para este projeto, nomeadamente através do teste e validação de algoritmos para deteção de faces, *tracking* de objetos e reconhecimento facial. É também objetivo desta dissertação a apresentação de um novo conjunto de dados público especificamente ligado a ambientes de sala de aula, totalmente anotado frame a frame, e estando também totalmente preparado para poder ser utilizado em tarefas de *tracking*, visto que as suas anotações foram pensadas para tal. Para além das anotações essenciais para tarefas de deteção, este terá também em cada frame a indicação da ação que está a ser realizada, de modo a que possa servir de *input* a trabalhos futuros. Por fim, serão também validados duas versões do mesmo método nos nossos dados, aumentando assim a robustez dos resultados e conclusões apresentados. Para que todos estes objetivos possam ser cumpridos, este documento encontra-se organizado em cinco capítulos que contém diversas contribuições e que passaremos a explicitar.

O primeiro capítulo enquadra este projeto no seu contexto e apresenta o seu âmbito, permitindo localizar esta tese de uma forma geral, definindo qual o problema que se pretende resolver, os objetivos traçados para a dissertação,bem como as principais contribuições de investigação realizadas, apresentando por fim a estrutura do documento.

O segundo capítulo apresenta o trabalho relacionado com as tarefas desta dissertação e que servirão para o desenvolvimento futuro de um sistema não cooperativo para registo de assiduidade em ambiente de sala de aula, dando uma revisão detalhada da literatura e também a apresentação de uma simples definição de cada uma das tarefas. De modo a que seja mais simples proceder à leitura do estado-da-arte, este capítulo está estruturado por secções onde serão discutidos de forma consecutiva os diversos métodos de deteção de objetos, começando pelos métodos mais tradicionais e terminando nos de aprendizagem profunda, concluíndo que são estes últimos que servirão para o nosso propósito. Seguidamente, é feito o resumo no que diz respeito ao estado-da-arte do seguimento de objetos, efetuando-se várias distinções em relação a diversas formas de estudar esta tarefa. Apesar desse número elevado de distinções, optamos por dividir o mesmo entre o single-tracking e o multi-tracking, por considerar ser aquele que mais sentido faz no contexto do nosso problema. Por fim, são apresentadas as diversas etapas que estão envolvidas no reconhecimento facial, terminando com a análise de alguns dos métodos mais relevantes.

O terceiro capítulo procura desenvolver a recolha de dados, isto é, a construção do dataset que foi efetuada no contexto desta dissertação, assim como o processo utilizado para a captura e transmissão dos dados recolhidos. Além do referido, são também apresentadas descrições detalhadas dos datasets públicos relacionados com as diferentes tarefas que estamos a tratar, assim como também anunciamos e apresentamos uma descrição pormenorizada do nosso próprio dataset, PAPSE-UBI, que servirá os propósitos desta dissertação, encontrando-se totalmente anotado frame a frame. A análise dos datasets já existentes permitiu-nos identificar algumas características relevantes e também alguns fatores limitadores. Pelo facto de estarmos a lidar com pessoas e com a recolha de dados sensíveis, consideramos que seria também interessante discutir algumas questões relacionadas com a privacidade e a segurança/integridade dos dados, sendo discutida numa das seccções deste capitulo as várias adaptações que fizemos com vista a garantir que o projeto conseguisse sempre respeitar a privacidade dos envolvidos.

A construção do dataset é sem dúvida o ponto mais relevante deste capítulo, tendo três seccções distintas onde é discutido: na primeira é explicitado o processo de captura dos mesmos, é clarificado o conceito de uma sessão de captura e aquilo que se procurou fazer mediante as limitações existentes. São ainda apresentadas diversas estatísticas do dataset, nomeadamente ao nível do número de imagens que o compõe, o número e distribuição dos voluntários por sessão, o seu género e o uso ou não de máscara. Após efetuada esta análise, e tendo em conta os fatores limitadores que observamos em outros datasets, começamos

também por explorar a variabilidade dos nossos dados, nomeadamente analisando aqueles que nos parecem ser os casos que mais podem dificultar não só a aprendizagem do modelo, mas também os eventos onde uma futura deteção poderá falhar com maior probabilidade. Analisados os dados, foi também discutido o processo de anotação das imagens, um processo custoso do ponto de vista temporal, mas essencial para se poder prosseguir com o projeto. Foi apresentado o software utilizado, assim como as justificações para as principais decisões tomadas, nomeadamente quanto ao número de classes a discriminar e os seus respetivos atributos. Esta seccção foi finalizada através de uma explanação dos casos mais dificeis de anotar.

O quarto capítulo apresenta a comparação efetuada entre os diversos métodos de estado-da-arte, das três tarefas envolvidas, tendo sido submetidos a comparações justas e com métricas bem definidas para que os resultados que daqui se possam extrair tenham validade. No que diz respeito à tarefa de deteção foi efetuada uma comparação entre diversos métodos de deteção e diversas configurações, sendo a regra geral o uso dos datasets COCO e PASCAl para o finetuning quando necessário e o uso de average precision com diversos limiares. No que diz respeito ao tracking, esta tarefa foi dividido entre o single e o multi-tracking, sendo cada um avaliado de acordo com as métricas que consideramos serem as mais relevantes para uma comparação justa. Por fim, no reconhecimento facial, foi testado um método que comparava diretamente duas abordagens:o uso de um limiar fixo contra uma limiar adaptativo. Neste capítulo foi ainda discutida a comparação, usando o nosso próprio dataset, do melhor método resultante da análise de deteção anterior aplicando finetuning dos seus pesos aos nossos dados contra um treino de raiz dos mesmos, terminando com a comparação visual de algumas imagens.

Por fim, no quinto capítulo são discutidas as conclusões deste trabalho desde o seu início, apresentando conclusões sobre o estado da arte, a contribuição do dataset, as comparações dos vários resultados obtidos ao longo do projeto e ainda a resolução específica de uma das questões inicais, relacionada com a deteção de pessoas. Finalmente, é ainda discutido o trabalho futuro, apresentando algumas das possíveis ideias a desenvolver neste âmbito.

Abstract

Over the years, high school dropout, college dropout, in particular, has always been a hot topic. With the advancement of technology and Artificial Intelligence and Machine Learning, we necessarily have to think of ways to help mitigate this problem. If there are factors that we cannot control, such as the economic ones, there are others where our actions can be directed. One factor that allows us to evaluate the risk of dropping out of school is student attendance. Although this data can be manually analyzed to make these detections, it would be more efficient to have a capable system of recording this attendance, since the human capacity to analyze data is finite, and often can only infer this situation too late. Of course, a system that only registers attendance will not give a definitive answer, but it will be an essential first step. Thus, a system that can reconcile the detection of a subject and his face while being able to constantly monitor where the subject is, always to be able to identify him even if he moves from one place to another, together with facial recognition, seem to be determining factors to bring a system of this calibre to a successful conclusion. This type of system is generally very much related to the quality of the data and its annotations, so it is vital to collect or obtain quality data to help solve the various problems presented.

Considering what has been described, the main goal of this dissertation is to try to start solving the problem of school dropout, namely through the study, validation and testing of several state-of-the-art methods in the area of object detection, namely people and faces, but also tracking. The same work will have to be done on face recognition methods, being able to indicate the best state-of-the-art methods for each task. As mentioned in the previous paragraph, a significant limitation to this type of task is the data quality since it is not always possible to find a set that perfectly fits our context. Thus, to solve this gap, we will also present a dataset with about 40,000 images, thoroughly annotated frame by frame and that we believe to be an asset in solving this problem. In addition to the above, and in order not only to give a more meaningful and targeted response to our detailed data but also to provide a preliminary view of how one of the system's tasks might work, we will present two experiments with our data in the area of detection. The first will involve finetuning our data, while the second will involve training it from scratch and then presenting its results as proof of the correct choice of the state-of-the-art method.

Keywords

Artificial Intelligence, Biometrics, Computer Vision, Face Detection, Facial Recognition, Machine Learning, Subject Detection, Tracking, Video Image Analysis.

Contents

1 Introduction **1**

1.1 Scope and Motivation . . . 2
1.2 Problem Statement and Objectives . . . 2
1.3 Document Organization . . . 3

2 Related Work **5**

2.1 Object Detection . . . 5
2.1.1 Traditional Methods . . . 6
2.1.1.1 Viola–Jones . . . 6
2.1.1.2 Scale-Invariation Feature Transformation . . . 8
2.1.1.3 Histogram of Oriented Gradients . . . 10
2.1.2 Deep Learning-Based Methods . . . 11
2.1.2.1 R-FCN . . . 11
2.1.2.2 R-CNN . . . 12
2.1.2.3 Fast-RCNN . . . 13
2.1.2.4 Faster-RCNN . . . 14
2.1.2.5 Mask R-CNN . . . 15
2.1.2.6 You Only Look Once . . . 17
2.1.2.7 Single Shot Detector . . . 18
2.2 Object Tracking . . . 20
2.2.1 Single Object Tracking . . . 21
2.2.1.1 SiamRPN++ . . . 22
2.2.2 Multi Object Tracking . . . 24
2.2.2.1 DeepSort . . . 24
2.3 Facial Recognition . . . 26
2.3.1 Yolo-Face . . . 26
2.3.2 Dual Shot Face Detector . . . 28

3 Data Collection, Hardware and Configurations **31**

3.1 Related Datasets . . . 31
3.1.1 COCO . . . 32
3.1.2 PASCAL VOC . . . 33
3.1.3 VOT . . . 33
3.1.4 OTB-2015 . . . 35
3.1.5 MOT . . . 36
3.1.6 LFW . . . 37
3.1.7 Wider Face Dataset . . . 37
3.1.8 Face Detection Data Set and Benchmark (FDDB) . . . 38
3.2 PAPSE-UBI: Prevent Abandonment and Promote Success in Education at the Universidade da Beira Interior . . . 38

3.2.1 Data Transmission and Camera Specifications 39
3.2.2 Privacy Issues 42
3.2.3 Data Collection 43
3.2.4 Data Variation Factors 46
3.2.5 Data Annotation 49

4 Experiments and Results **53**
4.1 Subject Detection 53
4.2 Face Detection 56
4.3 Subject Tracking 58
4.4 Facial Recognition 60
4.5 PAPSE Dataset Results 63

5 Conclusions and Future Work **67**
5.1 Conclusions 67
5.2 Future Work 68

Bibliography **71**

List of Figures

2.1 Original image to integral image. Adapted from [Soc20] 7
2.2 Abstract representation of a pyramidal image representation. Adapted from [Min19] 9
2.3 Gradient representation: Magnitude and Direction. Adapted from [Sat16]. 10

3.1 Class *Person* image examples from the COCO dataset. 32
3.2 Class *Person* image examples from the PASCAL VOC dataset. 33
3.3 Sequence *gymnastics1* from the VOT2019 dataset 35
3.4 Sequence *Woman* from the OTB-2015 benchmark. 36
3.5 Examples taken from the MOT dataset training set. 36
3.6 Sample examples from the LFW dataset. 37
3.7 Sample examples from the WFD dataset. 38
3.8 Sample examples from the FDDB dataset. 38
3.9 Captured Frame with the chosen setup. 40
3.10 Installed setup. 41
3.11 Presence matrix of 45 voluntary students over 6 sessions. 43
3.12 Frames per session. 44
3.13 Frames per session. 45
3.14 Mask usage per session. 46
3.15 Example of a classical frame from session 1. 47
3.16 Example of drawback: Walking. 48
3.17 Example of drawback: Partial Occlusion. 48
3.18 Examples from a big set of students and also a secnario where everyone is leaving the room. 49
3.19 Overview from the CVAT software. 50
3.20 Annotations examples from PAPSE-UBI dataset. 52

4.1 Accuracy variation in YOLOv3 using four different configurations with four different image sizes. 56
4.2 Comparison between accuracy and the number of comparisons with the database on LFW and COLOR FERET datasets. 60
4.3 Inference from the finetuned model, on the left, and the one that was trained from scratch, on the right. 64
4.4 False detection of a subject. 65
4.5 Inference mistake from the sratch model, on the left, and the one that was finetuned, on the right. 65

List of Tables

3.1 COCO Dataset image distribution by split. 32
3.2 PASCAL VOC Dataset image distribution by split. 33
3.3 VOT 2019 sequences specifications. 34
3.4 Overview of different sequences size on each Dataset. 36
3.5 Available Cameras at SociaLab. 39
3.6 Available Lens at SociaLab. 39
3.7 Selected Cameras. 40
3.8 Selected Lens. 40
3.9 Technical Specifications of the camera. 41
3.10 Body attributes in PAPSE-UBI dataset. 51

4.1 Performance summary of current object detection methods trained on COCO dataset executed over class **person** of COCO-testdev [LMB+15] and finetunned on PASCAL VOC 2012 [EEVG+15] and executed over class **person** of the validation set. 54
4.2 Accuracy on FDDB dataset. 57
4.3 Accuracy on Wider Face dataset. 57
4.4 Performance summary of current multi-object tracking methods trained on MOT17 [MLTR+16] and finetunned over the MOT Challenge datasets 2DMOT15, MOT16 [MLTR+16] and MOT20 [DRM+20] 58
4.5 Performance summary of current single-object tracking methods on VOT16, VOT18 and VOT 19. 59
4.6 Relation between the number of comparisons with the database and accuracy, using a fixed threshold on LFW. 61
4.7 Relation between the number of comparisons with the database and accuracy, using a fixed threshold on COLOR FERET 62
4.8 Results from YOLOv5X finetuned on PAPSE-UBI. 63
4.9 Results from YOLOv5X from scratch. 63

Acronyms List

ACM	Association for Computing Machinery
AI	Artificial Intelligence
CCS	Computing Classification System
CNN	Convolutional Neural Network
DW-XCorr	Depth-wise Cross-Correlation
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
JDE	Joint Detection and Embedding
UBI	Universidade da Beira Interior
ML	Machine Learning
R-FCN	Region-based Fully Convolutional Network
RoI	Region of Interest
ROLO	Recurrent YOLO
RPN	Region Proposal Network
SIFT	Scale-invariation Feature Transformation
SSD	Single Shot Detector
SVM	Support Vector Machine
UP-XCorr	Up-Channel Cross-Correlation
XCorr	Cross-Correlation
YOLO	You Only Look Once

Chapter 1

Introduction

During the last decades, Artificial Intelligence (AI) has gained extraordinary relevance in several research areas such as medicine, computer science, voice recognition, biometrics analysis, or big data processing, among others. The increased interest presented in these different areas results in developing practical applications in these areas, contributing to intelligent systems' explosion. Those systems usually work by extracting patterns from raw data, producing their knowledge, and helping solve particular tasks in an AI sub-domain called Machine Learning (ML). In ML, one of the significant concerns is the correct extraction of high-level and abstract features, i.e., data characteristics that allow us to perform useful generalizations. Until recent times, this process was made by hard-coded methods, using the correlation between features in order to decide which ones were the best which has a high temporal cost. However, with the advances in computation in general and in neural networks, the feature extraction converges into a new field called Deep Learning. Using new methods based on Deep Learning, we can solve the major issue explained before, and the algorithms can automatically learn abstract features even with many variation factors. For example, if we consider identifying a subject, Deep Learning-based methods can learn the key features even if we are dealing with different ethnicities, poses, or brightness in constructing complex systems by using simple data. Methods based on Deep Learning have surpassed state-of-the-art methods in many different tasks. One of the main reasons for this success is their architecture, constituted by different processing layers with many adjustable parameters leading this neural network field to become standard in tasks such as audio, image, or video processing.

As expected, video image analysis has become an essential field of research supported by Deep Learning advances, leading to developing new and innovative systems to solve biometric problems. One of the several applications of this is developing authentication systems based on facial recognition and monitoring human beings' behavior in a controlled environment, like classrooms, in a restrictive period, which still is a work-in-progress problem.

An authentication system based on facial recognition is a system that is capable of matching a face detected in a digital image or video with a previously known database of faces. To achieve a system that can continuously monitor human behavior and keep track of the identity, we will need many different research areas, such as object detection of faces, in particular, that consists of identifying a human face in a digital image achieve human behavior monitorization. We also need tracking of human faces to guarantee that we can still identify and relate both faces if a subject moves. Finally, face recognition to give the

correspondent ID to the previously detected face.

In particular, this document aims to describe the proposed solutions to solve this problem, starting with an extensive overview of the state-of-the-art methods and expose the needed tasks and their respective schedule to succeed in this task.

1.1 Scope and Motivation

According to the last report [Gab19] of the Universidade da Beira Interior (UBI), taking into account the 2019/2020 school year, around 1143 students were at high risk of university dropout in that period of a total universe of more than 7500 students. In the same report, many reasons were listed as the main responsible for that dropout, such as poor economic conditions or personal reasons. Although UBI solved some cases, it is crucial to identify these students as soon as possible. Many strategies may be used to identify these cases early, and according to the European Union (EU)[Ann11], one of the first indications that a student may be at high risk of university dropout is low attendance rates. Even though humans may process attendance rate data, an automatic procedure will boost those detections and avoid students dropping out in the future.

Many research areas are needed, and many stages must be completed to achieve a non-cooperative procedure that can automatically detect university dropout. Having that in mind, this work's scope falls into the intersection of AI and ML. In particular, since we will deal with video/image processing, Deep Learning will undoubtedly play a vital role. According to the 2012 version of the Association for Computing Machinery (ACM) Computing Classification System (CCS) [Ass12], the scope of this master's project is defined by the categories named:

- **Computing Methodologies**
  Artificial Intelligence - Computer Vision Problems: Object Detection
- **Computing Methodologies**
  Artificial Intelligence - Computer Vision Problems: Object Recognition
- **Computing Methodologies**
  Artificial Intelligence - Computer Vision Problems: Tracking

Since the previously mentioned areas are vast and may lead to many different approaches, urges to identify which problem are we addressing and which objectives we want to achieve, which will be done in the next section.

1.2 Problem Statement and Objectives

As previously clarified, we want to reduce university dropout rates, and to achieve that, a big project is being developed in UBI, and this work, in particular, is part of it. Although

some cases at high risk of university dropout are manually detected, a lot of them are missed, or even if we can detect them, we cannot reach them early enough to solve their issues. The problem addressed in this work is finding a way of automatically detecting them earlier, which will reduce the need for human resources in this task and optimize the overall attendance register process.

The earlier detection of these situations cannot be solved just with one module, but this work's primary goal is finding an innovative and efficient way to register attendance in a controlled environment such as a classroom because it is one of the earlier signs of university dropout like presented in section 1.1. The plan consists of installing a camera in one of the UBI classrooms, record videos of those classes in that particular classroom, and then sending the data to a central system that will process the data. This system must be scalable and flexible enough to be used in different institutes and circumstances.

To achieve the primary goal, we will need to fulfill other secondary goals. The building of such a system implies the study, test, and validation of the state-of-the-art algorithms, namely in Object Detection, Object Recognition, and Tracking. Besides these secondary goals, we also want to create a dataset composed of the images we will record to sustain our results that will be presented in a master's dissertation.

At the end of this work, we hope that our primary goal is fulfilled, which would mean that we contribute to the reduction in university dropout. To achieve that, we want to:

- validate state-of-the-art methods in object detection,
- object tracking, and
- object recognition in different datasets, which will result in choosing adequate ones for our problem;
- Build a fully annotated dataset;
- Validate detection methods in our data.

1.3 Document Organization

The rest of this thesis's project is organized as follows: Chapter 2 presents a detailed overview of the state-of-the-art object detection methods, analyzing the evolution of them by using some demonstrative examples using different approaches, doing the same with Object Tracking and Facial Recognition.

Chapter 3 will present a review of the various datasets related to our problem, give a brief description of these, and present some images for demonstration. We will also present our dataset, PAPSE-UBI, analyzing the whole process that led to its finalization, from collecting images to its transmission and ending in the characteristics of the collected data.

Issues related to data privacy will also be addressed, as well as the dataset annotation.

Chapter 4 will present the results of our experiments on the various problems we set out to address, namely in subject detection, subject tracking and facial recognition, and also the evaluation of subject and face detection in our dataset.

Finally, chapter 5 will present the conclusions of this work as well as ideas for future work.

Chapter 2

Related Work

The current chapter will present an extensive overview of the state-of-the-art methods and approaches by reviewing the fundamental techniques and concepts that allow those methods to work and the limitations associated with them. Section 2.1 introduces the concept of object detection and clarifies the two distinct types of methods that may be used, making the transition to further detailed explanations and exemplifications in the subsequent subsections namely distinguish between Machine Learning-base methods and Deep Learning-base methods. In section 2.2 we will discuss the methods found in the literature about both single and multi tracking.

2.1 Object Detection

Object Detection is one of the common problems addressed in computer vision. It is used worldwide and consists of detecting and identifying instances of a particular object referred to a specific class such as dogs, cats, or persons in a digital image. This field of research is split into different subdomains, namely, face detection. Object Detection may be used in different real applications such as video surveillance, person or vehicle detection, or even people counting. Despite its utility on its own, object detection may be crucial for other tasks such as object recognition or tracking since it indicates the exact place of an object in a scene, increasing the efficiency of those processes.

The central concept behind object detection is that every class has unique characteristics that may be recognized and learned by a method and that exclude an object from belonging to another class than the chosen one. These characteristics must be recognized even in different environments with unfavorable conditions such as low illumination, features scaling or position, or obstacles, that is, an object from the same or a different class occluding the one we want to detect. Generally speaking, we can divide object detection methods into two distinct groups. The first one is Machine Learning-based methods where we first define a set of features that we believe that better represent the classes that we want to identify by using scientific methods to correlate them and then give the chosen characteristics as input to a technique like Support Vector Machine (SVM) to perform classification. On the other hand, most modern methods rely on Deep Learning-based methods that can automatically detect the most critical features of an image and perform classification simultaneously, becoming widely used and are currently state-of-the-art methods.

In the next subsections, we will overview both types of methods, although the second ones will be more extensively reviewed because of their practical relevance nowadays.

2.1.1 Traditional Methods

Traditional methods rely on a two-stage process: first, a set of features is defined, and then a classifier is trained. Among many different techniques or methods that can be used to define the set of features, we will explore three different ones: Viola-Jones, Scale-invariation Feature Transformation (SIFT), and Histogram of Oriented Gradients (HOG). These methods were chosen by their relevance in this field, but they are only a small representation of the complete set. In the next subsections, we will make a small overview of these traditional methods to create a historical review, even though they were unlikely to be used in our project.

2.1.1.1 Viola–Jones

Viola-Jones [VJ01] is a framework that can detect different classes of objects despite being more used to detect faces. It was initially explained in 2001 by Paul Viola and Michael Jones, and it is nowadays still considered an efficient method to do so. Generally speaking, this framework combines four different concepts to detect objects, namely Haar-like Features, Integral Images, the AdaBoost algorithm, and in the end, a cascade classifier.

In order to best understand the algorithm, we will first shortly describe the concepts listed above. When the main goal is to extract features from an image, most methods extract them directly from the input image. In contrast to this, Viola-Jones frameworks extract their color intensities' values directly, producing a single value. Haar-like features see an image in grayscale, defining the image into the light and darker regions. The single value previously referred to is calculated based on the intensities' sum in the light region subtracted by the intensities' sum in a dark region. Using this method is useful to extract essential features, namely edges, straight and diagonal lines that can be used to identify an object.

The process of calculating these sums can have a high temporal cost, and to reduce that cost, the concept of integral image is added. This concept is the middle term that allows the transformation of the original image into the integral image. The reduction of time occurs by summing all the pixels values above and from the left, adding up its value in the end, as clarified in figure 2.1. This process increases the speed of calculating the Haar-like features since we only need to perform subtractions in a specific area instead of looping in all the cells.

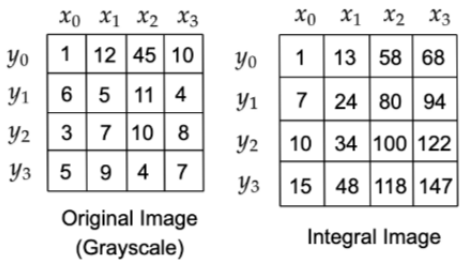

Figure 2.1: Original image to integral image. Adapted from [Soc20]

If we look closely to figure 2.1 we can understand that, for example, the value of (x_1, y_1) in the Integral image is the result of the sum of the cells above and to the left of the same cell in the original image, that is $(y_0, x_0) + (y_0, x_1) + (y_1, x_0) + (y_1, x_1) = 1 + 12 + 6 + 5 = 24$ and the same calculation is applied to every cell which allows the faster calculation of a certain region just by performing subtractions and a sum, i.e, imagine that we want to calculate the original value of the (y_1, x_3) integral image. We can do that calculation just by subtracting $(y_3, x_3) - (y_2, x_3) - (y_3, x_2) + (y_2, x_2) = 147 - 122 - 118 + 100 = 7$ avoiding the need of looping all the matrix. [Soc20]

Another essential role in the Viola-Jones framework is played by the AdaBoost algorithm that allows the framework to select the best feature among all the available. This algorithm is iterative and works by selecting the feature with the lower error rate in each iteration, having first calculated the error rate for all the features. The final output of this algorithm is a prediction function resulting from the weak classifiers' works in each iteration. The number of weak classifiers, that is, the number of iterations, is chosen by the person working with the framework. Finally, a Cascade Classifier is also used in this framework to implement a fast and accurate multi-stage classifier. Each stage classifier is the result of the AdaBoost algorithm, and this step works sequentially. When we advance into the next step, the number of a weak classifier on each strong classifier increases, and we only advance to the next stage if the particular classifier of the current stage thinks that the characteristics evaluated may be part of the particular class of object that we want to detect being automatically discarded if the opposite occurs.

The Viola-Jones framework work by constant cooperation of these four described modules and can be divided into two different stages: training and testing. The main idea behind training is to construct an accurate cascade classifier that can classify an object or discard it depending on the class. On the other hand, the testing stage aims to correctly detect all the given class objects in a given image. The training stage starts by pre-processing our data combining Haar-like features with the concept of integral image. We first need to extract features from the training dataset. According to their paper, it is highly recommended to use images with 24 x 24 size, which will result in approximately 160.000 Haar-like features, since each type of feature may be represented with different sizes and positions in that window. The integral image will be significant in this process because it avoids the calculation in all of these features. After the integral image conversion, we will

need to construct the cascade classifier that will consider the AdaBoost algorithm. Despite the increase in speed using the integral image, it is impossible to use 160.000 features directly, so the authors present two solutions to surpass this issue. The first one consists of reducing features by selecting which ones are useful and which ones are not. The selection process works by using the AdaBoost algorithm, which will split the features into two groups: one with the useful features and the features that must be discarded. From the remaining features, we will evaluate them by stages whether they will be discarded or not based on the true and false positive rate per stage. Finally, in the testing stage, we will evaluate the possible sub-windows based on the parameters that we define, and an object is detected if a sub-window passes through all the stages in the cascade classifier.
The biggest downside of this method is the incapacity of detecting an object in different orientations or arrangement.[VJ01]

2.1.1.2 Scale-Invariation Feature Transformation

Scale-Invariation Feature Transformation (SIFT) [Low99] is an object detection system that was firstly proposed in 1999 by David G. Lowe and consists in the use of features that are invariant to image scaling, translation, and rotation. According to the author, these features must also be partial tolerant to illumination changes and affine or 3D projection. The algorithm may be explained in four distinct modules: feature point detection, feature point localization, orientationn assignment and feature descriptor generation. The cost of extracting suce features is high however since we adopt a cascade approach the operations with higher cost are only applied once.

This algorithm defines itself as scale invariant which means that even if the scale changes the main properties of an object keeps the same, namely edges and corners, for example, avoiding the transformation of one of them into another which happens in same methods that are not scale invariant. It is relatively simple to understand this property but is important to be able to explain how this method achieves it. SIFT uses the Laplacian of Gaussian (LoG) to achieve the finding of common features even in different scales which allows the finding of edges and corners in an image. The edges and corners that are founded can then be used to finding keypoints. Despite being a good approximation, LoG is not totally scale invariant which means that we need to add an extra step that consists in the normalization of that expression, translated in practical terms to the multiplication of LoG with the second derivative of σ, since Laplacian is the second Gaussian derivative. The application of this normalization allows the method to correctly detect objects and its respective features even with different scales. Generally speaking, SIFT applies this method combined with pyramidal representations of the image that allows a good representation. An abstract representation of this can be seen in figure 2.2.

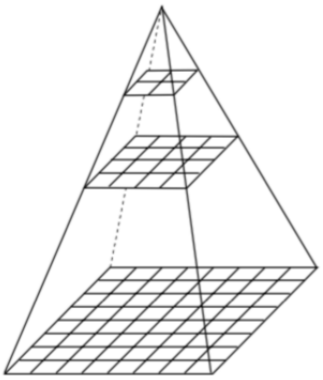

Figure 2.2: Abstract representation of a pyramidal image representation. Adapted from [Min19]

The potential interest points are detected by comparing different points in different levels of the pyramid, i.e, we select a specific point in a level and compares it with 8 neighbours in the same level and 9 neighbours in the levels above and below. This approach identify the potential interest points without considering scale or orientation and its name relies on the transformation of image data into scale-invariant coordinates.

After we detect the interest points, the next stage is to determine a detailed model that is able accurately determine its location and scale. The full details of this stage will not be extensively explained because they are beyond the scope of this work but in the end keypoints are selected based on their stability.

When we already possess the valid keypoints we reach the core of this algorithm and we will need to assign orientation to each keypoint. This assignment is based on local image gradient directions and can be explained as follows: we first start by implementing a square window around the feature and compute edge orientation for each pixel. After this, we define a minimal threshold for each gradient magnitude and excludes the ones that are below that limit. The remaining ones are inserted into an histogram weighted not only by its gradient magnitude but also by a Gaussian-weighted circular window with a σ that uses a scale 1.5 times higher than the one of the keypoint, giving invariance to these transformations.

At last, the SIFT algorithms generates a descriptor that outputs a representation that allows different levels of local shape distortion and low illumination conditions. The descriptor is composed by a normalized 128-dimensional vector and in this stage a list of feature points is given as input. The list is organized by location, scale, and orientation which gives the opportunity to build a coordinate system around the feature point. Like implicit said before, the descriptor is a histogram formed using the gradient taken by the grayscale image. A grid must be used and despite the size of it may be changed according to the point scale, the original paper [Low99] appoints to a 4 x 4 spatial grid of gradient angle histogram. Each spatial bin has an histogram divided into eight which justifies the initial size of the dimensional-vector. The main purpose of this window is to avoid abrupt

changes in the descriptor when small changes in the window occurs, given a small impact to the gradients that are far from the center.

2.1.1.3 Histogram of Oriented Gradients

HOG [DT05] is used for object detection and is based on feature descriptors that extract useful information and discard unnecessary features. In the particular case of HOG, besides feature descriptors, it can also convert the original image into a feature vector with a variable length. The main reason behind this conversion is to further use this image representation in SVM. The selecting process used by the feature descriptor relies on the histogram of gradients used to identify features of an image. As previously explained in the last subsection, these gradients are useful to identify edges and corners in the images using the changes of intensity in different regions.

The central concept of this method is the calculation of the gradients. That calculation is accomplished by the use of horizontal and vertical kernels that filters the image. The gradient's central point is that the magnitude of each gradient increases in the regions with higher contrast, i.e., where the intensity values change more. Gradients must have a magnitude and a direction, but it is essential to underline each calculation method. Magnitude is calculated by the maximum of the magnitude in three color channels, while the angle results from the maximum angle of the maximum gradient.

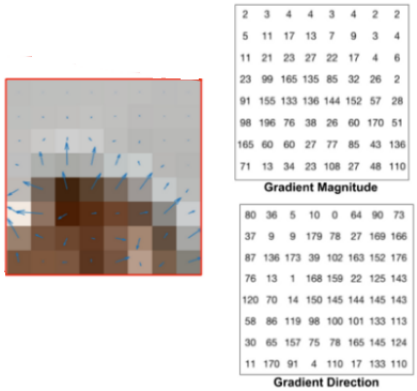

Figure 2.3: Gradient representation: Magnitude and Direction. Adapted from [Sat16].

Despite the calculation being essential, it is crucial to building a histogram from these gradients. As we can see in figure 2.3 the blue arrows represent the directions and grow proportionally to the gradient. To construct the histogram, we will need to compare the gradient direction with the gradient magnitude to feed the histogram of gradient bins. The most common way to define our histogram is as 9-bins, which means that we split the angles into groups of 20 each. For example, if we have a magnitude of four and our gradient direction is ten, we will equally divide the gradient's value into the [0] bin and the [20] bin with two gradients for each. After the process is completed by observing the histogram, we can easily understand which angles are more common, i.e., where is

more likely to find an object since it is more likely that the gradient converges into the non-background part of the image.

2.1.2 Deep Learning-Based Methods

The deep learning methods have the advantage of producing a set of features and then classifying them all in a joint pipeline, being this way able to overcome the traditional methods since even if the latter are fully optimised, the deep learning will be able to find it.

In the following sections, we will briefly systematise some of these methods.

2.1.2.1 R-FCN

We can distinguish between two subnetworks when we talk about the Region-of-interest (RoI) pooling layers: a shared fully-convolutional subnetwork independent of ROIs or an RoI-wise sub-network that does not share computation.

In a general way, we can say that the Region-based Fully Convolutional Network (R-FCN) [DLHS] uses a shared fully convolutional architecture and follows a two-stage object detection strategy when we first find a region proposal, and then we make region classification. We extract candidate regions by using the Region Proposal Network (RPN), a fully convolutional architecture, and after that, we share the features between RPN and R-FCN. When we have the proposal regions, they are classified by R-FCN into object categories and background. Using this approach, all learnable weight layers are convolutional, and we compute them on the whole image, producing score maps (k^2) and k^2(C+1) channel output layers with C object categories and an extra one for the background. The set of score maps corresponds to a spatial grid that encodes an object category's relative positions. The final step on R-FCN is a position-sensitive RoI pooling layer that receives the last convolutional layer's outputs and produces scores for each RoI. RoI layer performs selective pooling, and each of the k x k bin joins responses from a single map. In order to learn specialized position-sensitive score maps, the RoI layer uses end-to-end training.

To better understand this approach, there are some details on which we need to focus, and the first one is the **backbone architecture**. R-FCN is based on ResNet-101 that consists of 100 convolutional layers followed by global average pooling and then a 1000-class fully connected layer. Instead of using this architecture, R-FCN removes the fully connected layers and the average pooling, using only the convolutional layers to produce the feature maps. Another important aspect of this approach is the **position-sensitive score maps and the position-sensitive RoI pooling**. As explained before, to explicitly encode position information into each RoI, we use a regular grid to divide the regions of interest. After the last convolutional layer produces the score maps for each category, we define a position-sensitive RoI pooling operation between two positions of the score

map. The pooling operation performed in this step is average pooling, but max-pooling can also be performed. After this step, all the position-sensitive scores vote on the RoI, and a dimensional vector is produced for each RoI by averaging scores. To evaluate the cross-entropy loss during training, we compute the softmax responses across categories, which is also useful to rank the ROIs during the inference step. Another critical step is the bounding box regression that is addressed similarly, but we append another convolutional layer ($4k^2$) just for doing this, and the position-sensitive RoI pooling is performed on this set of $4k^2$ maps, outputting a vector of the same size for each RoI. The bounding box is defined (x,y,w,h) by the aggregation of the previous vector into a 4-d vector by average voting, and then the position-sensitive RoI pooling layer learns the score maps for object detection. After this moment, there is no learnable layer, and because of that, the region-wise computation cost is meager, and training and inference are faster than usual.

We can say that it is easy to train the R-FCN architecture because we use pre-computed region proposals and then perform end-to-end training. To measure our model, the loss function used is the summation of the cross-entropy and the box-regression loss, and we define positive examples as the ROIs that have an IoU overlap with a ground-truth box of at least 0.5. post-processed using a threshold of 0.3 and using non-maximum suppression.

The fully convolutional architecture used in this model benefits from the general modifications performed in FCNs for semantic segmentation. Specifically, the ResNet-101's effective stride is reduced from 32 pixels to 16 pixels, leading to increased score map resolution. Besides that, all layers before and on the conv4 stage remain unchanged. However, the stride=2 operation in the first conv5 block is modified to stride=1. The whole algorithm modifies all the convolutional filters on the same block to compensate for the stride reduction.

If we look closely yo the position-sensitive score maps, we realize that they should be activated at a specific relative position of an object due to specialization. If a boundary box overlaps an object, most of the RoI bins are activated, and they can now vote, leading to a high score. On the other hand, if a candidate box does not overlap an object, fewer bins are activated, which leads to a lower score.

2.1.2.2 R-CNN

This method [GDDM] tries to solve the problem of selecting a huge number of regions. The proposed method that will solve that is called Selective Search and can be summed up in 3 steps:

- Generate initial sub-segmentation, then generate many candidate regions.
- Use a greedy algorithm to combine similar regions into larger ones recursively.

- Use the generated regions to produce the final candidate region proposals.

When the algorithm is finished, we have 2000 candidate region proposals, warped into a square and fed into a Convolutional Neural Network (CNN) that will output a vector (feature extractor). After this, the algorithm will predict an object's presence and predicts four extra values called "offset values." The purpose of these extra values is to increase the bounding box's precision, adjusting it in the region proposal.

2.1.2.3 Fast-RCNN

Fast R-CNN [Gir15] was designed to solve the problems caused by R-CNN while improving speed and accuracy. In a general way, Fast R-CNN has higher detection quality and does not require any disk storage for feature catching. All network layers may be updated by training, and it performs in a single-stage.

When we consider Fast R-CNN network architecture, we realize that it takes an entire image as input and a set of object proposals. It uses several convolutional and max-pooling layers to process the image and, in return, produces a convolutional feature map. After this first stage is complete, the region of interest pooling layer extracts a vector with the same size for each object proposal and inputs it into a sequence of fully connected layers and then branches for one out of two output layers: one that produces *softmax* probability and another one that outputs four real-valued numbers for each object class. Those four values are the bounding-box positions for one of the classes.

The region of interest pooling layer uses max pooling to convert features in a valid region of interest into a feature map. This layer divides the RoI window into sub-windows and then max pool the values in each sub-windows into an output for all features.

In fast R-CNN training, stochastic gradient descent mini-batches are sampled hierarchically, choosing N images and selecting R/N ROIs from each image. By doing that, is from the same image will share computation and memory in the forward and backward passes. Using different ROIs of the same image, we may think that the training process would be slow because of the correlation between ROIs. However, according to this paper, the theoretical hypothesis does not seem to be a practical concern. The fast R-CNN training process has only one stage that optimally combines the softmax classifier and bounding box-regressors.

In order to better understand the training process, it is important to fully understand three concepts and some specifications used in this paper:

- **Multi-task loss**: Fast R-CNN network has two output layers where the first one outputs a discrete probability distribution per RoI and the second calculates the bounding-box regression offsets for each object class. Each training RoI is labeled with a ground-truth class and a ground-truth bounding-box regression target.

We define a multi-task loss on each labeled RoI to train for classification and bounding box regression at the same time. The first part of the loss is a log loss for real class, and the second one is defined over a tuple/sample of true bounding-box regression targets.

- **Mini-batch sampling**: During the fine-tuning stage, mini-batch is constructed from N=2 images randomly uniformed chosen. The mini-batches used have size R=128, sampling 64 ROIs from each image. 25% of the ROIs are taken from object proposals that have an IoU of at least 0.5. The other 75% are chosen from object proposals with a maximum IoU with the ground interval between 0.1 and 0.5.

- **Back-propagation through RoI pooling layers**: In a general way, we can say that for each mini-batch RoI and each pooling output unit, the partial derivative is accumulated if the argmax is selected by max pooling.

2.1.2.4 Faster-RCNN

Region-based convolutional neural networks and region proposal methods are more extensively responsible for significant advances in object detection. Initially, region-based CNNs had a huge computation cost, but their value has decreased a lot. This method usually includes Selective Search in the region proposal step despite better options such as Region Proposal Networks (RPNs) that combine convolutional layers with the classic algorithms. RPNs will use the convolutional feature maps used by region-based detectors and, on top of that, will add a few convolutional layers that will serve the purpose of regression in the region bounds and objectness at each location on the grid. RPNs are also designed to train end-to-end and to efficiently predict region proposals with different scales and aspect ratios using anchor boxes to achieve that.

Faster R-CNN [RHGS15] comprises two modules: a fully convolutional network that proposes regions and a detector that will use them. If we look closely at the first one, we understand that this module takes an image as input and returns a set of rectangular object proposals with an objectness score attached. To generate region proposals, a small network will slide over a convolutional layer, taking a spatial window of size N x N from the convolutional feature map. Each window will be mapped into a lower-dimensional feature and fed two fully connected layers: regression and classification. Each sliding-window location will predict multiple region proposals at the same time: the regression layer will predict 4k outputs encoding the coordinates of k boxes, and the classification layer outputs 2k scores that represent the probability of being an object or not an object, where k represents the number of maximum possible proposals for each location."The k proposals are parameterized relative to k reference boxes, which we call anchors." Anchors are centered in the sliding window and are related to aspect ratio and scale. By default, three scales and aspect ratios are used.

If we focus our attention on the properties of the anchor, an important one is Translation-Invariant. This property is valid to the anchors and also to the functions that compute proposals. This property means that if something translates an object in our image, then the proposals must also translate it, and the function should be able to predict the proposal.

Another innovation from this method is the way it addresses multiple aspect ratios and scales by using multi-scale anchors as regression references. The method consists of a pyramid of anchors that will classify and regress bounding boxes regarding the anchor boxes. The method will only use images and features maps of one size and then apply filters of a single size.

RPNs will assign a binary class label to each anchor. There are two types of anchors that will receive a positive label: the ones with the highest Intersection-over-Union with a ground truth box or the ones in which IoU overlap higher than 0.7 with any ground-truth box. In general, this second condition is enough to determine the positive samples. On the other hand, we assign a negative label to an anchor if its IoU is lower than 0.3. In other cases, anchors will not be labeled and will not contribute to the training.

This method adopts Fast R-CNN in the detection network, so we will need a way to share convolutional layers between Fast R-CNN and Faster R-CNN. This paper proposes three different approaches:

- **Alternating Training**: When using this approach, we first train RPN and use the proposals to train the Fast R-CNN, and then the network optimized by Fast R-CNN is used to initialize RPN with an iterative process.
- **Approximate Joint Training**: In this solution, both modules are merged into one single network during training, and in each SGD iteration, the forward pass generates region proposals are treated as pre-computed proposals when we need to train a Fast R-CNN detector. The backward propagation phase takes place as usual, but both RPN and Fast R-CNN losses are combined. Despite being faster in training than the previous approach, it ignores the derivative of the proposal boxes. Because approximations are used, some errors may occur.
- **Non-Approximate Joint Training**: At last, this approach tries to fix the previous problem using an RoI pooling layer that is differentiable to consider the gradients previously ignored.

2.1.2.5 Mask R-CNN

Mask R-CNN [HGDG18] is a framework to detect objects that efficiently detect objects in an image while generates a high-quality segmentation mask for each instance. Mask R-CNN is simple to train and extends Faster R-CNN. Instance segmentation is challenging

because we need to detect all objects in an image while segmenting each instance.

This method extends Faster R-CNN by adding a branch for prediction segmentation masks on each RoI executing at the same time that the branch for classification and bounding box regression. A mask branch is a fully connected network that predicts a segmentation mask in a pixel-to-pixel manner when applied to each RoI.

Mask R-CNN is designed to be simple, and like Faster R-CNN has two outputs for each candidate object: a class label and the bounding box-offsets, but they add a third output that is the object mask, but unlike the first two outputs, the object mask needs much more refined spatial layout extraction from the object.

In terms of the procedure, the first two stages of Mask R-CNN are very similar to those used in Faster R-CNN. The first stage, RPN, is similar, and in the second phase, we also output a binary mask for each RoI, which collides with the classical procedure where classification depends on mask predictions. This approach applies bounding-box classification and regression in parallel, simplifying the multi-stage pipeline presented in the original R-CNN.

In the training stage, we define a multi-task loss on each selected RoI, which includes the loss from classification, the loss from the bounding boxes, and the loss from the mask. The definition of mask loss allows the network to generate masks for every class without competition among classes, which means that we rely on the classification branch to predict the class label. By doing this, we separate the mask and the class prediction, which is not common. Instead of using a per-pixel softmax and multinomial cross-entropy loss, we apply a fully connected network to semantic segmentation.

A mask encodes an input's object's spatial layout, but unlike class labels or box offsets that are collapsed into short output vectors by FC layers, we can address the spatial structure extraction by the pixel-to-pixel correspondence provided by convolutions. Pixel-to-pixel behavior needs RoI features, and because of that, the creators of Mask develop a RoIAlign operation.

RoIAlign operation extracts a small feature map from each RoI. First, an operation of RoIPool quantizes a floating number RoI to the discrete granularity of the feature map, and then the quantized RoI is subdivided into spatial bins, which are themselves quantized, and finally, feature values covered by each bin are aggregated. Quantization introduces misalignments between the RoI and the extracted features. This difference may not affect classification but definitely will have a large negative effect on predicting pixel-accurate masks. A RoIAlign layer that moves the harsh quantization of RoIPool, aligning the extracted features with the input, was proposed to solve this issue.

This approach can be used in a large specter, and to demonstrate that, the authors instantiate Mask R-CNN with multiple architectures. They distinguish between the convolutional backbone architecture used for feature extraction and the network head for bounding-box recognition, and mask prediction applied separately to each RoI.

2.1.2.6 You Only Look Once

You Only Look Once (YOLO) [RDGF16] presents a new object detection approach, representing the problem as a single regression one, straight from image pixels to bounding box coordinates and class probabilities. YOLO's architecture is straightforward and allows a single convolutional network to predict bounding boxes and their class probability.

All the individual components of YOLO are united in a single neural network that then uses features from the entire image to make predictions of the bounding boxes. Another impact of this is that it enables end-to-end training and real-time speed while keeping a high average precision.

The system divides the image into a grid, and if the center of an object is inside a grid cell, then that cell is responsible for detecting that object. Each grid predicts bounding boxes and also calculates a score of confidence for those boxes. That score reflects the level of confidence that the model has in the box's judgment ability and if it contains an object or not. If the box does not contain an object, the confidence score should be zero; Else is the IoU between the predicted box and the ground truth.

If we look closely at each bounding box, we see that it consists of 5 different values: one that represents the center of the box's coordinates relative to the bound of the grid cell. We also have the height and the width, predicted relative to the whole image, and finally, the confidence prediction. At last, a grid cell also predicts a conditional class probability.

YOLO is implemented as a convolutional neural network where the initial convolutional layers extract features from an image, and the fully connected layers predict the output probability and coordinates. It has 24 convolutional layers and then two fully connected layers, inspired by the GoogleLeNet [SLJ+14] model. One of the differences between this model and the one used by GoogleLeNet is substituting the inception modules with reduction layers. Besides YOLO, a faster version was designed to push the boundaries of fast object detection, and it uses a neural network with fewer convolutional layers and filters.

When we consider the training step, YOLO's creators explained that the model is pretrained on the ImageNet, using the first 20 convolutional layers of YOLO's architecture to end the process pooling and a fully connected layer. After this process finishes, the model is converted to detection mode, adding convolutional and connected layers to improve performance. The final layer of the model is responsible for predicting class probability and also the bounding box coordinates.

In every image, many grid cells do not contain objects, and because of that, their score of confidence is zero, which leads to model instability and causes divergence. Another direct result of this instability is the overpower of the gradient of the grid cells that contain objects. A way to fix this is by making small modifications to the loss function, decreasing the loss from confidence scores, and increasing the loss from bounding box coordinates.

Another vital aspect of YOLO is that it predicts multiple bounding boxes per grid cell. However, we want one bounding box predictor to be responsible for each object, so we manage to do that by selecting the bounding box, which predicts the higher IoU with the ground truth. We guarantee that we choose the best bounding box for each case, which leads to specialization: each predictor will be the best in particular features (size or ratio, for example). The loss function only penalizes the classification error if an object is present and if that particular predictor has the higher IoU.

Advantages and Limitations

- YOLO is super fast in the test stage because it only requires a single network evaluation.
- Sometimes, large objects or objects near the boundaries are detected by multiple cells.
- YOLO has spatial constraints on bounding box prediction, limiting the number of objects that it can predict and explains why it struggles with smaller objects.
- The loss function treats errors that happens in small boxes and big boxes the same way. Error in big boxes are not problematic, but the ones in small boxes have a significant impact on IoU.

2.1.2.7 Single Shot Detector

Single Shot Detector (SSD) [LAE+16] tries to detect objects in images using a single deep neural network. At prediction time, SSD produces scores for the presence of an object in each default box and adjusts the box to obtain optimal fit to the object shape. SSD network combines predictions from multiple feature maps with different resolutions to handle objects of different sizes. SSD is easy to train and integrate into systems that require a detection component.

SSD does not resample pixels or features for bounding box hypotheses and is as accurate as other approaches. Briefly, we can say that SSD is a single-shot detector faster than the previous state-of-the-art and more accurate. SSD predicts category scores and box offsets for a default set of bounding boxes using small convolutional filters. To obtain higher accuracy, predictions are made on different scales from features map with different scales and separated by ratio. It is expected that we obtain high accuracy, and we should be able

to perform end-to-end training.

SSD uses a feed-forward convolutional network that outputs bounding boxes and scores for objects' existence in the boxes. After that, the final detections are produced by non-maximum suppression. The base network is based on a standard architecture used for high-quality image classification, and then a few auxiliary modules/structures are added to the network to produce detections. The added structures allow the model to behave with some critical features like **multi-scale feature maps for detection** that consists of adding convolutional feature layers to the end of the base network. The purpose of this is to allow predictions of detections at multiple scales, which is sustained by the progressive decrease of the layer's size. Another critical feature is **convolutional predictors for detection** which sustains that each added feature layer can produce a fixed set of detection predictors using a set of convolutional filters. The primary element for predicting parameters is a small kernel that produces either a score for a category or a shape offset relative to the default box coordinates. At each location where the kernel is applied, it produces an output value, and the bounding box offset output value is then measured relative to a default box position to each feature map location. Finally, another essential feature of the model is **default boxes and aspect ratios** that are linked to bounding boxes. Those boxes tile the feature map, and each box position relative to its corresponding cell is fixed. After that, in each feature map cell, the offsets are predicted relative to the default box shapes in the cell and the per-class scores that indicate the presence of a class instance in which of those boxes. We compute the class scores and the four offset values relative to the original default box shape for each box at a given location.

The significant difference between training SSD and a regular detector (that often uses region proposals) is that ground truth information needs to be assigned to specific outputs. Once the assignment is done, the loss function and backpropagation are applied in end-to-end training. We need to perform another task in the training stage to decide which default boxes correspond to a ground truth detection and then train the network accordingly. We establish a relation between each ground truth box and the default box with the best Jaccard overlap. Then we match the default boxes to any ground truth box with Jaccard overlap higher than 0.5.

A classical problem in object detection is **choosing scales and aspect ratios for default boxes** to handle objects with different scales. SSD solves the problem using feature maps from different layers in a single network while sharing parameters. The lower layers capture more delicate details of the input objects, and many studies suggest that adding global context pooled from a feature map will help flat the segmentation results. Since we combine predictions for all default boxes with different scales and aspect ratios from all locations of a large number of feature maps, we have a large and diverse set of predictions covering various input object sizes and shapes, which is essential to help the learning process.

After the matching step, many default boxes are negative, which leads to instability between the positives and negatives samples in the training process. Since we have more negatives than positives, the solution is to ignore some of them. We sort them by the highest confidence loss for each default box and only use the top ones. we can also do data augmentation to make the model more robust. To achieve that, we pick every image of the dataset and randomly decide if we will use the original image, randomly sample a patch, or select a patch with minimum Jaccard overlap.

2.2 Object Tracking

Object tracking consists of locating moving objects over time in a video. This process is fundamental because it solves some issues caused by the limitations of object detection. If we only use object detection, we don't have information about the past movement or a way to connect the objects in one frame to its previous.

It is safe to say that tracking works when object detection fails. We can use it to solve problems when the object is partially or entirely occluded or when two or more objects cross each other to distinguish them. Another application of tracking is in blurred objects when visually the object does not look the same, but we can identify it with tracking. If, for some reason, our view of an object changes, it could be challenging to identify it without the context, and the same applies to scale change. Another classical failure in object detection occurs when the background and the object have similar colors, making the distinguish process harder. Low resolution and illumination variation are also some constraints that tracking helps to minimize. [YJS06]

The large majority of object tracking techniques applies two key concepts:

1. **Motion model**: It is essential to understand an object's behavior and be able to predict the potential position it in future frames. This ability reduces the search space, increasing the speed of processing. Despite being an essential condition, it is not enough to construct a robust algorithm because it still fails when objects abruptly change direction or speed.

2. **Visual appearance model**: To make better predictions, trackers need to understand the appearance of an object and distinguish it from the background. Unlike the previous concept, in some cases, the visual appearance model alone is enough to track, specifically in single object trackers.

In a general way, tracking algorithms are faster than object detection because they do not try to learn all the object variations. Tracking algorithms can be split into detection based or detection free trackers.

- **Detection based tracking:** This type of detectors receives consecutive video frames,

and it tries to track the objects. It is a popular approach because new objects are detected, and disappearing objects are terminated automatically.

- **Detection free tracking:** This approach needs manual initialization of the objects that we want to track. We need to identify them in the first frame, and then the algorithm localizes them in the following frames. One of its limitations is the scenario where new objects appear in middle frames, and since we do not identify them, it can not track them.

Another division that is usually done in trackers is related to the number of present objects in our environment. We can distinguish them into two categories:

- **Single object tracker**: In this scenario, we only want to track one object in the environment despite others' possible existence. The object that we want to track is initialized in the first frame.
- **Multiple object tracker**: Here, we want to track all the possible objects in the environment over time.

Trackers can also be separated into online or offline.

- **Offline trackers**: They are used when we need to track objects in a recorded video for any purpose. We can use the past and the future frames to achieve higher accuracy.
- **Online trackers**: We use them when we need predictions immediately, and because of that, we can not use future frames to improve the results.

Another distinction that is possible to make is based on a learning strategy that can be offline or online.

- **Offline learning tracker**: The trackers are trained offline, and they do not learn anything in run time. All the concepts are understood offline, and its success depends on that process.
- **Online learning tracker**: On the other hand, online trackers learn about the object using the initialization frame where we drew a bounding box. Since we draw a bounding box in any object to be recognized, this approach is more general.

In the next subsection, we will analyze some methods considering the division between single and object tracking because we think that is one of the most critical divisions since it directly influences the method's capacity to analyze one or many objects.

2.2.1 Single Object Tracking

When we talk about single-object tracking, we can identify many methods that can perform tracking in a single object. **Recurrent YOLO (ROLO)**[YK19] is a single object tracking method that combines object detection and recurrent neural networks. It is

a powerful combination between YOLO and LSTM: the visual features are collected by YOLO and location inference priors, and for each frame, LSTM receives an input feature vector and gives the location of the tracked object. **SiamMask** [WZB+19] is also a single object tracker that produces rotated bounding boxes and class-agnostic object segmentation masks. To achieve the previous, we need to initialize the method with a single bounding box to track the object. Since we can only use one single bounding box, multiple object tracking cannot be possible with this method. **Joint Detection and Embedding (JDE)** [WZL+20] is a very recent proposal similar to RetinaNet and is a single shot detector that achieves object tracking by appearance embedding vectors that are outputted when processing the frames. These vectors are compared to the previously detected objects using an affinity matrix.

In the next subsection we will talk about a specific single object tracking method called SiamRPN++.

2.2.1.1 SiamRPN++

Siamese network-based trackers have received much attention in recent times, but despite their outstanding performance when we compare accuracy and speed, it is far off from the state of the art methods in accuracy. Reports claim that many trackers based on Siamese Networks use AlexNet as architecture, and there was no apparent advantage when a more complex architecture was used. The authors of this paper [LWW+18] tried to analyze the reasons that explained this and understood that the main reason is the destruction of the strict translation invariance. Since the target may appear in different positions in the search region, the learned feature representation must stay spatial invariant, and in modern architectures, the only one that satisfies this is AlexNet [KSH12]. The solution consists of training a SiamRPN based tracker with ResNet as a backbone network and using that. Since we are using ResNet, we can use some characteristics to our benefit like a layer-wise feature aggravation for the cross-correlation operation, which may help the tracker predict the similarity map from features learned at multiple levels.
The Siamese network-based tracking algorithm faces visual tracking as a cross-correlation problem and learns a tracking similarity map from deep models with a Siamese network structure where one branch is used for learning the feature presentation of the target and the other one for the search area. In this scenario, the target patch is usually given in the first frame of the sequence, and the goal is to find the most similar patch from the next frames. This behavior is possible due to two restrictions while designing a Siamese tracker. One is the contracting part, and the feature extractor used in the trackers have an intrinsic restriction for strict translation invariance, which allows an efficient training and inference. Besides this, it also has another intrinsic restriction for structure symmetry, convenient for similarity learning.
The core reason that justifies that deep networks were not used in the Siamese track is divided into two different aspects: the padding in the deep network will destroy the strict translation invariance, and the other is related to RPN their requirement to have asym-

metrical features for classification and regression. The first problem may be solved with a spatial aware sampling strategy. One network is immune to this problem: AlexNet, precisely because it does not have padding, and we cannot escape from this issue in recent networks such as ResNet or MobileNet like in the large majority of recent big networks.
The authors tested their hypothesis by performing experiments on a network with padding and defined shift as the max range of translation generated by a uniform distribution in data augmentation. The experiment consists of the placement of targets with three different shift ranges and in three disjoint experiments. After the experiments converge, the test dataset's heatmaps are aggregated, and then results are visualized. With zero shift, the first experiment showed that despite the probabilities on the border area are degraded to zero, a strong center bias is learned independently of the target's appearance. The other two experiments conclude that the model is less likely to fall into the mentioned problem when shift increases. The quantitative results also showed that an aggregated heatmap of 32-shift is closer to the location distribution of test objects, which proves that the spatial aware sampling strategy reduces the break of strict translation invariance property, reducing or even eliminating the center bias.
After using deep networks, it becomes possible to aggregate different deep layers. To perform visual tracking, we need to have rich representations that span levels from low to high and resolutions from fine to coarse. A layer in isolation is not enough to obtain good results even with the depth of features, and in general, the compounding and the aggregation of these representations improve the results of inference either in recognition or localization. It was common to use external networks, which results in multi-level features that cannot provide very different representations. In the particular case of this network, multi-branch features are extracted to allow them to cooperate and infer the target localization.
The combination weights are split into two different domains: one for classification and another for regression. The weight is end-to-end optimized offline together with the network. This method does not explicitly combine convolutional features but instead learn classifiers and regressions separately. There are several cross-correlation layers, and next, we list some of them:

- **Cross-Correlation (XCorr)**: This type of layer predicts a single channel similarity map between target template and search patches in SiamFC.
- **Up-Channel Cross-Correlation (UP-XCorr)**: THis layer outputs multi-channel correlation features by cascading a heavy convolutional layer with several independent XCorr layers SiamRPN.
- **Depth-wise Cross-Correlation (DW-XCorr)**: At last, this layer predicts multi-channel correlation features between a template and search patches.

To solve this issue, a lightweight DW-XCorr was designed to achieve efficient information association and contains ten times fewer parameters than the last one with similar performance. This reduction is obtained by adjusting a conv-bn block from each residual

block to suit the tracking task. Another difference from this method is that the bounding box prediction and anchor-based classification are both asymmetrical. The biggest gain by changing cross-correlation to depthwise correlation is the reduction in computational cost and memory usage, and also the numbers of parameters on the template and search branches are balanced, resulting in a more stable training procedure.

2.2.2 Multi Object Tracking

When we talk about multi-object tracking, we can identify many methods that can perform tracking im many objects at the same time. **DeepSort** [WBP17] uses a cosine metric learning method to learn a feature space where the cosine similarity is effectively optimized through the softmax regime's reparametrization. In this method, the bounding boxes are computed using a pre-trained convolutional neural network trained on a large-scale person re-identification dataset. It is a solid start point because it is simple to implement, offers reliable accuracy, and runs in real-time. **Track R-CNN** [SBMT20] In this method, the object detection uses Mask R-CNN on top of a ResNet-101 backbone. The tracker is created by integrating 3D convolutions that are applied to the backbone features. Another modification of this method is that it extends Mask R-CNN by an association head to associate detections over time. It is a fully connected layer that receives region proposals and outputs an association vector for each proposal. The training consists of a video sequence adaptation of batch hard triplet loss. The final result is the decision made by the system that will choose which detections will be reported. The matching between the frame detections and the current proposals is done using the Hungarian algorithm, allowing pairs of detections with association vectors that must be smaller than a given threshold. **Tracktor++** [BMLT19] predicts an object's position in the next frame by calculating the bounding box regression without training or optimizing the tracking data. In a general way, it uses Faster R-CNN with 101-layer Resnet and FPN as an object detector. This method's main idea is to use the regression branch of Faster R-CNN for frame-to-frame tracking by extracting features from the current frame and then use the previous frame, specifically its object locations, as input to the RoI pooling process in order to regress their locations into the current frame. Tracktor++ also uses motion models and short-term re-identification. This re-identification process tracks a fixed number of frames and then compares the newly detected track to them for possible re-identification.

In the next subsection we will describe with more detail other method called DeepSort.

2.2.2.1 DeepSort

DeepSort [WBP17] uses an object detection algorithm such as YOLOv3 and a system to track obstacle that also works with occlusion. The algorithm found to achieve that is by using a Kalman filter [LLJD15] and The Hungarian Algorithm [KY55]. Based on a score, the Hungarian Algorithm (Kuhn-Munkres) can associate an obstacle from one frame to another. There are many score metrics such as IoU that compare the bounding boxes

between the ground truth and the predicted one, and if the bounding box is overlapping the previous one, it is probably the same. Another metrics is *Shape Score* that increases the score if the shape or size did not vary too much during two consecutive frames. The *Convolution Cost* is another metric that runs a CNN on the bounding box and compares this result with the one from the previous frame, which means that if the convolutional features are similar same, then the object will probably be the same. As a significant result, if there is a partial occlusion, the convolutional features will stay partly the same, and association will remain, which solves one of the biggest problems while tracking objects.

The big question is how to do the association, and briefly, we can say that the process begins with two lists of boxes from YOLO: a tracking list and a detection list. In those lists, we will calculate IoU, shape, and convolutional score. It is also possible to have a cost function measuring each score. Taking IoU as an example, the bounding boxes will overlap in some cases. If something similar to this happens, we set the maximum IoU value to 1, and the rest will receive 0. When we do this, we can understand where detection and track matches. After this information, we will need to find the lowest tracking value in the matrix for each detection. We are now able to say what element in detection matches what element in tracking, and therefore we can output matched detections, unmatched detections, and unmatched trackings.

On the other hand, the Kalman filter is top-rated for tracking obstacles and predicting current and future positions. It is used on every bounding box after its match, and this filter uses to predict and update functions that are used to calculate state mean and covariance that are the core of this algorithm. To better understand the algorithm, we will define two concepts: Mean is the bounding box's coordinates, and covariance is our uncertainty on this bounding box having these coordinates. This method is composed of two steps: prediction, which will predict future positions and update, correct those positions, and improve the prediction part by changing uncertainty. The Kalman Filter works as a cycle where prediction and update influence each other.

At the beginning of the algorithm, we have a measurement of 3 bounding boxes, for example. The algorithm will define them as three new detections, and for each box, we will initialize the Kalman matrices with coordinates of the bounding boxes. After this, one iteration has three bounding boxes of the same object, the algorithm matches them with the three former boxes, and we can now apply the predict and update functions. The process predicts the actual bounding boxes at the current iteration from the previous ones and then updates our prediction with the current measurement.

The prediction phase consists of a matrix multiplication that outcomes our bounding box's position at the current time, based on the previous position. The goal is to construct correct matrices in order to have a useful calculation. The first matrix (F) defines the state transition, and it is the core of the implementation because what we put in there, we will

change x (position). The matrix contains a time value representing the difference between the current frame and the former frame timestamp. In this scenario, we consider the velocity as constant, and therefore we use the Constant Velocity model. The second matrix that we need to consider is the Process Covariance Matrix (Q), representing the noise matrix and shows us how much confidence we give in the system. This matrix will be added to our covariance and will define our global uncertainty.
The update phase is a correction step that includes the new measurement and helps improve our filter. It started with the measuring of an error between measurement and predicted mean. After this first step, we calculate the Kalman Gain used to estimate the importance of an error.
Another critical step is how to set up the matrices correctly. To make a correct set up of the matrices, we will need a measurement vector, a measurement matrix, and a measurement noise. The measurement vector(Z) is the measurement at a specific time, and we do not input velocities here. On the other hand, we have the measurement matrix (H) that makes the math work between all the different matrices. Finally, the measurement noise (R) measures the noise from the sensor, and we need to define a noise for the YOLO algorithm in terms of pixels.

2.3 Facial Recognition

Facial recognition is commonly defined as identifying the identity of a subject using their face by comparing patterns based on the person's facial details. This process can be split into three different steps:

1. Face Detection: It is a fundamental aspect and consists of detecting and localizing human faces in images and videos.

2. Face Capture: In this step, a face is transformed into digital information based on the subjet's facial features.

3. Face Match: At last, this step is responsible for verifying if two faces are from the same subject.

Face recognition can be used in two different tasks: identification or authentication. The first one allows us to identify who a subject is, while the second one is responsible for verifying if a subject is the subject that he claims. This type of recognition is widely used because it is easy to deploy and implement but also because the process is faster than other recognition methods such as fingerprint or irises, for example. The following subsections will be focused on the first step of the facial recognition process.

2.3.1 Yolo-Face

In the particular case of faces, it is known that YOLO does not have the best performance when it takes to small objects/areas, which allows us to infer (and it was experimentally

proven) that the performance in faces will not be good. This paper [CHP+20] suggests an approach based on YOLO but with small differences in order to detect faces better. The significant differences of this face detector in comparison with YOLOv3 are made by adjusting anchor boxes and the regression loss function. This paper's results showed that this new method's accuracy outperforms the original one, and because of that, it is the right solution for this particular task.

The first stage in face recognition is to detect and locate faces in images or videos, and the main goal is that an accurate detection algorithm can benefit from the performance of a good system, and the reverse is also valid. A big concern/problem in face detection is that detection accuracy in different scales changes a lot for the same image, restricting the identification. The proposed solution to solve this issue is based on YOLOv3 and adds a new loss function while adjusting the anchor boxes. The experiments performed on this model, specifically with the Wider Dataset, showed that the performance increases massively (between 18 to 21% in terms of accuracy), which makes this solution a robust one and better if compared with simple YOLO.

This novel method aims to have a similar detection speed compared with YOLO and have better accuracy, hoping that the previously mentioned changes take effect. This method uses darknet-53 as a network backbone, and it is composed of a feature extraction network and three other detection networks. If we focus our attention on the feature extraction network, we realize that it is based on darknet-53 constituted by a hybrid combination between darknet-19 and ResNet. It is essential to highlight the two consecutive convolutional layers (3 x 3 and 1 x 1, respectively). The use of this robust network allows the increment of speed while keeping a similar performance to ResNet-152. The authors suggest that the backbone structure impacts the overall performance since, as it is well known, YOLOv3 struggles with small objects, and with the introduction of this new backbone, it is expected to solve that problem.

One of the changes was on the anchor boxes hoping that the new ones are more appropriate for face detection. In anchor boxes, scales and ratios are the most important hyperparameters, and the relation between anchor boxes and the detected targets should exist and should also contain a large spectrum of different possibilities. The major contribution of this approach is in the shape of the anchor boxes. Since this method is specific to detect faces instead of general object detection, we do not need all the possibilities, and we may focus on general concepts such as "the height of a face is always larger than the width." By doing this, we are reducing the possibilities and increasing the accuracy of the system. This method purposes two kinds of anchor boxes:

- Original YOLOv3 anchor boxes with a twist: It is converted from flat boxes to slim boxes. In this context, flat means that the box's heights are less than widths while slim means narrow and taller ones.
- K-means clustering anchor boxes: In this type of anchors, we apply k-means cluster-

ing on the WIDER FACE training dataset to get the dimensions of bounding boxes. This process starts with an empirically set of the number of seeds and then randomly select anchor boxes as initial clustering centers, adjusting the anchor boxes by calculating the IoU of the firstly chosen anchor boxes and the other. The IoU will be used as the distance of anchor boxes, and after that, all the face labels are divided into classes. After this step, each class anchor box size's mean values are taken as the new cluster centers. This process is repeated until convergence is achieved.

In terms of the loss function, if we compare it with the original one from YOLO that is composed of four parts (regression loss, confidence loss, classification loss, and the loss responsible for not having any object) and in this case, the distribution of the weights is designed for multi-class object detection which is not the case in face detection. To adjust the function, the proportion of the weights changed from 1:1:1:1 to 2:1:0.5:0.5 but keeping the same parts of the original function: loss of coordinate regression, loss of confidence in anchors with objects, loss of confidence in anchors with no objects, and finally, the loss of classification. In a general way, the IoU of a predicted location and its respective ground truth are used to evaluate the optimizer, and the MSE function is used as regression loss. IoU has a problem with overlapping bounding boxes: it is infeasible to optimize. In order to solve this issue, a new metric was proposed: GIoU. This new metric has a strong correlation between optimizing the MSE function and the metric itself.

2.3.2 Dual Shot Face Detector

DSFD [LWW+19] proposed a new face detection network and they highlight three different contributions: a Feature Enhance Module (FEM) for enhancing the original feature maps to extend the single shot detector to dual shot detector. They also adopt a Progressive Anchor Loss (PAL) that is constituted by two different sets of anchors to facilitate the features. At last, the authors use an Improved Anchor Matching (IAM) by integrating novel anchor assign strategy into data augmentation. This authors proposed a new method that takes into account the context relationship between anchors by incorporating a feature enhance module that incorporates multi-level dilated convolutional layers to enhance the semantic of the features. On the other hand, the objective loss in detection is usually a weighted sum of classification loss and box regression loss. However, in this method, the anchors sizes are adaptively chosen depending of the stage in which we're in with the purpose of facilitating the features. At last, to make the model robust, a lot of detection methods do data augmentation. Dual shot face detector on the other hand combines anchor assign to provide better data initialization for anchor matching and don't ignore random sampling in data augmentation.
If we look closely to the Dual Shot Face Detector we understand that DSFD is more flexible and robust than the average standard methods. It is important to highlight the following innovations that are the most important changes:

- Feature Enhance Module: This module is able to enhance original features to make them more discriminable and robust. This modules uses different information in-

cluding upper layer original neuron cell and current layer non-local neuron cells. This method first uses up-sample upper feature maps to do element-wise product with the current ones. At last, features maps are splitted to three parts and after that three sub-networks containing different numbers of dilation convolutional layers.

- Progressive Anchor Loss: In opposite to the traditional detection loss, progressive anchor sizes are designed not only for different levels but also different shots in this particular framework. According to the state of the art, low-level features are more suitable for small faces, this methods assign smaller anchor sizes in the first shot and use larger ones in the second one-. The loss is calculated based on the number of positive and negative anchors and the number of positive anchors, respectively. In addition to this concepts, Softmax loss between face and background is used and also the loss between the parameterizations of the predicted box and ground-truth box. In comparison with enhanced feature maps in the same level, original feature maps have less semantic information for classification but more high resolution information for detection. According to the authors, original feature maps can detect and classify smaller faces. In order to predict, only the second shot is used because no additional computational cost is needed.

- Improved Anchor Matching: The anchor matching methods that exists nowadays is bidirectional between the anchor and ground-truth face. In this case, and in order to improve the recall rate of faces and ensure anchor classification ability simultaneously and IoU (t=0.4) is used to assign anchor to its ground-truth faces.

Chapter 3

Data Collection, Hardware and Configurations

This chapter will present an overview of some related datasets that helped us in our experiments in the different explored areas such as Object Detection, namely Common Objects in Context (COCO) and Pattern Analysis, Statistical Modelling and Computational Learning Visual Object CLasses (PASCAL VOC); On Tracking such as Visual Object Tracking (VOT) or Object Tracking Benchmark 2015 (OTB2015) and Multi Object Tracking (MOT); Finally on Face Recognition using Labeled Faces in the Wild (LFW). We will briefly discuss the data collection conditions and also add some pictures to illustrate each dataset.When building a new dataset, it is essential to explore the existent ones to identify the positive pattern on the one hand, but on the other hand, we can quickly identify their limitations which will help us in our dataset design. Each of the following datasets is usually explored in related problems and are widely used as a standard in various evaluation protocols and benchmarks, which contributes the most to our choice, since we can compare our results with different studies and researches.

In addition, we will also present our newly and fully annotated dataset (PAPSE) that directly results from our data collection in classroom environments. After presenting and clarifying our dataset, we will also explain our reasoning in the annotation procedure and how we accomplished it. We believe that this dataset can be very helpful in the solving of our main goal and provide data to test our hypothesis.

3.1 Related Datasets

To build an effective and precise student attendance registering system, we will need a lot of different images and scenarios to get a robust solution that can operate almost without any mistakes, resulting in a good learning process that is directly related to the chosen dataset. Since our work will be linked with students, it is logical to choose datasets with people involved in the available images. However, we also want to include general objects evaluation in our data validation since we may need to identify other objects in future works or even help identify a subject. Since, for now, our identification process will be done offline, we will not need this particular task to use videos in our evaluation. In the tracking task, we will test two different approaches based on regular sequences of a video, which means that we will not use disconnected images like in the previous task. The first approach is based on single object tracking, and the second one is based on multi-object tracking, using a different dataset. Finally, we will use a dataset to explore the face recognition module using aligned faces.

Several general conditions could be applied to datasets, and we will try to describe them in the following sections. In some cases, we have outdoor scenarios, while in others, we will have indoor ones. The same principle may be applied to the camera angle or the number of objects presented on each frame. In terms of colors, we will always choose colored datasets since we believe that color is an essential feature in our problem.

The following sections will describe each datasets' main characteristics and features, and we will also integrate some sequences to clarify our point, beginning with the object detection datasets and ending in facial recognition.

3.1.1 COCO

The COCO dataset [LMB+15] is a large database constituted by many samples to be used in different tasks such as object detection or segmentation that still is in continuous update, with new images added every year. In our particular case, we will be interested in the object detection stage with bounding boxes. In this stage, there are two different class scenarios: stuff and thing. The first one is related to more abstract elements such as grass or the sky, while the second one (that is the one we will use) groups elements like cars or people. The 2017 version of this dataset divides their objects into 80 different classes and has around 164 000 images distributed by three different splits, as table 3.1 below shows.

Table 3.1: COCO Dataset image distribution by split.

Split	Number of Images	Ground Truth
Train	≈ 118 000	Yes
Validation	≈ 5000	Yes
Test	≈ 41 000	No

As we can see in the table above, the train and validation splits are fully annotated at frame-level, using bounding boxes to identify each subject or object, which is very helpful to perform some experiments. Despite that, the test dataset is not, which means that we need to submit our data to a benchmark if we want to evaluate it. The COCO dataset provides mixed scenarios between indoor and outdoor and a large variety of camera angles and position like the ones illustrated in figure 3.1 below.

Figure 3.1: Class *Person* image examples from the COCO dataset.

3.1.2 PASCAL VOC

The PASCAl VOC dataset [EEVG+15] is a database that allows object class detection tasks, and that was closed in 2012, which means that there will not be further updates to the provided samples. However, it is still possible to continue to use their benchmark for evaluation purposes and continues to be one of the standard datasets in this field, which allows us to compare results with other people's experiments, providing robustness to our results. Despite not being the primary goal of our project, this dataset also contains segmentation elements and action classification. The last version of this dataset in the object detection classification contains two different splits, and it serves 20 different classes that include the Person class and has around 22 000 images distributed according to the described in table 3.2 below.

Table 3.2: PASCAL VOC Dataset image distribution by split.

Split	Number of Images	Ground Truth
Train/Validation	≈ 17 000	Yes
Test	≈ 5000	No

As we can see in the table above, one of the main differences of this dataset compared with the previous one is only having two splits which means that we need to separate the train and validation test if we need to do so. Another interesting fact is the substantial reduction in the number of images when we directly compare it with COCO, affecting the training procedure in a deep learning-based model, if used alone, due to the lack of images. If we analyze the existence of labeled images, like in the COCO dataset, we only have cover in the training/validation split at frame-level but without any representation in the test set. The PASCAL VOC dataset is also constituted by indoor and outdoor scenarios with a large variety in the number of identities per frame, and it does not present a sequential list of frames which means that they are independent of each other and some examples of these images can be found in figure 3.2 below.

Figure 3.2: Class *Person* image examples from the PASCAL VOC dataset.

3.1.3 VOT

The VOT dataset [KML+16] is commonly used to train and test single tracking methods in short tracking situations and long tracking situations. The most recent challenges of this dataset also have examples that allow testing real-time trackers. This dataset is one of the standard datasets, and results of different methods using these sequences can be found in a large group of different benchmarks, which is helpful to compare with our results.

As it was implicitly said before, this dataset provides videos that are split into sequential frames. The number of Frames Per Seconds (FPS) and the image size of each frame are variable. Even though an image has more than one element in some cases, the annotations are always linked with just one element for each image.

Table 3.3: VOT 2019 sequences specifications.

Sequence Name	Number of Frames	Image Size (Pixels)
agility	100	854 x 480
ants1	325	1024 x 1024
ball2	41	1280 x 720
ball3	171	480 x 810
basketball	725	576 x 432
birds1	339	1280 x 720
bolt1	350	640 x 360
book	175	640 x 480
butterfly	151	640 x 480
car1	742	640 x 480
conduction1	345	1280 x 720
crabs1	160	1280 x 720
dinosaur	326	320 x 240
dribble	111	960 x 540
drone1	355	1280 x 720
drone_across	147	1280 x 720
drone_flip	113	1280 x 720
fernando	292	640 x 480
fish1	366	460 x 345
fish2	310	640 x 360
flamingo1	1377	1280 x 720
frisbee	248	1920 x 1080
girl	1500	640 x 480
glove	120	640 x 480
godfather	366	432 x 240
graduate	844	560 x 240
gymnastics1	567	320 x 180
gymnastics2	240	1280 x 720
gymnastics3	118	1280 x 720
hand	267	320 x 240
hand2	145	960 x 540
handball1	377	384 x 288
handball2	402	768 x 576
helicopter	708	848 x 480
iceskater1	661	480 x 270
iceskater2	707	640 x 352
lamb	100	960 x 540
leaves	63	1280 x 720
marathon	101	960 x 540
matrix	100	800 x 336
monkey	101	960 x 540
motocross1	164	640 x 360
nature	999	960 x 540
pedestrian1	140	320 x 240
polo	111	960 x 540
rabbit	158	640 x 480
rabbit2	100	960 x 540
road	558	1210 x 681
rowing	101	960 x 540
shaking	365	624 x 352
singer2	366	624 x 352
singer3	131	854 x 480
soccer1	392	640 x 360
soccer2	129	1201 x 681
soldier	138	1280 x 720
surfing	120	960 x 540
tiger	365	640 x 480
wheel	101	960 x 540
wiper	341	640 x 480
zebrafish1	400	1920 x 1080

A more detailed description of this dataset is presented above on table 3.3 where clarification of each sequence is made. The **sequence name** represents the original name of those frames in the dataset, the **number of frames** is the total length of the sequence, while the **Image size** represents the size of each image in pixels. As we can see from the table above, we also have differences in the image arrangement beyond the sizeable image-size variability, including both horizontal and vertical images. This variability is explained by the different scenarios and different positions of the cameras in the recording procedure.

The most significant difference in this dataset when compared to the detection datasets presented in previous sections is the existence of sequential frames in opposite to the random and unlinked images in COCO or Pascal VOC, for example. The sequences presented in this dataset are usually related to activities such as playing basketball or doing gymnastics, or even cars in a traffic jam. We can see an example of a sequence taken from this dataset in figure 3.3 below.

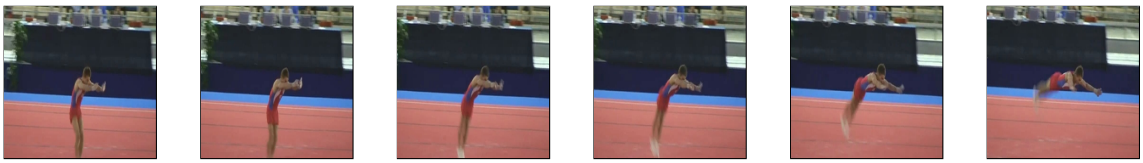

Figure 3.3: Sequence *gymnastics1* from the VOT2019 dataset

3.1.4 OTB-2015

The OTB-2015 dataset [WLY15] is a visual tracking benchmark that contains 101 widely used video sequences compiled from different datasets. Some of the video sequences presented here overlap with those described in the last section in the VOT dataset, which means that we need to be careful while evaluating our results. The authors also include a list of different attributes representing the challenges on each sequence and an indication of low resolution. Examples of those attributes are illumination and scale variations, occlusion or deformation, motion blur or fast motion, in-plane rotation or out-of-plane rotation, or even if some portion of the target leaves the view and, in some cases, background clutters.

Since it collects different video sequences from different datasets, the main rule from the last dataset is not followed here. Despite all video sequences are fully annotated on each frame, there are some cases where we have more than one instance being followed on each sequence. In those cases, the sequence name includes the number of the identity that is being followed. For example, if we have two people jogging in a sequence, we will have two different sequences representing each subject. Like in the previous single tracking dataset, OTB presents a large variability in image sizes and the length of the sequences. We can see a sequence sample taken from this dataset in figure 3.4.

Figure 3.4: Sequence *Woman* from the OTB-2015 benchmark.

3.1.5 MOT

The MOT dataset is widely used and has become a standard dataset when dealing with multi-object tracking experiments in single and long tracking situations. We can find several methods and researches that use this dataset in their experiments. Like in previous datasets, that factor was crucial for us to decide how to compare results.

Identical to other standard datasets, MOT has a challenge server associated with submitting results of our methods to validate our results. They also provide several video sequences with frames fully annotated for training purposes and different sequences for testing with raw images. Our research uses different versions of this dataset, namely MOT15 [LTMR$^+$15], MOT16 [MLTR$^+$16], MOT17 [MLTR$^+$16], and MOT20 [DRM$^+$20]. The main difference between these different drafts, in some cases, consists only in fixing annotations mistakes that were spotted by reviewers, although in some cases, new sequences were added. We present a basic overview in table 3.4 below with the number of sequences both in training and test sets on each studied version of the dataset.

Table 3.4: Overview of different sequences size on each Dataset.

Dataset	Number of Train Sequences	Number of Test Sequences
MOT15	11	11
MOT16	7	7
MOT17	21[1]	21
MOT20	4	4

[1] All MOT16 sequences are used with a more accurate ground truth.

As we can see from the table above, the four different datasets present a different number of sequences with the typical rule of only the training sequences being fully labeled. In the particular case of MOT17, in comparison with its previous version, 14 new sequences were added, and seven had their annotation fixed. It is essential to notice that some of the images of this sequence will overlap with MOT16, which is a factor in consideration in possible training procedures. In figure 3.5 we can see a sequence from the MOT20 dataset.

Figure 3.5: Examples taken from the MOT dataset training set.

3.1.6 LFW

The LFW is a public benchmark [HMBLM08] consisting of sets of face photographs with approximately 13.000 images collected from the web. This database is labeled on each face with the name of a subject in the image and includes around 5860 different person identities. Despite many subjects involved, only around 1680 of them include more than two different pictures while the others identities have only one image. Besides the regular images, this database also includes aligned images which usually improves the performance of different tasks related to faces, such as face verification or recognition.

Despite the robustness and widely use of this database, there are some limitations that we should notice before using LFW. Since we only have 13.000 samples, the database is probably not extensive or diverse enough to prove a definitive point, but it can be used to start our research. On the other hand, the lack of images and their alignment will also contribute to avoiding scenarios with occlusion or with faces in non-natural or random positions, which is not ideal for a real-world scenario. The authors also stated that some minor errors in the annotations are still without any fix not to overcomplicate the database. Therefore, we will need to decide either we use them or remove the identified subject when using the database. A few examples of this dataset can be found in figure 3.6.

Figure 3.6: Sample examples from the LFW dataset.

3.1.7 Wider Face Dataset

The Wider Face Dataset [YLLT16] is a publicly available face detection benchmark directly selected from the Wider dataset. Contrary to one of the limitations of the previous dataset, this one comprises 32 203 images which allow the inference of better conclusions since it has more diversity than the previous. Besides the more considerable amount of images, this dataset also has examples of images in different poses and expressions in different illuminations and scales. Beyond this classical data variation, it also has occlusion examples and even subjects using makeup which is rare in this type of database.

The data structure is constituted by three distinct sets: train, validation and test. Each of the previous sets has 61 different classes, and the total number of images is split into them and observes the following distribution: training (40%), validation (10%) and testing (50%). The total number of labelled faces ascend to 393 703 faces since one image can have groups of people opposite other datasets where each image corresponds to just one entity. The previously mentioned ground truth annotations are not available in the test set, which demands the submission of each user predictions in their private evaluation server. We can see a few examples taken from the training set of this database in figure

3.7.



Figure 3.7: Sample examples from the WFD dataset.

3.1.8 Face Detection Data Set and Benchmark (FDDB)

The FDDB [JLM10] is a dataset of faces that were directly extracted from the LFW dataset. This dataset was explicitly designed to help find a solution to the face detection problem, and it aggregates around 2850 images. We can find 5171 faces in this set of images that are annotated according to several criteria. For example, even if a face is well visible in an image, it will not be annotated due to size restrictions, namely in height and width. The authors also consider other factors such as low resolution, occlusion or head-pose as a constraint to the annotation procedure.

The annotations are provided in 10 different folds which makes possible the evaluation of each one of them or the global performance. The main criteria in the annotations follow the non-ambiguity rule, which allows the dataset to be consistent but takes some variability and challenging cases from the available data. Some examples form this dataset may be found in figure 3.8 below.


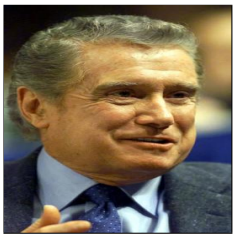

Figure 3.8: Sample examples from the FDDB dataset.

3.2 PAPSE-UBI: Prevent Abandonment and Promote Success in Education at the Universidade da Beira Interior

After we analyzed the previous datasets, we understood that despite some of them present good value to support our research, there were some limitations. The first and most obvious one is the lack of images in a classroom in those datasets. We could not find a dataset that has images from the specific environment that we want to test. Another limitation that was noticed in some datasets was the intentional discard of faces that were not in standard formats, and some cases where images were aligned to boost the consistency of

the dataset, but in a real-world scenario, this is not a good solution since people will move and will not always be in regular formats. On the other hand, and even though some are very rich and robust large-scale datasets, they do not fully annotate at the frame-level in the testing set. Having this in mind, this dissertation aims the develop a new large-scale dataset that satisfies the previously mentioned requirements. Specifically, we present the PAPSE-UBI dataset, a fully annotated large-scale dataset at the frame-level and pixel-level to monitor students' identification and behaviour within a non-cooperative environment.

We will present in the following sections our reasoning in the building of our dataset, starting by explaining the technical procedures that allow us to record and transmit data, the related privacy issues with this procedure, following by data collection sessions and statistics. We will also explore the data variation factors, ending the chapter with the data annotation procedure.

3.2.1 Data Transmission and Camera Specifications

In order to create our dataset, it was necessary to find an efficient and scalable way to capture and transmit our data as quickly as possible, always guaranteeing its privacy and security. The first step we took to operationalise this part of the project was to analyse all the image capture equipment, namely cameras and lenses, available in our research laboratory (SociaLab). Then, based on previous work, it was possible to access the SociaLab inventory reflected in tables 3.5 and 3.6 below.

Table 3.5: Available Cameras at SociaLab.

Manufacturer	Model
Canon	EOS 5D
JAI/Infaimon	BM-141GE
Stingray F504B ASG	073661
JAI/Infaimon	AD-080GE
Infaimon/IDS	5481VSE-C-SD32
IDS	UI-1240ML-C-HQ
Protos	NCL580
Canon	XM2
uEye	UI-1225LE-M / 080056

Table 3.6: Available Lens at SociaLab.

Manufacturer	Model
Nikon	Nikkor/GMZ3D85900MCN
Infaimon	GMTHR33520MCN
Infaimon	OPT-MHR47518MCN
Infaimon	HR F2.8/50mm
Infaimon	OPT-MHR38014MCN
Infaimon	302298
Infaimon	OPT-MN38014MCN-1
Infaimon	11-1248
Canon	TV3514
Canon	ET-83C

As it is possible to see from the tables above, there are several possible combinations of tables and lenses, so we needed to understand if all cameras and lenses were compatible or if some needed to be excluded from our evaluation. After careful consideration, we realised that some of the cameras and lenses previously listed could not be used, either because they were damaged or because they were not compatible with any of the models we had at our disposal. After this analysis, the data capture equipment we had at our disposal was reduced to what is presented in tables 3.7 and 3.8.

Table 3.7: Selected Cameras.

Manufacturer	Model
Canon	EOS 5D
JAI/Infaimon	BM-141GE
Stingray F504B ASG	073661
JAI/Infaimon	AD-080GE
IDS	UI-1240ML-C-HQ
uEye	UI-1225LE-M / 080056

Table 3.8: Selected Lens.

Manufacturer	Model
Infaimon	GMTHR33520MCN
Infaimon	OPT-MHR47518MCN
Infaimon	HR F2.8/50mm
Infaimon	OPT-MHR38014MCN
Infaimon	302298
Infaimon	OPT-MN38014MCN-1
Infaimon	11-1248
Canon	TV3514
Canon	ET-83C

Given the above combinations, and again based on previous work carried out in this laboratory, and which consisted of capturing several sessions with all possible camera and lens combinations, we understood that the best pair to use would consist of the IDS pair (UI-1240ML-C-HQ) with the Infaimon lens (302298). We can see an example of a capture using this configuration in figure 3.9.

Figure 3.9: Captured Frame with the chosen setup.

Despite all the experiments performed and the use of the best possible combination, we believe that this configuration would not serve the interests of this work since the image quality, namely by the lack of resolution and the somewhat dark image, would compromise the data capture, adding from the beginning factors that would harm the data quality, compromising the performance of future models.

Thus, one of the first tasks that we associated with the project was the construction of a technical setup that not only allowed a higher quality data capture but also allowed data transmission more efficiently, since the previous alternative required the use of memory cards or cables directly connected to a server in our laboratory. However, since we did not have in our laboratory a good camera to perform the data capture that we needed, and after reading many reviews and talk with professional people in the camera market, we

conclude that the best camera to perform our task and fulfil the requirements previously mentioned was the **DAHUA-2299-FO** with the characteristics presented in table 3.9.

Table 3.9: Technical Specifications of the camera.

Characteristic	Description
Compression Formats	H.265 / H.264 / MJPEG
Lens/Zoom	2.8mm fixed
Viewing angle	102 ° (H), 71 ° (V)
Network protocols	HTTPS, TCP, UDP, ARP, RTP, RTSP
Maximum Digital Resolution	5MP (2592x1944)
Stream	1080P
Power Supply	12V
PoE	Supports

As we can see in the table above, this camera supports the Real-Time Streaming Protocol (RTSP), which will be helpful in our data transmission as well as the Power over Ethernet (PoE) that will promote the communications between our setup and the server where the data will be stored. In figure 3.10 we can see the camera that was successfully installed in the 6.03 classroom of the Computer Science Department in UBI.

Figure 3.10: Installed setup.

This camera was installed in the right part of the classroom to capture a large part of it. If the camera was placed in the centre, we could probably capture the entire classroom, but due to the restrictions explained in section 3.2.2, this was the best possible solution. We will store our data in a computer placed in the SociaLab, and to do so, we need a way to send the captured data in the classroom to our laboratory. As we can see in one of the pictures presented in figure 3.10, the camera is connected to a router connected to a port of the university's internal network. To do so, we need to ask for permission from the UBI informatics centre that added our camera address and IP to the main router of UBI. After the

conclusion of that process, we use the RTSP to connect the camera with software that we develop to store the data. This solution is spotless because since the camera is connected to the same internal network that we can only access in the laboratory, we do not need to use cables to directly connect the camera to the server, which would not be accessible due to the infrastructure limitations, i.e., the classroom is in a different floor of the laboratory. We believe that with this setup, we can improve both the data quality and its transmission.

3.2.2 Privacy Issues

Privacy and data protection are a concern of ours in this work. Therefore, in order to ensure that the project would run without any problems in this regard, we contacted two different entities in order to present the concept and what we intended to carry out and, taking that into account, what would be the rules we would necessarily have to respect in order to obtain their approval. Given that both the data storage and transmission part and the data capture itself raises ethical and data protection questions, we contacted UBI's ethics commission and its data protection office for clarifications.

The main problems that were pointed out to us by both entities regarding data storage were the need to guarantee that the database would be anonymised entirely, that the data could only be manipulated within the university and only for academic purposes and that it would be necessary to have authentication to access it. Concerning data capture, we were asked that all participants were duly informed of the purpose of the capture, what the expected contribution of their participation would be and the risks associated with it, always ensuring the possibility of withdrawing at any time. After all these clarifications, the participant would need to sign a document authorising their participation. It was also requested that part of the room not be captured by the camera to ensure that if the recording occurred in an actual class, and some students did not want to participate, they could do so without losing the lesson. Finally, it was also requested that no sound should be captured as this would go beyond what was necessary for this work and could be considered intrusive.

In order to guarantee that all the previous assumptions were met, we ensured that the data is only accessed by a limited number of people and after authentication and can only be accessed when connected to the university's internal network. Furthermore, to guarantee the anonymization of the database, we associate a numeric ID to each participant, never associating the name of the subject and his/her ID in the dataset. As we are also only interested in capturing images, the requirement of no sound recording is reasonable and was accepted. Naturally, the data was and will be used only for academic purposes, and the camera was set up so that there were places outside the capture zone. Even if the camera is turned on, images are only recorded if we activate the protocol in the laboratory, which gives us total control over the data recording process, avoids the recording of unauthorized classes, and helps to control the amount of disk space that we need. Finally,

we asked all participants to sign a form expressing their intention to participate in the project, and the project ran without any significant problems.

We think that by consulting these two entities that approved our project and with the current data protection law in force, our work could be done without any concerns, and we hope that our research could achieve great results while respecting people's privacy.

3.2.3 Data Collection

Since we want to reduce university dropout rates and one of the main reasons that contribute to this problem was the attendance register, we create the PAPSE-UBI dataset intended for automatic student presence registration and student behavioural analysis during class. However, this last part does not directly belong to the scope of this master's dissertation. At the moment, our dataset is composed of more than 1 hour of video images from six different class sessions with 45 distinct subjects, recorded on different days to ensure variability of the light conditions and from the subjects involved. We can see a matrix representation of the attendance of each voluntary in figure 3.11. The initial goal of this work was to record actual lessons with a fixed group of classes in order to be able to make more valuable comparisons that were not possible due to lockdown. The solution found to vary this problem was to conduct simulated class sessions, and having in consideration the Covid-19 pandemic, some health precautions were adopted during session recordings, such as maximum people in the classroom (20 people) and social distancing. Besides the small group of people presented in the sessions, they were recorded in short periods (around 10 minutes) to minimize possible transmissions.

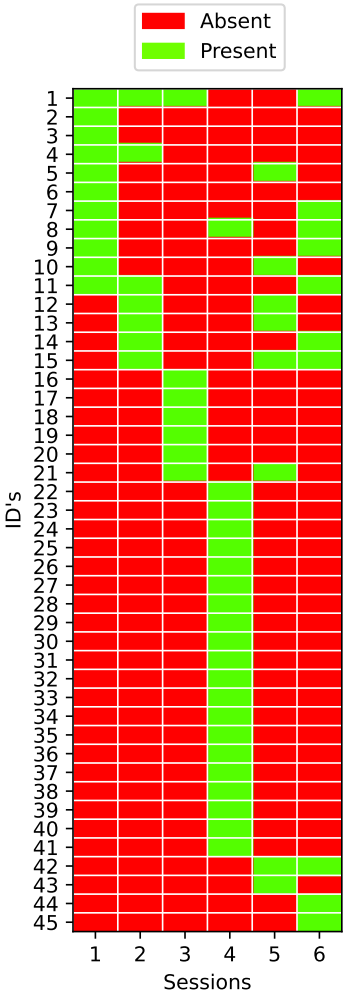

Figure 3.11: Presence matrix of 45 voluntary students over 6 sessions.

In order to be able to maintain consistency in the data and ensure that the actions are as natural as possible, thus ensuring that the fact that this is a simulation of a lesson does not make the sessions monotonous and predictable, we give each subject a profile to simulate all the possible student scenarios. For example, we will have good behaviour students who will pay attention to the class and take notes, but we will also have bad students profile using a cellphone or are continually talking and backwards. Some students will wear masks, and others will not make that decision

personally and without any constraints. Since facial recognition systems were massively affected by people wearing masks, considering that essential facial landmarks are now hidden, our dataset contributes to solving this problem by providing mask usage data. We will record our sessions in an indoor classroom environment, and we will try to have the most variable data possible. Each session begins with the students entering the classroom, individually or in groups, going to their respective places in the classroom and simulating an entire class behaviour from the unpacking of the backpack, ending with the respective packing of the backpack and leaving the room. The session ends when the last student leaves the room. As stated earlier, this dataset consists of 6 sessions and 40,427 images divided by each session as shown in figure 3.12.

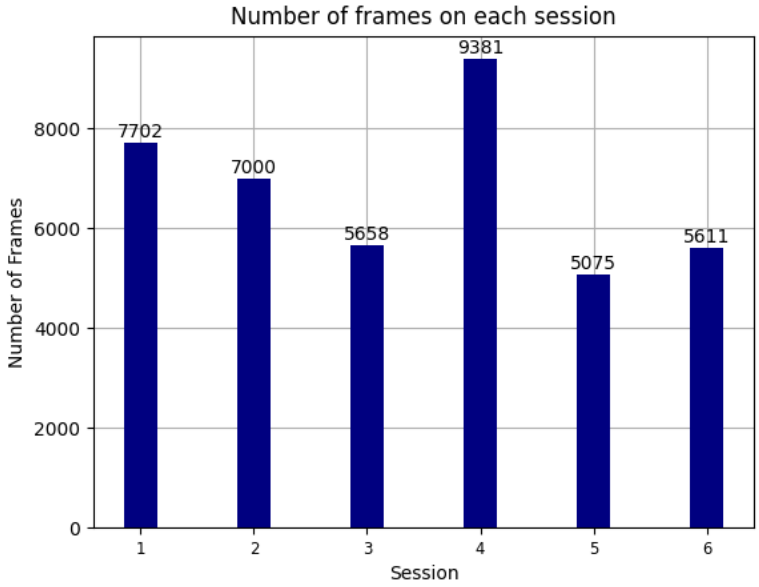

Figure 3.12: Frames per session.

As we can see in the figure above, there was always care taken to try to have a consistent number of images in each session. The major exception is session four since it was the only possibility we had to record an actual class, so the timing of finishing the recording coincided with the end of the class decided by the teacher in charge. Despite the more significant number of images in this session, we thought it would be helpful to get images of a real class, not only because of the more significant number of students available but also to ensure a better representation of the natural context, which would allow us to assess whether or not our simulations were meeting our goal and we believe that this was the case.

One of the tasks that could have been useful would have been to repeat the sessions with the same subjects to make sure that we would get good representations of the same individual with different clothes and possibly also different behaviours in each session, increasing the number of images we would have of each individual and therefore a better representation of him.

Just as we always tried to balance the number of images associated with each session, we also always tried to have a similar number of men and women throughout the dataset so that there would not be a bias in the dataset. Unlike the recording time, this data did not depend exclusively on us since we were dependent on the available volunteers. In figure 3.13 we can see the gender distribution per session.

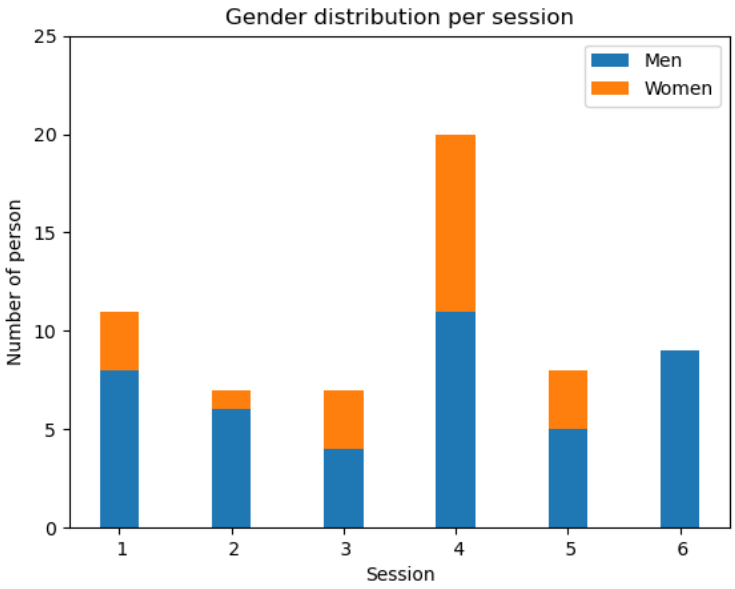

Figure 3.13: Frames per session.

As we can see in the image above, the number of men present in each session is always higher than the number of women, and in most sessions, there is some balance. The most prominent exceptions are sessions 2 and 6, emphasising the latter since all the nine participants were men. On the other hand, the most balanced session was session 4, where from 20 people, 11 were male, and 9 were female. Although there are only 45 different subjects, the chart above includes the participation of those in more than one session, bringing the total to 62 people, where 43 are male, and 19 are female. Despite this factor, we believe that in this first phase of the project, such difference will not be relevant since, despite occasional exceptions, we will have enough representatives of each gender to ensure correct learning of the dataset.

Since one of the long-term goals of this work is the correct facial recognition of each subject, it seemed particularly important to present the statistics of the mask usage in each session. These data can be seen in figure 3.14.

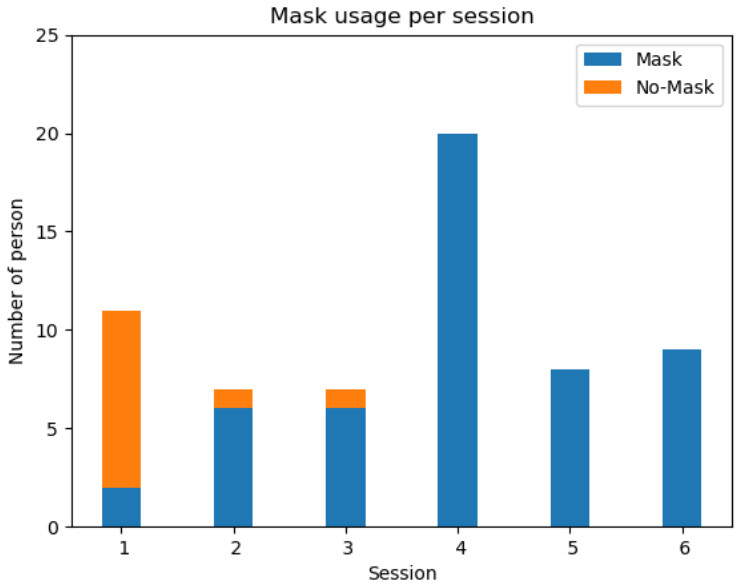

Figure 3.14: Mask usage per session.

Observing the figure above, we can see that in the first session, it was observed that the vast majority of volunteers did not wear a mask, leading to the fact that the collection of data from this session could favour the development of a dataset for facial recognition. However, unfortunately, this was the exception compared to the other sessions, justified not only by the free will given to each participant but also by worsening the health situation, particularly visible from session 4, leading to all participants wearing a mask. This scenario is particularly limiting for good face recognition since most of the critical features for this recognition are covered by the mask, but it turns out to be a good contribution at a time when face recognition has been trying to reinvent itself.

3.2.4 Data Variation Factors

This section will discuss some of the data variation factors that can make the dataset more competitive and present some cases that we consider more unusual compared to standard datasets. As discussed in part in the state-of-the-art, some factors hinder the detection process, namely in people and faces. The use of masks appears one of the most recent factors influencing this process, but occlusion or differences in illumination also affect the process.

As far as the collected images are concerned, they all have the same size and resolution and therefore, there is no advantage of one over the other. Regarding the positioning of the camera and possible changes in image quality, this will not occur as the camera is fixed in the same place since session 1, not contributing to any disturbance of the data. Since we are dealing with an indoor environment, we do not expect external factors such as rain or foggy to influence the data.

We can see in figure 3.15 a classic example of our data which, in this case, belongs to session 1 and which will serve as a starting point for the clarification of most of the images present in the dataset and which consists of a group of people in a classroom, simulating an academic environment. In the image below, we can identify people who are paying attention, and others who share content on their cellphones. Others are taking notes and, while doing so, lean in their chair, making it difficult to see their body and face correctly.

We can also see that although most people are spaced apart due to pandemic constraints, we have one example in the centre where people are closer together to increase the variability of the data. These different actions result directly from our idea of assigning profiles of good and bad students, thus ensuring natural classroom behaviours, even if in a simulated session.

Figure 3.15: Example of a classical frame from session 1.

We can also see in figure 3.16 that a subject is walking in the background, and it may be harder to find its face because the space between the camera and the subject is large, the subject is moving, and there are other people in between. People wearing masks may also influence accurate face detection since part of the critical features in a face, such as a mouth or a nose, are occluded. This scenario was also one of the proposals we made to increase the variability of the data by simulating the entrance of a late student into the class while everyone else is already adequately seated.

Figure 3.16: Example of drawback: Walking.

In figure 3.17 another situation occurs since one of the subjects in the front line is packing and are partially occluded by the backpack but is also occluding the subject behind him. A dataset must have these kinds of examples because while it certainly makes detection difficult and may initially contribute to some errors, it is a situation that will occur in a classroom, so it is a real-world scenario that has to be considered.

Figure 3.17: Example of drawback: Partial Occlusion.

In image 3.18 below is one of the most challenging situations in this dataset, where we have the largest group of people in the dataset and who, for that reason, have the slightest space between them when seated and crowd easily at the entrances and exits of the room.

Figure 3.18: Examples from a big set of students and also a secnario where everyone is leaving the room.

This session was, as previously mentioned, an example of a real class, and as we can see, the number of people present is much higher than the previous images, which will allow the training model to also have contact with a number closer to the real one in a class, ensuring greater realism to the dataset.

We believe that this dataset will be able to provide solutions to the challenges we initially set ourselves, namely in what it will allow us to detect people and their faces, also presenting several occlusion scenarios, people in groups and having seated people closer and further away from the camera, extending the variability of the data. There are also differences in lighting, as several sessions were recorded on purpose at different times of the day, with lights on and off. To add more value to this dataset, we will also perform its annotation, explained in the next section.

3.2.5 Data Annotation

Any artificial intelligence system has as the primary key the quality of the data and its variability, but also the quality and detail of its annotation process, the latter being a time-consuming and expensive process, leading many standard datasets, like the ones we present, to only include them in their training set. Our option in this work was to fully annotate at a frame level, using bounding boxes, the 40 000 images, thus contributing in a valuable way to the elaboration of a complete dataset. In this section, we will also detail the annotation process, explaining the decisions we made to make the process transparent, starting by demonstrating the software used for the annotation and clarifying its functionalities. We will then list the various attributes present in the annotation and the number of bounding boxes per subject, ending with some annotation examples.

In figure 19 we can have an overview of the CVAT software that was used in the annotation process. The main functions that we use in the annotation process are also labelled in the same picture.

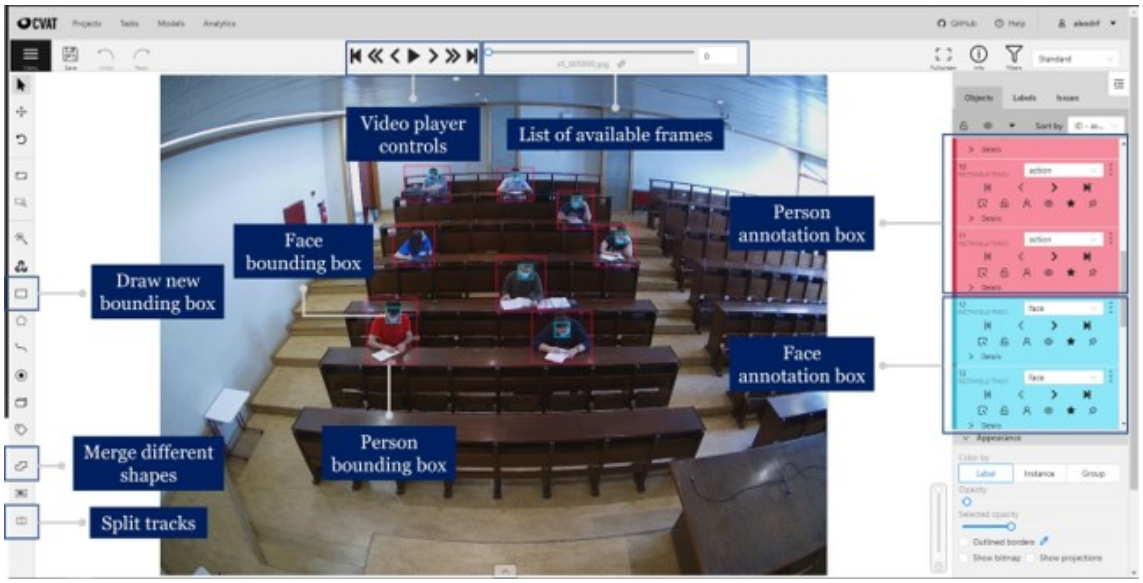

Figure 3.19: Overview from the CVAT software.

The annotation process is started after the upload of the images is completed. In the sessions that contained a high number of students, pre-processing was also carried out that allowed the automatic annotation through the use of a people detector that facilitated in some cases the annotation but did not replace it because adjustments are always necessary and, in some cases, even from scratch because the automatic annotation was incorrect.

In the figure above, we can see the main functions used and that we are going to clarify. If, in some cases, like the video player controls or the list of available frames, their names are straightforward and leave no room for doubt, it is essential to clarify the concept, for example, of merged different shapes. We can connect consecutive frames with this function, leaving our dataset prepared to deal with problems that need to use tracking, being the inverse operation to this one in charge of the split tracks button. In the image, it is also possible to see that all the subjects have two bounding boxes drawn, one that goes around the subject's face and another that borders their body. The decision to give the dataset these two bounding boxes was due to the future need to be able to perform facial recognition and track each subject's behaviour. Additionally, apart from the student's identification in the face bounding box, we also provide gender information and mask usage. The body's bounding boxes also have that information in addition to the action class.

In addition to the attributes previously mentioned and familiar to both the annotation of the student's face and body, some attributes are unique to the body and result directly from the future need to study behaviour and understand its actions. Table 3.10 below lists these attributes and their description.

Table 3.10: Body attributes in PAPSE-UBI dataset.

Action	Description
Stand Up	It occurs when the student gets up from the chair.
Backwards	A student is facing backwards and the face is not visible.
Cellphone	The student uses his mobile phone and it is visible.
Crossing Arms	Represents the course of the action and is deactivated at its end.
Drink	Whenever a student is drinking and the drink is visible.
Eat	Whenever a student is eating and the food is visible.
Give - Receive	It represents the delivery of an object to a subject.
Glasses	Every time someone adjusts their glasses.
Laugh	Every time someone laughs or smiles.
Packing	Represent storing objects in a schoolbag or pencil case.
Unpacking	Represent removing objects from a schoolbag or pencilcase.
Put Mask On	-
Put Mask Off	-
Raise hand	When the subject raises his hand or arm.
Salute	When the subject greets someone.
Sit Down	It occurs when the student sits on the chair.
Speak	The action occurs when the mouth is visibly moving.
Walking	Whenever the student is standing and moving around.
Whisper	When the student has a visible mouth and speaks next to another student.
Write	Whenever the student writes or leafs through books.
Show Notes	Show the notebook or sheet to someone.
Poke	Every time the student touches someone.

All the attributes present in the table above result from actions that occurred more than once in the dataset and that we consider could be relevant to infer future behaviour. The dataset is coherent since all sessions consider the attributes mentioned here, even if some were only considered in a more advanced phase of the annotation, proceeding afterwards to correct the already annotated images. Similarly to the bounding boxes, these attributes are also annotated in all the dataset images, making them suitable for a future action recognition evaluation. This annotation is always done at the subject level except for two actions, namely Give-Receive and Show Notes, which are annotated simultaneously on both people involved. As far as the frequency of actions in the dataset is concerned, there is a clear predominance of more directed towards a classroom, but we have always taken care that all actions occur in the simulated sessions.

In figure 3.20 below we can see three examples taken from three different sessions. These images are the result of viewing the respective annotations applied to their source images. Note that the bounding box representing the face is green in the figures below as long as the subject is wearing a mask and turns blue whenever the mask is not on. The body bounding box is always red in both cases.

Figure 3.20: Annotations examples from PAPSE-UBI dataset.

In the images above, we can see six examples of our annotated images. Starting from the top row and from left to right, we can see three examples of different situations. The first image denotes the annotation of many subjects, but they still have some space between them, not making the annotation particularly difficult. On the other hand, the second image shows a more sensitive situation with several people crowded together, so the annotation must be meticulous in defining the limits of each subject, but also to make sure that nobody is left unannotated. Finally, the third image intends to show only one of the cases in which the annotation is performed simultaneously in two different people for the same action. Moving to the bottom row, we find more elucidative examples of the colour difference in the bounding box of people with and without masks and examples of people entering and leaving the room, corresponding to the beginning and end of each session.

We believe that the presentation of this dataset will be an added value in solving the original problem of this dissertation due to the variability that it presents and the number of images fully annotated frame by frame, in its entirety, something rare about datasets of this size. Nevertheless, on the other hand, by also including information of the actions that the subjects do throughout the sessions and joining all the images in single chains, we will allow that in the future, the dataset can also be used for tracking and action recognition, exceeding the primary purpose for which it was created.

Chapter 4

Experiments and Results

In order to make concrete evaluations of our dataset, it is vital to understand which state-of-the-art methods present the best results and which are the optimal configurations. In this way, in this chapter, we will summarise several state-of-the-art methods for several problems such as detection, particularly of people and faces, tracking, and some tests in face recognition. For the detection problem, we will use the standard COCO 2017 and PASCAL 2012 datasets, while for the tracking problem, we will perform a comparison using various versions of the MOT dataset for multi-tracking and the VOT dataset for single tracking. We will always use these datasets to compare state-of-the-art methods so that it is fair and equal for all.

Finally, we will finish this chapter with an evaluation of the detection task already in our dataset, making a comparison between the method that has obtained better detection results in standard datasets, training from scratch on our data and also through a finetuning on our data, thus being able to answer the original problem regarding the detection of people and faces.

4.1 Subject Detection

After the literature review, we could identify some methods that have obtained better results in recent times. Thus, we chose YOLOv3 and YOLOv5, SSD, Mask R-CNN and Efficient-Det as the methods to compare on the COCO 2017 and PASCAL 2012 datasets. Therefore, all results presented here resulted from pre-trained weights on the COCO 2017 or finetuned weights on PASCAL VOC using various configurations.

Regarding the evaluation protocols, we always use the COCO test-dev server since this dataset has no annotations in the test set, and this is an efficient way to compare results. Nevertheless, the predictions used and that were submitted were performed locally by running all the methods. On the other hand, due to the impossibility of using the PASCAl VOC server, which was discontinued for several months at the date of this publication, it was decided to use the validation set to compare the methods. Both datasets have several classes, but for consistency with the chosen problem, we only compare the Person class through Mean Average Precision (mAP) and Average Precision (AP) with different IoU. We can see in table 4.1 the results of our experiments concerning subject detection using this protocol.

Table 4.1: Performance summary of current object detection methods trained on COCO dataset executed over class **person** of COCO-testdev [LMB+15] and finetunned on PASCAL VOC 2012 [EEVG+15] and executed over class **person** of the validation set.

Method	**mAP** ↑	**mAP$_{50}$** ↑	**mAP$_{75}$** ↑
COCO test-dev | ***Person*** [LMB+15]			
Mask R-CNN [HGDG18]	0.399	0.705	0.404
YOLOv3 [RDGF16]	0.505	0.792	0.534
SSD-300 [LAE+16]	0.337	0.635	0.321
SSD-512 [LAE+16]	0.396	0.706	0.392
EfficientDet-D0 [TPL20]	0.422	0.686	0.439
EfficientDet-D1 [TPL20]	0.485	0.753	0.515
EfficientDet-D2 [TPL20]	0.525	0.789	0.561
EfficientDet-D3 [TPL20]	0.560	0.817	0.605
EfficientDet-D4 [TPL20]	0.586	0.839	0.637
EfficientDet-D5 [TPL20]	0.604	0.855	0.661
EfficientDet-D6 [TPL20]	0.612	0.859	0.669
EfficientDet-D7 [TPL20]	0.628	0.869	0.686
EfficientDet-D7x [TPL20]	0.631	0.870	0.684
YOLOV5S [JSB+21]	0.502	0.776	0.540
YOLOV5M [JSB+21]	0.569	0.819	0.620
YOLOV5L [JSB+21]	0.594	0.837	0.647
YOLOV5X [JSB+21]	**0.637**	**0.878**	**0.690**
PASCAL VOC 2012 val | ***Person*** [EEVG+15]			
Mask R-CNN [HGDG18]	0.619	0.939	0.706
YOLOv3 [RDGF16]	0.335	0.651	0.306
SSD-300 [LAE+16]	0.425	0.795	0.417
SSD-512 [LAE+16]	0.493	0.8270	0.481
EfficientDet-D0 [TPL20]	0.554	0.826	0.628
EfficientDet-D1 [TPL20]	0.598	0.875	0.660
EfficientDet-D2 [TPL20]	0.629	0.890	0.624
EfficientDet-D3 [TPL20]	0.649	0.902	0.718
EfficientDet-D4 [TPL20]	0.676	0.918	0.748
EfficientDet-D5 [TPL20]	0.681	0.920	0.753
EfficientDet-D6 [TPL20]	0.693	0.925	0.772
EfficientDet-D7 [TPL20]	0.694	0.924	0.770
EfficientDet-D7x [TPL20]	0.691	0.922	0.765
YOLOV5S [TPL20]	0.540	0.834	0.639
YOLOV5M [JSB+21]	0.591	0.852	0.668
YOLOV5L [JSB+21]	0.613	0.893	0.694
YOLOV5X [JSB+21]	**0.701**	**0.932**	**0.768**

Regarding the methods used, five different methods are presented in the table above, and in three of them, we also present the results of several configurations. In the case of Mask R-CNN, only the standard configuration of that method was tested. In the case of YoloV3, the results presented correspond to the use of the pre-trained weights of Yolov3-spp-ultralytics with an image size of 608, being further on, namely in figure 4.1 a visual comparison of some changes in image size with different weights presented. Another of the methods explored was SSD, represented here with two configurations using different input image sizes. The same technique of increasing the image size was also followed in several tests to the EfficientDet and, in this case, the minimum image size used, corresponding to the EfficientDet-D0, was 512, while the maximum, applied in the EfficientDet-D7X

was 1536. Finally, four different YoloV5 configurations were also tested, being that in this case, the image size is constant and 640, according to the authors' recommendations, being the difference between the various models related to the depth of the model and the layer channel.

Analysing the results, namely in the COCO dataset and using the mAP as a comparison criterion, we realise that the method with the worst performance is the SSD-300, although there is an improvement in its performance when increasing the input image size, still not enough to consider it as an option to solve our problem. At the same level, we find Mask-RCNN. Also, the YOLOv3, even with the more significant input image configuration, could get a result only slightly higher than 0.5. Comparing the several configurations by EfficientDet, we also noticed that the previous trend is maintained, i.e., when we increase the image size, the mAP improves gradually, obtaining its maximum value in the EfficientDet-D7X, that is our second-best result, only surpassed by YOLOv5X. The last one also presents the advantage of having a smaller input image size, leading to a faster response time than the previous one and managing to obtain a better global result. It is also interesting to note that YOLOv5X obtains much more significant results than all other methods, even at its best settings. On the other hand, the worst configuration of YOLOV5X is also quite competitive with most of the previous results. If we take into account the mAP_50, the result improves even more, as expected, and still manages to outperform all other methods.

Observing the results also obtained for the PASCAL VOC 2012 dataset, we understand that the previously described tendencies are maintained, but they manage to obtain even better overall performances. This pattern makes us consider that eventually, this dataset may be easier to learn than COCO, possibly because it has fewer classes and, as such, needs to distinguish fewer features, but it may also be related to the fact that the specific examples of the Person class in this dataset are smaller when compared with COCO. We highlight particularly the case of Mask R-CNN that obtained a brutally superior performance in this dataset, even though in general terms YOLOv5X still has the best performance, presenting the best results in both datasets, which will justify our choice as the ideal method for solving our problem.

As we said before, in most cases, there seems to be a relationship between increasing the input size of the image and its performance. However, likely, this relationship will not persist indefinitely, if only because of computational limitations. The figure below tries to make this relationship clearer, using four different weights from YOLOv3 and four variations in input image size, namely 320, 416, 512 and 608. This comparison aims to understand if there is any sign of convergence in the size of an image and its respective performance, on the one hand, the other hand, if the differences between several more complex configurations tend to limit the improvement of this performance.

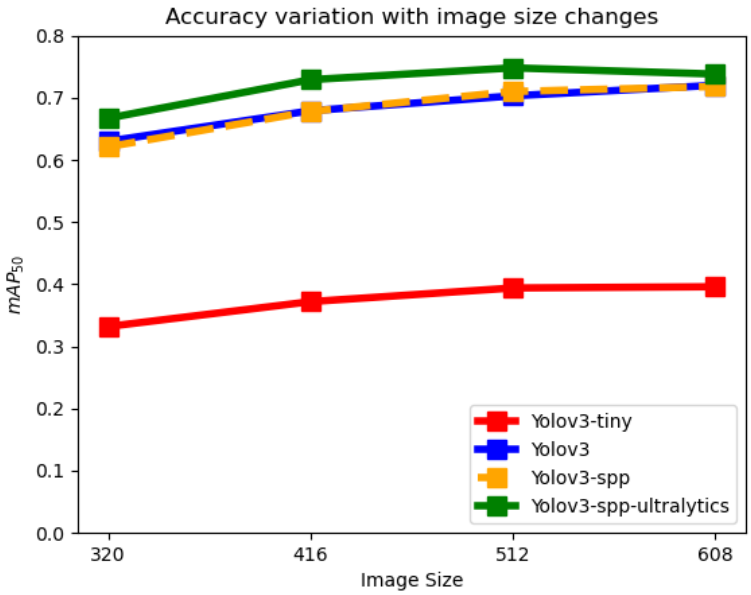

Figure 4.1: Accuracy variation in YOLOv3 using four different configurations with four different image sizes.

One of the first conclusions that we can draw by looking at the figure above is the apparent difference between the most straightforward configuration, in red, and the others. We can also understand that the original version of YOLOv3, in blue, and the small changes made in the yellow dashed model, do not translate into a very significant improvement, and there is practically an overlap between the two models. The model in green is the one with better results, being the one included in table X. Another conclusion to be drawn to do with the visual confirmation of what was discussed above: the accuracy of the models seems to increase with the input size of the image. However, like the hypothesis that had been put before, the growth seems to reduce as we enlarge the image, and the three best models all end up converging to similar areas in the graph.

4.2 Face Detection

After exploring people detection, it is also relevant to perform some experiments regarding face detection because even though both are detection tasks, in the specific context of this problem, it is important to distinguish between these two classes.

To make a fair comparison, we will use the same method, DSFD, in two different datasets: FDDB and Wider Face. In both these datasets, we will evaluate them in their different subsets: in the case of FDDB, they divide the dataset into ten random different subsets from the validation set, while on Wider Face, they only consider three different subsets based on the dificulty of the face detection. In the end, we will calculate the average performance of the method in all subsets.

Table 4.2: Accuracy on FDDB dataset.

FDDB	Accuracy (%)
#1	97.32
#2	97.76
#3	96.98
#4	98.48
#5	98.53
#6	97.29
#7	97.69
#8	96.01
#9	97.77
#10	97.85
Average	**97.55**

Table 4.3: Accuracy on Wider Face dataset.

Wider Face	Accuracy (%)
Easy	96.70
Medium	95.74
Hard	90.54
Average	**94.33**

Looking at the two tables above, the first conclusion we can draw is that DSFD had a better average performance on the FDDB dataset when compared to Wider Face, achieving about 3% more average accuracy. Analyzing in more detail the FDDB dataset, we realize that the accuracy value of each subset of the dataset is quite similar in all of them, which makes sense since choosing images randomly will allow, in most cases, to obtain balanced datasets since the initial dataset is also balanced. As for the Wider Face, we notice that the average accuracy goes down over the three sets, which also makes sense since we are continually increasing the difficulty of face detection, i.e., we are presenting more ambiguous examples or where there are factors that contribute to the detection detriment, such as occlusions. These experiments of the same method on two different datasets are only intended to have an order of magnitude of the expected values for face detection, not necessarily representing a definitive comparative factor but rather a beginning of a review of this subject.

4.3 Subject Tracking

As in the previous section, we believe it would be essential to perform a comparative analysis of different tracking methods for the future implementation of the non-cooperative attendance registration system, particularly in this part for behavioural analysis. Therefore, to obtain a complete analysis, we will analyze single-tracking and multi-tracking methods, comparing the former by using several versions of the MOT dataset. The methods chosen after the literature review were SiamRPN and SiamMask for the case of single-tracking and Tracktor, JDE and FairMOT for the specific case of multi-tracking.

We will use as metrics in multi-tracking, Multi-Object Tracking Accuracy (MOTA), the ratio of correctly identified detections over the average number of ground-truth and computed detections (IDF1), and the number of IDs found. As for single-tracking, we will use accuracy, robustness that is defined as the failure times, and Expected Average Overlap (EAO) for single-tracking. We can see the obtained results with multi-object tracking methods in table 4.4.

Table 4.4: Performance summary of current multi-object tracking methods trained on MOT17 [MLTR$^+$16] and finetunned over the MOT Challenge datasets 2DMOT15, MOT16 [MLTR$^+$16] and MOT20 [DRM$^+$20]

Method	**MOTA** ↑	**IDF1** ↑	**IDs** ↓
2D MOT 15 [LTMR$^+$15]			
Tracktor [BMLT19]	0.521	0.611	242
JDE [WZL$^+$20]	0.429	0.618	572
FairMOT [ZWW$^+$20]	**0.855**	**0.848**	**129**
MOT 16 [MLTR$^+$16]			
Tracktor [BMLT19]	0.577	0.652	170
JDE [WZL$^+$20]	0.731	0.689	1301
FairMOT [ZWW$^+$20]	**0.875**	**0.868**	**481**
MOT 17 [MLTR$^+$16]			
Tracktor [BMLT19]	0.617	0.658	292
JDE [WZL$^+$20]	0.743	0.691	1343
FairMOT[ZWW$^+$20]	**0.838**	**0.819**	**553**
MOT 20 [DRM$^+$20]			
Tracktor [BMLT19]	0.166	0.211	2936
JDE [WZL$^+$20]	0.459	0.375	18875
FairMOT [ZWW$^+$20]	**0.742**	**0.796**	**2601**

As we can see from the table above, the FairMot method consistently obtained the best MOTA and IDF1 results compared to the other methods, regardless of the dataset used. Furthermore, in most cases, it was also able to identify the lowest number of IDs, an indicator that it will not be encountering a high number of false positives since its IDF1 remains high even in those cases. On the other hand, also consistently, JDE presented itself as the second-best option in most datasets, leaving Tracktor as the method with the most damaging results in this comparison.

Looking in more detail at the table above, all the results seem consistent and make sense.

However, in MOT20, for the JDE, the number of IDs seems strangely high compared to the others, which makes us repeat the finetuning on this dataset several times, increasing and decreasing the number of training epochs accordingly not leading the result to change significantly. Therefore, we think that the model failed to adjust correctly in this case, which is also justified by the low value of IDF1, indicating that the false detections rose considerably in this case. The values presented for the JDE were the result of a 30 epoch finetuning. However, 15 epochs increased the number of IDs to 33692 and simultaneously lowered the MOTA to 36.1. In contrast, the 45 epoch training led to a decrease in IDs to values closer to those expected. Despite that, we had an astronomical amount of false negatives, leading to a decrease in the IDF1 to 0.114, which lead us to consider the result in the table as the best result.

In order to also consider the case where we use single-object tracking, we group the results in table 4.5, which can be seen below. We use different configurations that are represented in the table below by SiamRPN_alex_dxcorr [LYW+18], SiamRPN_r50_l234_dwxcorr [LYW+18],SiamRPN_mobilev2_l234_dwxcorr [LYW+18] and SiamMask_r50_l3 [WZB+19]. The nomenclature used to define the weights used follows the sequence starting with the name of the method (SiamRPN or SiamMask), followed by the backbone used (AlexNet, ResNet50 or Mobilenetv2). Finally, it is also indicated if depth-wise cross-correlation (dwxcorr) is used. It also important to note that all the mentioned weights are trained on VID, YoutubeBB, COCO and ImageNetDet datasets and after that they were finetuned on VOT16, VOT18 and VOT19.

Table 4.5: Performance summary of current single-object tracking methods on VOT16, VOT18 and VOT 19.

	Method	Accuracy ↑	Robustness ↓	EAO ↑
VOT16	SiamRPN_alex_dxcorr	0.618	0.238	0.393
	SiamRPN_r50_dwxcorr	**0.643**	**0.200**	**0.461**
	SiamRPN_mobilev2_dwxcorr	0.624	0.210	0.454
	SiamMask_r50	0.621	0.210	0.437
VOT18	SiamRPN_alex_dxcorr	0.576	0.300	0.352
	SiamRPN_r50_dwxcorr	**0.602**	**0.239**	**0.413**
	SiamRPN_mobilev2_dwxcorr	0.585	0.234	0.411
	SiamMask_r50	0.598	0.243	0.407
VOT19	**SiamRPN_alex_dxcorr**	**0.643**	**0.200**	**0.461**
	SiamRPN_r50_dwxcorr	0.596	0.472	0.287
	SiamRPN_mobilev2_dwxcorr	0.579	0.451	0.292
	SiamMask_r50	0.597	0.461	0.283

As we can see from the table above, the SiamRPN method always had better results in all datasets when compared to SiamMask, and in two of the three datasets, the configuration using Resnet50 as the backbone was the one that had better results. It is also relevant to note that using the AlexNet backbone seems to be the worst solution in two of the three datasets, but in VOT19, it was the configuration that obtained the best results, and in this particular case, with considerable distance compared with the competing methods.

In summary, as far as the people-tracking task is concerned, if one opts for single-tracking methods, it seems better to use the SiamRPN method instead of SiamMask, and as far as the former is concerned, the best configuration seems to be the one that uses Resnet as a backbone. On the other hand, if one opts for multi-tracking methods, the best solution is to use FairMot, the latter having much better accuracy than the others.

4.4 Facial Recognition

Given the circumstances, facial recognition became a more difficult task to perform in the limited time of this paper. However, we thought it would be essential to add a section comparing results on standard datasets with a method, Adaptive Threshold for Face Recognition and Authentication, that uses a fixed threshold and the possibility of choosing it adaptively. So, to perform this comparison, we used the LFW and COLOR Feret datasets and tried to understand how the accuracy value behaved given the number of comparisons with the database using fixed thresholds and the adaptative one.

In Figure 4.2, we can see a graph representing the evolution of accuracy in face recognition taking into account the number of comparisons with the database., i.e., the maximum number of times the system can compare the similarity value calculated for the current subject with those already in the database. In the specific case of this method, each time a subject is stored in the database, its previously calculated feature vector is recorded. Then, we assign a different threshold to each subject, according to the calculated similarity. By doing this, in the end, when the system is confronted with a new image, it will extract its feature vector and calculate the similarity between it and all the subjects present in the database, thus identifying which subject is in the image, as long as it is present in the database, choosing the necessary threshold for each subject instead of having a global one.

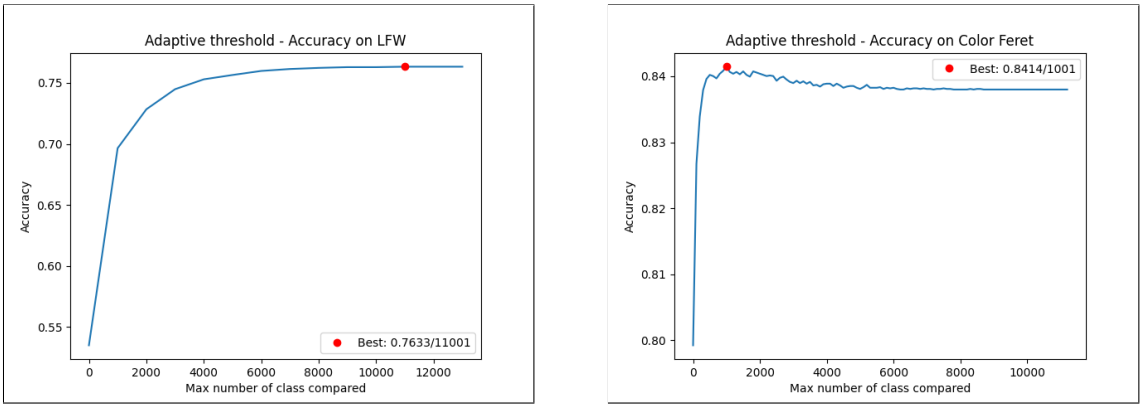

Figure 4.2: Comparison between accuracy and the number of comparisons with the database on LFW and COLOR FERET datasets.

Analysing the results using the adaptive threshold technique, the first conclusion we can draw is that the best accuracy result occurs when we compare 11001 different subjects in the database on LFW or 10001 on COLOR FERET, having at this stage practically a

comparison with all the elements therein. Naturally, when we increase the number of comparisons, the response time also increases, so depending on the system's purpose, it may be important not to have too high a response time. Although the best result occurs with 11001 comparisons with the database on LFW and 10001 comparisons with the database on COLOR FERET , we realise that after a certain point, even increasing the number of comparisons, the accuracy value does not change very significantly, so in the cases previously mentioned, we can choose to make fewer comparisons, even if slightly compromising the accuracy value. In the graph corresponding to Color FERET, we can also see a slight decrease in accuracy when we increase the number of comparisons with the database images. However, this fact is probably related to a smaller number of images for each class, causing that the number of comparisons does not increase the variability of the compared images, resulting in a worse performance.

From the previous explanation, we understand that it is possible to have an accuracy of around 76% or 84%, and it is essential to note that the value will not exclude any entity since each one will have its threshold. In order to compare this technique and understand if it is effectively a more robust solution than a global threshold, it is also important to understand the results with this solution and how the variation of a global threshold influences the results that are expressed in tables 4.6 and 4.7.

Table 4.6: Relation between the number of comparisons with the database and accuracy, using a fixed threshold on LFW.

Fixed Threshold	*Number of Comparisons*	*Accuracy*
0.25	1000	0.5322
	5000	0.5322
	7000	0.5322
	9000	0.5322
	13233	0.5322
0.3779	1000	0.5397
	5000	0.5397
	7000	0.5397
	9000	0.5397
	13233	0.5397
0.5	1000	0.6084
	5000	0.6084
	7000	0.6084
	9000	0.6084
	13233	0.6084
0.75	1000	0.8131
	5000	0.8131
	7000	0.8131
	9000	0.8131
	13233	0.8131

Table 4.7: Relation between the number of comparisons with the database and accuracy, using a fixed threshold on COLOR FERET

Fixed Threshold	*Number of Comparisons*	*Accuracy*
0.25	1000	0.7990
	5000	0.7990
	7000	0.7990
	9000	0.7990
	13233	0.7990
0.3968	1000	0.8072
	5000	0.8072
	7000	0.8072
	9000	0.8072
	13233	0.8072
0.5	1000	0.8298
	5000	0.8298
	7000	0.8298
	9000	0.8298
	13233	0.8298
0.75	1000	0.7988
	5000	0.7988
	7000	0.7988
	9000	0.7988
	13233	0.7988

Observing both the results obtained with the LFW dataset and with COLOR FERET, we understand that, contrary to what happened with the adaptive threshold technique, in this experiment, the accuracy is completely independent of the number of comparisons with the database, presenting the same accuracy result whether comparing 1000 subjects or the whole database. On the other hand, the fixed threshold chosen influences the accuracy results in both datasets, with accuracy increasing, in a generalized way, every time we increase the fixed threshold. In this case, however, it is essential to note that by increasing the threshold too much, valuing only the accuracy, we may be falling into an error and excluding the comparison with some entities, being, therefore, more cautious, in most systems, to choose values between 0.3 and 0.5 to avoid that there are people not being recognized. So, considering the above, the adaptive method seems to get better results in both datasets, being in the case of COLOR FERET even when compared to the higher threshold, and in the case of LFW, even though at the global level it does not have the best result, due to the fixed threshold of 0.75 it will be relevant to consider the adaptive one as well, so as not to ignore entities.

4.5 PAPSE Dataset Results

After comparing the different tasks presented, we thought it would also be pertinent to use our dataset and do some experiments with it. Since it is a very extensive dataset and fully annotated, we can train and test our results without worrying. In order to be able to do our experiments, we initially had to define our training and test sets. For this, using the advantage of our dataset being divided by sessions, we used the first four for training and the last two for testing.

The main goal of these experiments is to understand what will be the best solution, in our data, for the detection of people. The first consists of using the pre-trained weights of the YOLOv5X model, since they were the best in the comparative study of people detection in generic datasets, and from these perform a finetuning with our data. The second will consist of training a YOLOv5X model from scratch, not using any pre-trained weights, and comparing their performance, including some visual examples. We trained the first one for 40 epochs and the other with 300 epochs.

Table 4.8: Results from YOLOv5X finetuned on PAPSE-UBI.

Finetuned	$\mathbf{mAP_{50}} \uparrow$	$\mathbf{mAP} \uparrow$
Person	0.997	0.920
Face	0.979	0.616
Global	0.988	0.768

Table 4.9: Results from YOLOv5X from scratch.

From Scratch	$\mathbf{mAP_{50}} \uparrow$	$\mathbf{mAP} \uparrow$
Person	0.986	0.901
Face	0.975	0.592
Global	0.986	0.746

As we can see in tables 4.8 and 4.9, we present the global results obtained with our dataset and the specific result for each class, in this case, for Person and Face. The metrics used were the standard ones used in datasets that follow COCO style annotations, as is our case. Therefore, the first conclusion to be drawn is comparing the Face and Person classes which are, in both experiments, the ones that always present the worst mAP results. This fact is not surprising since it is also the one that has more variability and factors that may contribute to its difficulty in detection, since in the case of the person, in much of the dataset, people are seated in their respective places.

The results also seem to suggest that finetuning the YOLOv5X model with our data is preferable to training it from scratch, since the finetuned method showed better overall upper mAP results, as well as for both classes individually, reaching results quite close to 99% if an IoU threshold of 0.5 is used. It is also true that the from scratch result presents

high results, however, given the time it takes to train when compared to finetuned (about 146h against 19h), and since the latter presents superior results, it seems to be the best way to go in solving our problem.

In figure 4.3 we can see the inference of the same frame using the finetuned (left) and from scratch (right) models and a detection threshold of 0.5.

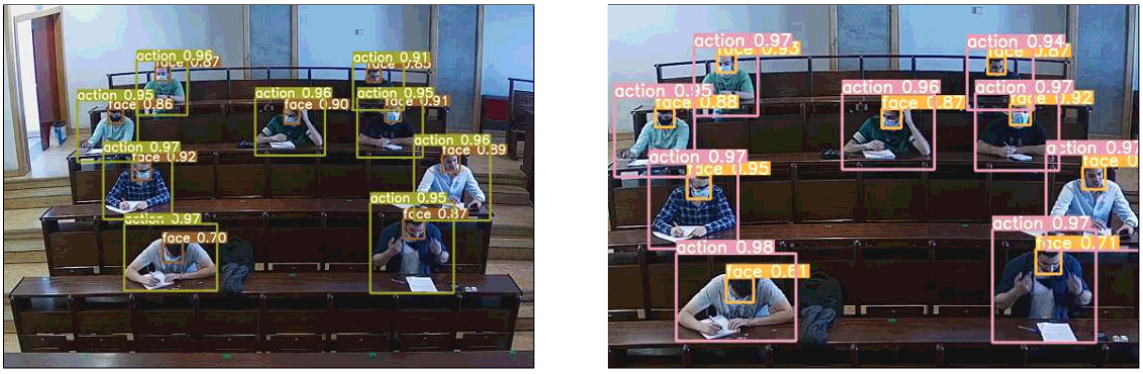

Figure 4.3: Inference from the finetuned model, on the left, and the one that was trained from scratch, on the right.

Looking at the images above, we see that the slight difference that seems to result from evaluating the two models also seems to be confirmed by a qualitative analysis of the data since all persons and faces were correctly detected in both cases. If for people, this is not surprising concerning faces, it can be suggested that in many cases, the lower value of the mAP will be the result not of a non-detection of the face but a non-perfect overlap with the annotation. Note that even in one of the subjects who are removing the mask, the models could identify his face correctly.

Despite the above, as in any system, even if it works perfectly, there will always be some flaws. In the specific case of inference in our dataset, the most common detection errors occur mainly when some hands or objects partially cover the face and, in some of these cases, the model cannot detect the face. However, the most obvious error in the whole dataset is the one found in figure 4.4 and occurs both in the finetuned and the trained from scratch method.

As we can see in the figure below, particularly in the first row containing students, the system misidentified a backpack as a subject. This error is relatively recurrent in the dataset, occurring in several frames, regardless of the method. Despite being identified as such, compared to the certainty, the model gives to the other cases, it is pretty tiny. Since it happens persistently and is likely related to the annotations we made, since some actions, namely packing or unpacking, the subject is annotated with the backpack, a fact that may be confusing the model.

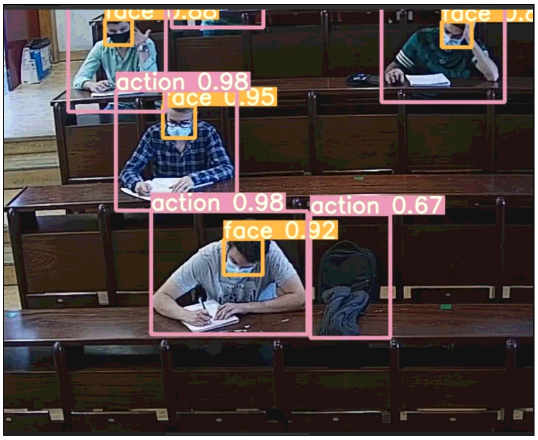

Figure 4.4: False detection of a subject.

In figure 4.5, we can observe two inferences in the same image, where on the left is the output of the trained model from scratch, while on the right is the output of the model that was finetuned.

Figure 4.5: Inference mistake from the sratch model, on the left, and the one that was finetuned, on the right.

As we can see in the image above, only the finetuned method was able to detect the first input of the subject, showing the slight improvement in the performance of the finetuned model compared to the other in these small details in the frame-by-frame detections. All in all, except for these minor errors, the system seems to be able to correctly and consistently detect both people and faces, and it is preferable to bet on a finetuned model of YOLOv5X rather than training from scratch, even if both give very high results.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This document's primary goal was to present the problem that we want to solve and how we think that we can address it to solve it. We need to find a way to register attendance in a classroom automatically. With this in mind, we believe that it is essential to understand three different problems fully: how can we detect an object (in particular a face) correctly; how can we keep track of an object if the system misses its place for a moment; and finally how can we recognize the face of someone. After this thinking, we analyze distinct types of methods.

After we perform a comparison by reviewing literature between ML methods and Deep Learning-based ones, in object detection, we find that the first group is no longer state-of-the-art and that the second group completely outperforms them. This happens because even if a classical method finds an optimal solution, Deep Learning will be able to find it and simultaneously classify whether in the first group we have two different stages. In the case of Deep Learning methods, we can say that the most recent methods (such as YOLO or SSD) outperform others in terms of speed, but sometimes their accuracy is not as good. In terms of tracking, we find that there are many possibilities to distinguish between types of methods, but we understood that the one that suits best our project is the distinction between single and multiple objects since all of our analysis is going to be produced offline. Likely, we will also use detection based tracking methods in order to speed our annotation process.

We installed our system in the 6.03 classroom because it was one of the available classrooms with conditions to install a camera without a significant perturbation and also because of its size, which allows us to respect all the privacy restrictions and at the same time record useful data. We also believe that our option of not recording sound was correct because it will not add essential features to us and, because of that, was an unnecessary, intrusive record.

Regarding the sessions that we have recorded, we can say that everything went as expected with the only major issue being the impossibility of recording more sessions with more volunteers due to the pandemic restrictions. Another minor issue related to these sessions is the loss of spontaneous behavior in a regular classroom since we are giving pre-defined roles and profiles to the volunteers in our sessions. We believe that including a real class in the dataset helped increase the dataset's spontaneous behaviour, although

we do not consider that there was too significant a loss in the simulated sessions. The creation and frame-by-frame annotation of the 40000 dataset images was undoubtedly a significant contribution of this dissertation towards solving the initial problem of automatically registering students in the classroom. Particular emphasis is given to the entire annotation process, which is long and quite time consuming, but it is essential to be able to make the necessary comparisons, both those already included in this dissertation and those that will be made in the future.

After the annotation process, we evaluated several methods to understand which would be the best to detect people, track them, and perform facial recognition. The results of our analysis unequivocally indicated that YOLOv5X would be the one most likely to do a good job since it was the most competitive in two distinct datasets (COCO and PASCAL VOC). Once this task was finished, we evaluated the single and multi-tracking methods, and it was clear that FairMot was the one whose results, in multi-tracking, stood out the most, while in single-tracking, two SiamRPN configurations stood out. Finally, concerning face recognition, it is impossible to state with certainty that the method used will be the best, but its accuracy results are pretty promising. We also performed two experiments on our dataset to finish our work, one using finetuning and the other training from scratch. The conclusion drawn was that for our problem, it is preferable to use finetuning.
After the conclusion of this work, we consider that the most significant contribution presented was the creation and annotation of the dataset, but also the work done in identifying the best state-of-the-art methods, as well as solving the problem of detecting people in our dataset should not be neglected. Nevertheless, there is room for improvement, and these improvements will be discussed in the next section.

5.2 Future Work

After the work developed in this dissertation, which answered several of the questions and tasks proposed, it is also essential to understand the following steps to improve the project. Our work would be essential to expand the experiments performed on our dataset, thus evaluating the tracking and facial recognition methods on our data, allowing us to give a more definitive answer on these two problems.

In addition, it would also be of great importance, should pandemic conditions allow it, to collect and annotate more data, this time with a more significant absence of masks, which would enable the use of facial recognition more assertively, also using actual classes and preferably using fixed groups of students so that comparisons are fairer. The model can be improved, thus building a database of subjects with more defined lines.

Since our data is already prepared for this, it would be interesting to study action recognition methods and thus take the necessary step towards evaluating student behaviour. Last but not least, getting all the responses to these tasks working as a whole, effectively

having a non-cooperative attendance recording system.

Bibliography

[Ann11] Anne-Mari Nevala, Jo Hawley, Dermot Stokes, Katie Slater, Manuel Souto Otero, Ruth Santos, Claire Duchemin and Anna Manoudi. Reducing early school leaving in the eu [online]. 2011. Available from: https://www.europarl.europa.eu/RegData/etudes/etudes/join/2011/460048/IPOL-CULT_ET(2011)460048(SUM01)_EN.pdf [cited november 2020]. 2

[Ass12] Association for Computing Machinery. Acm computing classification system [online]. 2012. Available from: https://dl.acm.org/ccs [cited november 2020]. 2

[BMLT19] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles, 2019. 24, 58

[CHP+20] Weijun Chen, Hongbo Huang, Shuai Peng, Changsheng Zhou, and Cuiping Zhang. Yolo-face: a real-time face detector. *The Visual Computer*, 03 2020. 27

[DLHS] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. 11

[DRM+20] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes, 2020. xvii, 36, 58

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005. 10

[EEVG+15] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, January 2015. xvii, 33, 54

[Gab19] Gabinete de Qualidade, UBI. Abandono escolar e retenção de estudantes [online]. 2019. Available from: https://www.ubi.pt/entidade/Editar-Sigla-03-01-2020-14_47_16 [cited november 2020]. 2

[GDDM] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. 12

[Gir15] Ross Girshick. Fast r-cnn, 2015. 13

[HGDG18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018. 15, 54

[HMBLM08] Gary Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. *Tech. rep.*, 10 2008. 37

[JLM10] Vidit Jain and Erik Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst, 2010. 38

[JSB+21] Glenn Jocher, Alex Stoken, Jirka Borovec, NanoCode012, Ayush Chaurasia, TaoXie, Liu Changyu, Abhiram V, Laughing, tkianai, yxNONG, Adam Hogan, lorenzomammana, AlexWang1900, Jan Hajek, Laurentiu Diaconu, Marc, Yonghye Kwon, oleg, wanghaoyang0106, Yann Defretin, Aditya Lohia, ml5ah, Ben Milanko, Benjamin Fineran, Daniel Khromov, Ding Yiwei, Doug, Durgesh, and Francisco Ingham. ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations, April 2021. Available from: `https://doi.org/10.5281/zenodo.4679653`. 54

[KML+16] Matej Kristan, Jiri Matas, Aleš Leonardis, Tomas Vojir, Roman Pflugfelder, Gustavo Fernandez, Georg Nebehay, Fatih Porikli, and Luka Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2137–2155, Nov 2016. 33

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012. Available from: `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`. 22

[KY55] H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955. 24

[LAE+16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, page 21–37, 2016. Available from: `http://dx.doi.org/10.1007/978-3-319-46448-0_2`. 18, 54

[LLJD15] Q. Li, R. Li, K. Ji, and W. Dai. Kalman filter and its application. In *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pages 74–77, 2015. 24

[LMB+15] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick,

and Piotr Dollár. Microsoft coco: Common objects in context, 2015. xvii, 32, 54

[Low99] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. 8, 9

[LTMR+15] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking, 2015. 36, 58

[LWW+18] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks, 2018. 22

[LWW+19] Jian Li, Yabiao Wang, Changan Wang, Ying Tai, Jianjun Qian, Jian Yang, Chengjie Wang, Jilin Li, and Feiyue Huang. Dsfd: Dual shot face detector, 2019. 28

[LYW+18] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018. 59

[Min19] Minghao Ning. Sift(scale-invariant feature transform) [online]. 2019. Available from: `https://towardsdatascience.com/sift-scale-invariant-feature-transform-c7233dc60f37` [cited november 2020]. xv, 9

[MLTR+16] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking, 2016. xvii, 36, 58

[RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016. 17, 54

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *In NIPS,*, 2015. 14

[Sat16] Satya Mallick. Histogram of oriented gradients [online]. 2016. Available from: `https://learnopencv.com/histogram-of-oriented-gradients/` [cited november 2020]. xv, 10

[SBMT20] Bing Shuai, Andrew G. Berneshawi, Davide Modolo, and Joseph Tighe. Multi-object tracking with siamese track-rcnn, 2020. 24

[SLJ+14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 17

[Soc20] Socret Lee. Understanding face detection with the viola-jones object detection framework, 2020. Available from: https://towardsdatascience.com/understanding-face-detection-with-the-viola-jones-object-detection/-framework-c55cc2a9da14 [cited november 2020]. xv, 7

[TPL20] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection, 2020. 54

[VJ01] Paul Viola and Michael Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001. 6, 8

[WBP17] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric, 2017. 24

[WLY15] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37:1–1, 09 2015. 35

[WZB+19] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip H. S. Torr. Fast online object tracking and segmentation: A unifying approach, 2019. 22, 59

[WZL+20] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking, 2020. 22, 58

[YJS06] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13–es, December 2006. Available from: https://doi.org/10.1145/1177352.1177355. 20

[YK19] S. Yun and S. Kim. Recurrent yolo and lstm-based ir single pedestrian tracking. In *2019 19th International Conference on Control, Automation and Systems (ICCAS)*, pages 94–96, 2019. 21

[YLLT16] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 37

[ZWW+20] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking, 2020. 58