



Sistema de Vigilância Inteligente para Ambientes Domésticos com Visão Computacional

Miguel Ferreira Costa
Aluno nº: 1210064

**Dissertação para obtenção do Grau de
Mestre em Engenharia de Inteligência Artificial**

**Orientador: Doutor António Constantino Lopes Martins, Professor Adjunto do
Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto**

Júri:

Presidente:

Doutor Paulo Sérgio dos Santos Matos, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Vogais:

Doutor Joaquim Filipe Peixoto dos Santos, Professor Ajunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Orientador:

Doutor António Constantino Lopes Martins, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Porto, 15 Outubro de 2024

Dedicatória

Ao meu pai e namorada por todo o apoio ao longo da minha vida académica, especialmente neste último ano.

A ti, mãe, que não pudeste estar presente no fim desta jornada, mas cuja força me acompanhou em cada passo. Sei que, onde quer que estejas, continuas a olhar por mim com orgulho.

Resumo

Nos dias de hoje, num mundo onde a segurança privada é cada vez mais crucial, a Inteligência Artificial emerge como uma aliada poderosa para proteger os nossos lares. No entanto, os sistemas de vigilância para ambientes domésticos ainda enfrentam desafios significativos, nomeadamente, na falta de personalização e na dependência de tecnologias tradicionais, que não conseguem atender às exigências modernas de segurança.

Esta dissertação apresenta o sistema iDetect, um sistema de vigilância inteligente com deteção e reconhecimento facial voltado para ambientes domésticos. O sistema utiliza uma combinação de tecnologias tradicionais e deep learning, alcançando uma precisão de 93% na deteção e uma acurácia de 97% para o reconhecimento facial, com base numa seleção criteriosa de algoritmos, utilizando uma versão adaptada da metodologia PRISMA. Devido à crescente utilização de smartphones, o sistema possui a sua aplicação móvel, totalmente configurável e intuitiva, que permite ao utilizador personalizar o sistema e receber notificações e alertas em tempo real.

Os resultados mostram que o sistema implementado oferece uma solução eficaz e acessível, que une o poder da Inteligência Artificial à portabilidade dos dispositivos móveis, proporcionando uma resposta mais rápida e eficiente às ameaças domésticas. Foi possível implementar um sistema que supera os atuais sistemas de videovigilância doméstica, fornecendo respostas mais ágeis e precisas, reduzindo significativamente o número de falsos positivos. O sistema desenvolvido representa um avanço no campo da segurança doméstica, ao disponibilizar uma solução inovadora e de fácil utilização que pode ser adaptada às necessidades individuais de cada utilizador.

Palavras-chave: Sistema de Vigilância, Deteção Facial, Reconhecimento Facial, Deep Learning, CNN, YOLOv8, Faster RCNN, Triplet Loss, Softmax.

Abstract

Nowadays, in a world where private security is increasingly crucial, Artificial Intelligence emerges as a powerful ally to protect our homes. However, surveillance systems for domestic environments still face significant challenges, particularly the lack of personalization and reliance on traditional technologies that fail to meet modern security demands.

This dissertation presents iDetect, an intelligent surveillance system with facial detection and recognition tailored for domestic environments. The system uses a combination of traditional technologies and deep learning, achieving up to 93% precision in detection and 97% accuracy in facial recognition, based on a careful selection of algorithms and utilizing an adapted version of the PRISMA methodology. Given the growing use of smartphones, the system includes a fully customizable and intuitive mobile application, allowing the user to personalize the system and receive real-time notifications and alerts.

The results show that the implemented system offers an effective and accessible solution, combining the power of Artificial Intelligence with the portability of mobile devices, providing a faster and more efficient response to domestic threats. It was possible to implement a system that surpasses current home surveillance systems, providing more agile and precise responses, significantly reducing the number of false positives. The developed system represents an advancement in the field of home security by offering an innovative and easy-to-use solution that can be adapted to the individual needs of each user.

Keywords: Sistema de Vigilância, Detecção Facial, Reconhecimento Facial, Deep Learning, CNN, YOLOv8, Faster RCNN, Triplet Loss, Softmax.

Agradecimentos

Os agradecimentos são uma extensão da dedicatória. Agradeço-vos: mãe, pai e namorada.

Deixo, ainda, uma palavra ao meu orientador: Professor Constantino Martins, pela partilha de conhecimento e pela ajuda durante a realização desta dissertação.

Índice

Lista de Figuras	xi
Lista de Tabelas.....	xiii
Lista de trechos de código	xiv
Acrónimos e Símbolos.....	xv
1 Introdução	1
1.1 Contexto	1
1.2 Problema.....	3
1.3 Objetivos.....	4
1.3.1 Objetivos Específicos	5
1.4 Motivação.....	6
1.5 Contributos	7
1.6 Estrutura do documento.....	8
2 Estado de arte e fundamentação teórica	9
2.1 Contextualização teórica.....	9
2.1.1 Inteligência Artificial	9
2.1.2 Machine Learning	10
2.1.3 <i>Deep learning</i> e Visão Computacional.....	10
2.1.3.1 Redes Neurais.....	11
2.1.4 Redes Neurais Convulsioneiras (CNN)	12
2.1.5 Aprendizagem por transferência	13
2.1.6 Reconhecimento Facial.....	13
2.1.7 Detecção Facial	15
2.1.7.1 Viola Jones (Haar Cascade).....	15
2.1.7.2 Histograma de Gradientes Orientados (HOG)	16
2.1.7.3 Single Shot Detection (SSD).....	17
2.1.7.4 Multi-Task Cascade Convolutional Neural Network (MTCNN)	18
2.1.7.5 YOLO (You Only Look Once)	19
2.1.7.6 RCNN, Faster RCNN e Masked RCNN.....	20
2.1.8 Extração de características.	22
2.1.8.1 Principal Component Analysis (PCA) e Linear Discriminant Analysis (LDA) ..	22
2.1.8.2 EigenFaces	23

2.1.8.3	FisherFaces.....	24
2.1.8.4	Principais Modelos Existentes.....	24
2.1.9	Métodos de Classificação.....	28
2.1.9.1	K-nearest Neighbour (KNN).....	28
2.1.9.2	Support vector machine (SVM)	29
2.2	Revisão Sistemática da Literatura	31
2.2.1	Questões de pesquisa	31
2.2.2	Fontes de informação.....	32
2.2.3	Termos de pesquisa	32
2.2.4	Extração de informação	33
2.2.5	Resultados	36
2.2.5.1	Quais são os algoritmos/técnicas necessários para implementar técnicas de reconhecimento facial?	36
2.2.5.2	O quão precisos são os algoritmos/técnicas de reconhecimento facial em situações desfavoráveis?	37
2.2.5.3	Que mecanismos/soluções de vigilância com recurso a reconhecimento facial semelhantes à solução proposta existem?	37
2.2.6	Discussão.....	38
2.2.6.1	Quais são os algoritmos/técnicas necessários para implementar técnicas de reconhecimento facial?	38
2.2.6.2	O quão precisos são os algoritmos/técnicas de reconhecimento facial em situações desfavoráveis?	44
2.2.6.3	Que mecanismos/soluções de vigilância com recurso a reconhecimento facial semelhantes à solução proposta existem?	48
2.3	Privacidade e proteção dos dados	51
2.4	Algumas aplicações de videovigilância.....	52
2.5	Sumário	53
3	Método e Implementação	55
3.1	Método	55
3.2	Arquitetura da solução	61
3.3	Implementação da solução.....	62
3.3.1	Módulo de Machine Learning.....	63
3.3.1.1	Datasets.....	63
3.3.1.2	Pré-processamento de dados	66
3.3.1.3	Modelos de deteção facial	70
3.3.1.4	Modelos de Reconhecimento Facial	72
3.3.2	Módulo de Notificações.....	77
3.3.3	Módulo dos Serviços de Backend	78

3.3.3.1	Serviço de informação do utilizador e base de dados.....	78
3.3.3.2	Serviço de Machine Learning	80
3.3.4	Dispositivos externos.....	82
3.3.5	Interface do utilizador.....	83
3.4	Sumário	87
4	Avaliação e Experimentação.....	88
4.1	Metodologia	88
4.2	Deteção Facial	89
4.2.1	Validação e experimentação dos modelos de deteção.	90
4.3	Reconhecimento Facial.....	92
4.3.1	Validação dos modelos de reconhecimento	92
4.4	Sumário	95
5	Conclusão	97
5.1	Objetivos alcançados	97
5.1.1	Objetivo 1 - Estado de Arte.....	98
5.1.2	Objetivo 2 e 3 - Datasets.....	98
5.1.3	Objetivo 4 e 5 - Modelos de deteção e reconhecimento	98
5.1.4	Objetivo 6 - Sistema de Vídeo.....	99
5.1.5	Objetivos 7, 8, 9 e 10 - Desenvolvimento do sistema de vigilância	99
5.2	Melhorias e Trabalhos futuros	100
5.3	Considerações finais	101
6	Referências.....	103

Lista de Figuras

Figura 1 - Típico sistema de vídeovigilância [6].	2
Figura 2 - Projeção do mercado de vídeovigilância [7].	2
Figura 3 - Arquitetura de um neurónio (esquerda) e arquitetura de geral de uma rede neuronal (direita) [24].	11
Figura 4 - Camadas de uma CNN [25].	12
Figura 5 - Etapas do reconhecimento facial [32].	14
Figura 6 - Classificador Haar [33].	16
Figura 7 – (a) Imagem original, (b) Gradiente Horizontal, (c) Gradiente Vertical, (d) Gradiente Horizontal e Vertical [42].	16
Figura 8 – (a) Divisão da imagem em células, (b) Histograma de cada célula [42].	17
Figura 9 – (a) Imagem original, (b) Características HOG [42].	17
Figura 10 - Estrutura SDD baseada no modelo ResNet50 [43].	18
Figura 11 - Arquitetura MTCNN [46].	19
Figura 12 - Arquitetura YOLO [35].	20
Figura 13 - Funcionamento do modelo RCNN [38].	21
Figura 14 - Funcionamento Faster RCNN [37].	21
Figura 15 - Funcionamento do modelo Mask RCNN [50].	22
Figura 16 - Imagens Originais (esquerda) e EigenFaces (direita) [55].	23
Figura 17 – Imagem Original (esquerda) e FisherFace (direita) [56].	24
Figura 18 – Treino de <i>triplets</i> [66].	26
Figura 19 – Arquitetura do modelo DeepFace [67].	26
Figura 20 – Funcionamento do KNN [72].	29
Figura 21 - Tipos de separação [74].	30
Figura 22 – Diagrama Prisma.	35
Figura 23 – Comparação entre algoritmos de detecção facial [82].	38
Figura 24 - Comparação entre os métodos de classificação em [83].	40
Figura 25 – Performance dos algoritmos com uma imagem de input [87].	41
Figura 26 – Performance dos algoritmos com imagem em tempo real [87].	41
Figura 27 – Comparação da solução com modelos existentes em [95].	42

Figura 28 - Performance das abordagens em [97].....	43
Figura 29 – Comparação de soluções em [98].....	43
Figura 30 - Resultados dos métodos tradicionais em[102].....	44
Figura 31 - Resultado dos métodos de deep learning em [102]	45
Figura 32 - Performance dos modelos em [108].	46
Figura 33 – Resumo do estudo do estado de arte.	53
Figura 34 - Fluxograma da solução.	57
Figura 35 - Arquitetura do Sistema.....	62
Figura 36 - Foto com 1 rosto (esquerda) e foto com 3 rostos (direita).....	64
Figura 37 - Exemplo de caixa delimitadoras.	64
Figura 38 – Dataset privado de reconhecimento facial.	65
Figura 39 –Imagem original e variações aumentadas para detecção.	66
Figura 40 - <i>Bounding Boxes</i> normalizadas.	67
Figura 41 - Resultado da junção da imagem com as respectivas <i>bboxes</i>	67
Figura 42 - Pipeline de pré-processamento do dataset detecção.	68
Figura 43 - Versões aumentadas no <i>dataset</i> de reconhecimento privado.	68
Figura 44 - Pipeline de pré-processamento do <i>dataset</i> de reconhecimento privado.	69
Figura 45 - Pipeline de pré-processamento do <i>dataset</i> de reconhecimento publico.....	69
Figura 46 - Arquitetura dos modelos de detecção.	70
Figura 47 – <i>Screenshot</i> de um conjunto de <i>triplets</i>	73
Figura 48 - Arquitetura do modelo <i>Triplet Loss</i>	74
Figura 49 - <i>Embeddings</i> gerados pelo <i>dataset</i> privado com 4 elementos.	75
Figura 50 - Arquitetura modelo <i>Softmax</i>	76
Figura 51 - <i>Endpoints</i> do serviço de informação do utilizador.....	78
Figura 52 - Diagrama de sequência do processo de notificação.....	80
Figura 53 - Diagrama de sequência dos comandos remotos do sistema.....	81
Figura 54 - Câmara de videovigilância com recurso à Raspberry Pi.....	83
Figura 55 - Configuração do sistema.	84
Figura 56 - Configurações avançadas do sistema.	84
Figura 57 - Sistema de vídeo em direto.	86
Figura 58 - Notificação de alerta e <i>dashboard</i>	87
Figura 59 - Testes dos modelos de detecção facial.	90
Figura 60 - Testes dos modelos de reconhecimento facial.....	93
Figura 61 - Repositórios do GitHub.....	111
Figura 62 - Pastas do serviço de dados do utilizador.....	111
Figura 63 - Pastas da aplicação movel.	112

Lista de Tabelas

Tabela 1 - Performance dos modelos no dataset LFW.	28
Tabela 2 — Questões de pesquisa.....	31
Tabela 3 — Fontes de informação.....	32
Tabela 4 — Domínios e <i>keywords</i> usadas na seleção dos termos de pesquisa.....	32
Tabela 5 — Query Strings.....	33
Tabela 6 — Critérios de Inclusão	34
Tabela 7 — Critérios de Exclusão	34
Tabela 8 — Resultados por fonte de informação.....	34
Tabela 9 - Modelos de deep learning abordados nos artigos.....	36
Tabela 10 - Tempo de treino dos modelos de detecção.	72
Tabela 11 - Tempo de treino e retreino dos modelos de reconhecimento.....	77
Tabela 12 - Performance dos modelos de detecção.	90
Tabela 13 - Performance dos modelos de reconhecimento.....	93

Lista de trechos de código

Trecho de código 1 - Endpoints de configuração dos modelos.	113
Trecho de código 2 - <i>Request</i> ao Serviço de Notificações.	114
Trecho de código 3 - <i>Request</i> para guardar alerta.....	114
Trecho de código 4 - Funções para retornar dados do utilizador.....	114
Trecho de código 5 - Classe <i>VideoStreamer</i> responsavel por aplicar os modelos em tempo real.	116
Trecho de código 6 - <i>Widget</i> responsavel por criar o ecrã do vídeo em direto.....	117
Trecho de código 7 - <i>Dataloader</i> para treinar o modelo Faster RCNN.	118
Trecho de código 8 - Classe para geração de tripletos de treino e validação.....	120
Trecho de código 9 - Lógica para criar uma pessoa autenticada no sistema de um utilizador.	120

Acrónimos e Símbolos

PRISMA	<i>Preferred Reporting Items for Systematic Reviews and Meta-Analyses</i>
IOT	<i>Internet Of Things</i>
IA	<i>Inteligência Artificial</i>
ML	<i>Machine Learning</i>
CI	<i>Critério de Inclusão</i>
CE	<i>Critério de Exclusão</i>
RF	<i>Reconhecimento facial</i>
SVM	<i>Support Vector Machine</i>
PCA	<i>Principal Component Analysis</i>
LSA	<i>Linear Discriminant Analysis</i>
KNN	<i>K-nearest Neighbour</i>
CNN	<i>Convolutional neural network</i>
MTCNN	<i>Multi-Task Cascade Neural Network</i>
HOG	<i>Histogram of Oriented Gradients</i>
LBPH	<i>Local Binary Patterns Histogram</i>
LBP	<i>Local Binary Pattern</i>
SVD	<i>Singular Value Decomposition</i>
SSD	<i>Single Shot Detection</i>
MMOD	<i>Max-marging Object Detection</i>
DBN	<i>Deep Belief Network</i>
BBOXS	<i>Bounding Boxes</i>
FPS	<i>Frames Per Second</i>

1 Introdução

Este capítulo inicial tem como objetivo fazer uma contextualização geral do problema, descrevendo também de forma detalhada todos os objetivos desta dissertação, assim como as principais motivações para a realização da mesma. Além do mais, expõe-se os principais contributos. No fim, é demonstrada a estrutura geral do documento.

1.1 Contexto

Nos últimos anos, os sistemas de vigilância tornaram-se ferramentas essenciais para garantir a segurança e a proteção dos nossos espaços e bens mais preciosos, seja em residências ou em ambientes comerciais [1].

Estes sistemas, normalmente, são compostos por câmaras que capturam imagens em tempo real, que podem ser gravadas e visualizadas posteriormente [1]. Entre os tipos mais comuns de sistemas de vigilância, destacam-se os CCTV's [2] que utilizam câmaras analógicas para a transmissão de vídeo, estando diretamente ligadas a um número limitado de monitores, e as câmaras de IP, que são dispositivos digitais que se conectam à internet, que permitem a visualização e gestão remota [2].

A evolução dos sistemas de vigilância foi significativamente impulsionada pelo avanço da internet e das redes sem fio, que transformaram a forma como a segurança é gerida [3]. Anteriormente, os sistemas de vigilância dependiam de infraestruturas complexas e limitadas, onde a monitorização só podia ser realizada em locais específicos, muitas vezes em salas de controlo dedicadas [4]. Com o avanço e utilização da internet, surgiu a possibilidade de realizar monitorização remota e em tempo real, de modo a que proprietários e gestores de segurança acedam às câmaras de vigilância a partir de qualquer dispositivo conectado, seja um computador, tablet ou smartphone [5].

Esta flexibilidade trouxe um controlo sem precedentes, onde os utilizadores podem não apenas visualizar as imagens em tempo real, mas também receber alertas instantâneos, ajustar configurações e até interagir com os sistemas de segurança à distância [4]. A integração com redes sem fio eliminou a necessidade de cablagem extensiva, reduzindo custos e facilitando a instalação, o que contribuiu para a popularização desses sistemas tanto em contextos residenciais como comerciais.

Nos contextos residenciais, os sistemas de vigilância são utilizados para monitorar entradas, saídas e áreas ao redor da propriedade, semelhante à Figura 1. A instalação de câmaras de segurança em casas tornou-se uma prática comum, de forma a oferecer aos proprietários uma camada adicional de proteção contra assaltos, invasões, furtos e roubos. Estes sistemas não só proporcionam um meio de dissuasão contra crimes, como também permitem a recolha de provas em caso de incidentes.



Figura 1 - Típico sistema de vídeovigilância [6].

De acordo com dados da Statista [7] o mercado global de videovigilância tem uma previsão de crescimento significativo, passando de US\$ 23,6 bilhões em 2019 para US\$ 62,4 bilhões em 2027, conforme ilustrado na Figura 2.

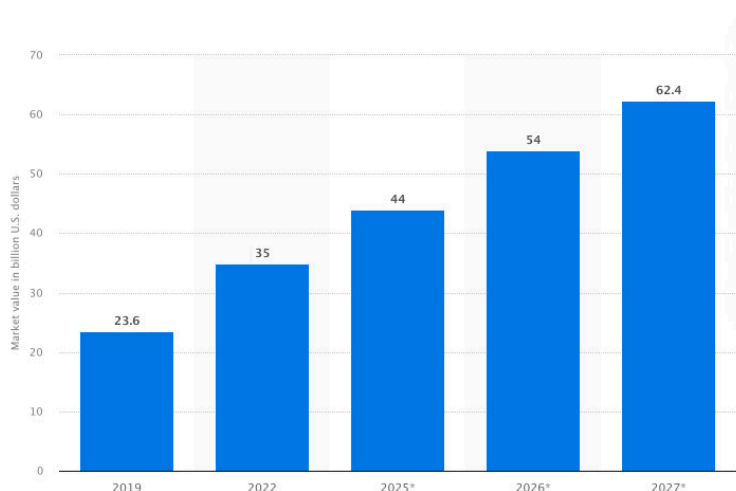


Figura 2 - Projeção do mercado de vídeovigilância [7].

Tendo em conta este facto, torna-se necessário que os investimentos continuem a ser direcionados para o desenvolvimento e a implementação de novos e mais poderosos sistemas de vigilância. A crescente preocupação com a segurança, aliada à necessidade de proteger bens e pessoas, exige que as soluções de vigilância se tornem cada vez mais acessíveis e eficazes. Este investimento, permite não apenas atender à procura crescente, mas também potencializar a capacidade de resposta a incidentes e melhorar a segurança em diversas áreas.

1.2 Problema

Os sistemas de vigilância tradicionais apresentam várias limitações significativas na identificação de intrusões em tempo real, principalmente devido à sua incapacidade de distinguir automaticamente entre situações normais e atividades suspeitas [8]. Esta limitação resulta frequentemente em respostas reativas, onde as imagens são analisadas apenas após a ocorrência de um incidente, muitas vezes tarde demais para prevenir danos ou perdas [9]. Além disso, os falsos alarmes são uma ocorrência comum, causados por eventos triviais como o movimento de animais ou condições climáticas, comprometendo a confiança no sistema [8].

Em Portugal, existem várias empresas que oferecem serviços de videovigilância, que também ainda não utilizam as mais recentes e poderosas técnicas de Inteligência Artificial. As câmaras disponibilizadas por estas empresas frequentemente contam apenas com sensores básicos de movimento, som ou luz, sem a integração de tecnologias avançadas de análise em tempo real. Entre os exemplos mais conhecidos em Portugal estão a Securitas Direct ¹ e Prosegur ², esta última que por sua vez, planeia investir em soluções de IA até 2030, com o objetivo de revolucionar as suas ofertas de segurança e videovigilância [10].

Uma possível solução para superar as limitações dos atuais sistemas de videovigilância seria a introdução de tecnologias de reconhecimento baseadas em dados biométricos, semelhantes aos que são utilizados nos atuais sistemas biométricos, nomeadamente, em lugares como aeroportos, bancos, áreas de pesquisa militar, áreas de segurança nacional e áreas de produção de empresas multinacionais [11]. Este tipo de tecnologia tem como objetivo autenticar indivíduos para fins de segurança com base num conjunto de características biométricas, nomeadamente o rosto, a íris, a impressão digital, entre outras [12]. Para o desenvolvimento destes sistemas a utilização de inteligência artificial é fundamental, através do processamento de imagem ou vídeo, com recurso a visão computacional [13].

No que diz respeito aos tipos de aplicações biométricas, o reconhecimento facial em particular, tem atraído muitos investigadores durante a última década, pois ao contrário dos outros tipos de biometrias, requer uma baixa cooperação por parte dos indivíduos [14]. Esta característica

¹ Alarmes para casas e empresas | Securitas Direct Portugal - <https://www.securitasdirect.pt/>

² Alarmes e Segurança para Casas e Empresas | Prosegur Portugal - <https://www.prosegur.pt/>

torna o reconhecimento facial particularmente valioso em cenários de vigilância, onde a identificação de indivíduos pode ser realizada de forma discreta e sem a necessidade de interação direta.

Sendo assim, a presente dissertação pretende abordar a lacuna existente na utilização de Inteligência Artificial nos sistemas de videovigilância, que embora seja uma tendência crescente, ainda não é uma prática bem consolidada nas grandes empresas de videovigilância.

1.3 Objetivos

O foco principal desta dissertação será o desenvolvimento de um sistema de vigilância totalmente inteligente capaz de identificar pessoas autorizadas ou desconhecidos num determinado ambiente através de técnicas avançadas de detecção e reconhecimento facial. O sistema também será capaz de enviar notificações e alertas ao utilizador do sistema, sempre que existir um comportamento suspeito. Ao integrar IA, o sistema proposto não apenas irá monitorizar e identificar indivíduos em tempo real, mas também aprenderá a diferenciar comportamentos normais de anômalos, reduzindo falsos alarmes e aumentando a eficiência da segurança. Desta forma, espera-se criar uma solução robusta que reforce significativamente a proteção dos espaços residenciais, oferecendo uma resposta eficaz às limitações dos sistemas atuais. Inicialmente, este sistema será projetado para um ambiente doméstico, mas terá a escalabilidade necessária para ser utilizado noutros cenários, por exemplo, num contexto empresarial, hospitais ou estabelecimentos públicos.

O sistema será integrado numa aplicação móvel totalmente personalizável, onde o utilizador poderá configurar diversos parâmetros do sistema. Para mais, a aplicação notificará o utilizador em tempo real sempre que um comportamento suspeito for detetado, proporcionando uma resposta imediata e eficiente.

Para alcançar estes objetivos, o trabalho focar-se-á primeiramente numa revisão da literatura aprofundada das técnicas de reconhecimento facial, incluindo a pesquisa e análise das técnicas mais avançadas e eficazes, como algoritmos de aprendizagem profunda e visão computacional. Será feita uma comparação de diferentes abordagens e identificadas as melhores práticas para implementação em ambientes de vigilância. Paralelamente, será desenvolvida uma aplicação móvel com uma interface intuitiva e amigável que seja de fácil configuração e gestão do sistema de vigilância. A aplicação integrará notificações em tempo real e, ainda, vídeo em direto, garantindo que o utilizador seja informado imediatamente sobre qualquer atividade suspeita.

Em resumo, este trabalho não só visa estudar e aplicar técnicas avançadas de reconhecimento facial, mas também desenvolver uma solução prática e eficaz que possa ser utilizada em diversos ambientes domésticos, aumentando a segurança e a tranquilidade dos seus utilizadores. Para tal, a investigação irá centrar-se na seguinte questão de investigação:

Como incorporar mecanismos de inteligência artificial para melhorar os tradicionais sistemas de videovigilância, de forma a criar um sistema de vigilância completamente inteligente?

O objetivo será, portanto, responder a esta pergunta, propondo uma solução inovadora que automatize e eleve a eficácia das tecnologias de vigilância, com recurso a algoritmos avançados de IA.

1.3.1 Objetivos Específicos

Para concretizar o objetivo geral deste trabalho, foram estabelecidos um conjunto de objetivos específicos que contribuirão para a solução final. Esses objetivos específicos são fundamentais para garantir que todas as etapas do projeto sejam abrangidas de forma clara e eficiente, permitindo o desenvolvimento de um sistema de vigilância inteligente com reconhecimento facial.

01. Investigar o presente estado de arte sobre as tecnologias de reconhecimento facial e entender o seu funcionamento.

02. Encontrar um/vários *datasets* de forma a alimentar o treino do modelo que irá fazer a deteção facial.

03. Criar um/vários *datasets* de rostos autorizados que irá ser utilizado para treinar os modelos de reconhecimento facial.

04. Implementar um ou mais modelos com recurso a inteligência artificial, capaz de realizar a operação de deteção facial em pessoas.

05. Implementar um ou mais modelos com recurso a inteligência artificial, capaz de fazer o reconhecimento facial de um indivíduo a partir de vídeo em direto.

06. Construção de um sistema que forneça vídeo em direto, com recurso a uma Raspberry Pi e uma câmara e no qual irão ser aplicados os diferentes modelos.

07. Criação de uma aplicação *mobile* que irá servir como interface do utilizador, onde o utilizador poderá efetuar diversas configurações no seu sistema, nomeadamente:

- **07.1.** Adicionar perfis de pessoas à sua *whitelist*, através do carregamento de fotos/vídeo.
- **07.2.** Visualizar em tempo real o estado do seu ambiente através de vídeo em direto e conseguir identificar que pessoas estão presentes no ambiente.
- **07.3.** Ativar ou desativar o sistema.
- **07.4.** Configurar os diferentes tipos de modelos para o seu sistema

08. Criação de um mecanismo de notificações *push* onde o utilizador é alertado quando o sistema deteta um comportamento suspeito:

- **08.1.** Notificar quando existe uma pessoa não familiar no ambiente.
- **08.2.** Notificar quando os modelos de reconhecimento facial estão prontos a ser utilizados após o treino.

O9. Construção de uma base de dados que irá armazenar as informações de um dado utilizador e as informações gerais do sistema.

O10. Para suportar todas estas necessidades, é necessário criar um ou mais serviços que irão interligar a solução, desde a aplicação *mobile*, o sistema de vídeo em direto e o sistema de notificações.

1.4 Motivação

A principal motivação para a realização deste trabalho foi o próprio desafio pessoal que estava agregado à sua execução, de forma a potencializar os conhecimentos adquiridos ao longo do mestrado. No decorrer deste, tive a oportunidade de explorar diversas áreas das tecnologias de informação, mas, desde o início, a visão computacional foi um campo que me fascinou profundamente. No entanto, até o momento, não tive a oportunidade de trabalhar diretamente com essa tecnologia. Desta forma, decidi que o trabalho final do mestrado seria a oportunidade ideal para mergulhar nessa área e aplicar os conhecimentos adquiridos num projeto prático. Além disso, a possibilidade de interligar a minha vida profissional com a vida académica foi uma das grandes motivações para a elaboração deste trabalho.

Como engenheiro de software focado no desenvolvimento de aplicações *mobile*, vejo uma grande oportunidade na convergência dessas duas áreas de conhecimento. O desenvolvimento de um sistema de vigilância inteligente com reconhecimento facial, pode beneficiar significativamente de técnicas avançadas de software e da crescente ubiquidade dos dispositivos móveis. Integrar essas duas vertentes permitirá criar soluções mais robustas e acessíveis, aumentando o impacto potencial do meu trabalho.

Os campos da visão computacional e do desenvolvimento de software para dispositivos móveis têm sido tendências nos últimos anos, impulsionados pela crescente procura por tecnologias inovadoras e eficientes. Trabalhar neste projeto não só contribui para minha formação académica, mas também representa um investimento significativo no meu futuro profissional. A experiência adquirida ao desenvolver este sistema não apenas ampliará a minha base de conhecimento, mas também proporcionará uma nova perspetiva sobre como as tecnologias emergentes podem ser integradas para resolver problemas do mundo real.

Em suma, este projeto é uma oportunidade para consolidar e expandir as minhas competências tanto na teoria quanto na prática. Representa um passo importante na minha trajetória como estudante e profissional, que me permite contribuir para a área de tecnologia da informação com uma solução inovadora e relevante. Ao enfrentar este desafio, espero não apenas crescer como engenheiro, mas também abrir novas possibilidades de carreira, de forma a explorar o vasto potencial da inteligência artificial e da visão computacional aplicada a sistemas de vigilância.

1.5 Contributos

Esta dissertação pretende realizar vários contributos significativos, sobretudo no domínio da Inteligência Artificial, com foco em subdomínios específicos que serão fundamentais para a implementar o sistema proposto. Os principais contributos serão:

No âmbito de Visão Computacional, esta dissertação aplicará diversas técnicas avançadas de processamento de imagem e vídeo. Isso incluirá a utilização de algoritmos sofisticados para a análise e interpretação de dados visuais, contribuindo para o avanço das metodologias existentes.

Em *Deep Learning*, o trabalho focar-se-á na criação e treino de modelos capazes de realizar reconhecimento de objetos e reconhecimento facial com alta precisão. A investigação e o desenvolvimento de redes neurais profundas adaptadas para esta tarefa não só melhorarão a eficácia dos sistemas de vigilância, mas também poderão ser aplicadas a outras áreas que necessitem de reconhecimento facial confiável e eficiente.

Além disso, a presente dissertação abordará a construção de *hardware* específico para o sistema de vigilância, contribuindo diretamente para o domínio da IoT e Espaços Inteligentes. O desenvolvimento de dispositivos de *hardware* personalizados, integrados com o *software* de inteligência artificial, proporcionará um sistema completo e funcional, adequado para aplicações domésticas e potencialmente escalável para outros contextos. Este contributo é particularmente relevante, pois envolve a combinação de *software* e *hardware* para criar soluções práticas e inovadoras que melhoram a segurança e a gestão de ambientes inteligentes.

Este trabalho irá também contribuir significativamente para a melhoria na precisão e robustez dos atuais sistemas de videovigilância, com a aplicação das diversas técnicas de inteligência artificial.

Em resumo, esta investigação visa não apenas aprofundar o conhecimento em áreas fundamentais da Inteligência Artificial, mas também desenvolver uma aplicação prática que combina Visão Computacional, *Deep Learning* e IoT. Os contributos esperados terão um impacto significativo tanto na pesquisa académica quanto na aplicação industrial, oferecendo novas soluções e melhorias em sistemas de vigilância inteligentes.

1.6 Estrutura do documento

Este documento está organizado em cinco capítulos principais, cada um essencial para o desenvolvimento da solução apresentada.

Primeiramente, no capítulo da introdução são fornecidos o contexto, objetivos, motivação e contributos do trabalho. Nele, é fornecida uma visão geral do problema a ser resolvido e a importância da solução proposta.

De seguida, capítulo 2, são apresentados os principais conceitos e tecnologias relacionadas à inteligência artificial, com uma fundamentação teórica abrangente sobre diversas técnicas de reconhecimento facial. Detalha-se o funcionamento de métodos como *deep learning*, redes neurais convulsionais e técnicas de extração de características e classificação. Além disso, a metodologia PRISMA é aplicada na formulação de perguntas de pesquisa, refinando a abordagem de seleção e análise da literatura existente.

No capítulo 3 é detalhada a metodologia utilizada no desenvolvimento do sistema na sua implementação. Inclui a descrição da arquitetura da solução, os processos de pré-processamento de dados, os algoritmos e modelos implementados para deteção e reconhecimento facial. Também aborda a implementação do *hardware* e da aplicação móvel que agrega a solução assim como todas as funcionalidades do sistema.

No capítulo 4 são apresentados os métodos de validação e experimentação dos modelos desenvolvidos. Inclui a avaliação da precisão dos modelos de detecção e reconhecimento facial e a análise dos resultados obtidos. Também irá ser discutida a eficácia e eficiência do sistema em diferentes cenários.

Por fim, no capítulo 5, são expostas as conclusões do trabalho realizado, resumindo os objetivos alcançados, a sugestão de possíveis melhorias futuras e, ainda, as considerações finais.

.

2 Estado de arte e fundamentação teórica

Este capítulo encontra-se dividido em cinco partes, onde inicialmente é apresentada a contextualização teórica de conceitos importantes para o trabalho a ser desenvolvido, de forma a abordar os conceitos chave de inteligência artificial e técnicas de reconhecimento facial. De seguida é apresentada a metodologia de pesquisa utilizada assim como todos os passos para a realização da mesma. Posteriormente, são discutidos os resultados obtidos. Além disso, é explorado o tema da privacidade e proteção de dados, seguido de uma discussão sobre outras possíveis aplicações do sistema o desenvolvido. Por fim, o capítulo conclui com um sumário das principais conclusões.

2.1 Contextualização teórica

Neste subcapítulo, irão ser abordados de forma progressiva e detalhada, uma série de conceitos essenciais para a compreensão do estudo que será desenvolvido ao longo deste trabalho. Inicialmente o subcapítulo irá começar com uma visão geral sobre Inteligência Artificial, seguida pela exploração de tópicos mais específicos e especializados.

A abordagem sequencial adotada neste capítulo visa proporcionar uma compreensão clara e estruturada dos diferentes níveis de complexidade e especificidade envolvidos. Esta progressão permite, não apenas uma melhor contextualização dos conceitos, mas também uma apreciação das camadas tecnológicas que sustentam as soluções de reconhecimento facial.

2.1.1 Inteligência Artificial

A inteligência artificial pode-se definir como o estudo de algoritmos que permitem que as máquinas tenham capacidade de raciocinar e desempenhar as funções cognitivas, como resolução de problemas, reconhecimento de objetos ou palavras e tomada de decisão, de forma que assemelhe ao próprio comportamento humano [15].

Esta área, envolve a criação de sistemas que podem aprender e adaptar-se a novas situações, fazendo recurso de dados e experiências anteriores, de forma a melhorar seu desempenho ao longo do tempo [16] . Este processo de aprendizagem contínuo é essencial para o desenvolvimento de aplicações avançadas, em diferentes ramos da sociedade [16].

2.1.2 Machine Learning

Machine Learning é uma área da Inteligência Artificial, onde são aplicados algoritmos computacionais projetados para simular a inteligência humana através da aprendizagem automática reforço [17][18]. As técnicas baseadas em ML têm sido aplicadas com sucesso em diversos campos, incluindo reconhecimento de padrões, engenharia espacial, finanças, entretenimento, biologia computacional, aplicações biomédicas e médicas, entre outros [17].

Este ramo pode ser dividido em três subcategorias, nomeadamente a aprendizagem supervisionada, aprendizagem não supervisionada e a aprendizagem por reforço [18].

Na aprendizagem supervisionada, é fornecido ao algoritmo um conjunto de treino de dados que representam a entrada e saída, onde cada dado possui um valor esperado, denominado de *label*. Este tipo de aprendizagem tem como objetivo resolver problemas de classificação, que visam classificar itens ou amostras de acordo com características observadas pelo supervisor, assim como problemas de regressão, que têm como objetivo calcular a relação linear entre uma variável dependente e uma independente, utilizando essa relação para efetuar previsões [19].

Na aprendizagem não supervisionada, os algoritmos analisam e agrupam conjuntos de dados não rotulados. Estes algoritmos descobrem padrões ocultos ou agrupamentos de dados sem a necessidade de intervenção humana e possuem a capacidade de descobrir semelhanças e diferenças nas informações [20].

Por fim, na aprendizagem por reforço, o algoritmo aprende com base em seus próprios sucessos e erros na realização de uma determinada tarefa. Este método é amplamente utilizado em aplicações onde a tomada de decisão sequencial é crucial, como na robótica e no desenvolvimento de jogos [21].

2.1.3 Deep learning e Visão Computacional

Deep learning é uma área do ML que tem como o objetivo conseguir que um computador aprenda por si mesmo e execute tarefas semelhantes às dos seres humanos, através da interpretação de grandes quantidades de informação [18], [22]. Este ramo utiliza uma classe específica de algoritmos denominados de redes neurais profundas, que são essenciais no ramo da visão computacional [22].

Relativamente à visão computacional, esta é uma área que se dedica ao desenvolvimento de sistemas capazes de interpretar e compreender imagens e vídeos [23]. Esta área envolve técnicas de processamento de imagem e reconhecimento de padrões, de forma a capacitar os computadores a executar tarefas como identificação de objetos e reconhecimento facial [23]. A visão computacional abrange desde algoritmos básicos de filtragem e segmentação de imagens até métodos avançados de aprendizagem profunda, que permitem a extração de características e a tomada de decisões com precisão muito semelhante à humana.

2.1.3.1 Redes Neurais

Redes neurais são um conceito que visa focar no processamento de dados de maneira semelhante ao cérebro humano. O cérebro é tido como um processador altamente complexo e que realiza processamentos de maneira paralela [24] e para isso, ele organiza a sua estrutura, ou seja, os neurônios, de forma que eles realizem o processamento necessário, sendo este processo é feito numa velocidade extremamente alta [24].

Nas redes neurais é realizado o processamento de informações tendo como princípio a organização de neurônios do cérebro. Como o cérebro humano é capaz de aprender e tomar decisões baseadas na aprendizagem, as redes neurais artificiais devem fazer o mesmo [24]. Sendo assim, uma rede neural pode ser interpretada como um esquema de processamento capaz de armazenar conhecimento baseado em aprendizagem e disponibilizar este conhecimento para a solucionar um determinado problema [16][18].

Na saída de um neurônio, o sinal O é obtido pela seguinte relação:

$$O = f(net) = f\left(\sum_{i=1}^n x_i \cdot w_i\right) \quad (1)$$

Onde w representa o peso do vetor e o x apresenta o valor de entrada do neurônio. A capacidade de aprendizagem de um neurônio é obtida ajustando os pesos de acordo com algoritmo de aprendizagem escolhido [24].

A arquitetura geral de uma rede neuronal artificial (Figura 3) consiste em três tipos de camadas de neurônios, nomeadamente as camadas de entrada, ocultas e de saída [24].

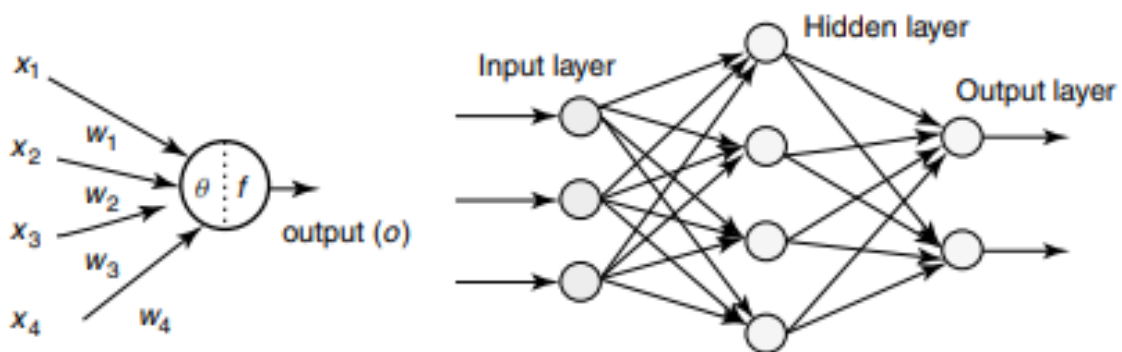


Figura 3 - Arquitetura de um neurônio (esquerda) e arquitetura de geral de uma rede neuronal (direita) [24].

2.1.4 Redes Neurais Convulsionais (CNN)

As Redes Neurais Convolucionais são uma classe de redes neurais profundas especialmente eficazes no processamento e análise de dados visuais, amplamente utilizadas no ramo de visão computacional. Estes tipos de redes são estruturados para processar e reconhecer padrões em imagens por meio da sua arquitetura, que inclui camadas projetadas para aprender hierarquias espaciais de características de forma automática e adaptativa [25].

Este tipo de rede é constituído por uma serie de camadas (Figura 4), nomeadamente:

1. **Camada de Entrada:** Camada responsável por manter os valores ds pixels brutos da imagem de entrada [25].
2. **Camada Convolutacional:** A camada convolutacional é o bloco de construção central de uma CNN. Nesta camada são aplicados uma série de filtros à imagem de entrada para produzir um mapa de características, de forma a destacar vários aspectos da entrada, como bordas, texturas e formas. Os filtros deslizam sobre a imagem de entrada, onde realizam multiplicações ponto a ponto e somam os resultados. Esta operação é crucial para detectar características espaciais na entrada de uma imagem[25], [26]
3. **Camada de Pooling:** As camadas de pooling são usadas para reduzir as dimensões espaciais, nomeadamente a largura e altura, dos mapas de características, o que diminui o número de parâmetros e a computação na rede. O tipo mais comum é o *max pooling*, que toma o valor máximo de uma porção do mapa de características coberta pelo filtro [25], [26].
4. **Camada Totalmente Conectada :** Estas camadas são semelhantes às encontradas em redes neurais regulares. Cada neurônio numa camada totalmente conectada está conectado a todos os neurônios na camada anterior. Essas camadas são usadas para combinar características aprendidas pelas camadas anteriores para classificar a imagem de entrada [25].
5. **Camada de Saída:** A camada de saída é reponsavel por produzir as probabilidades da classe prevista [26].

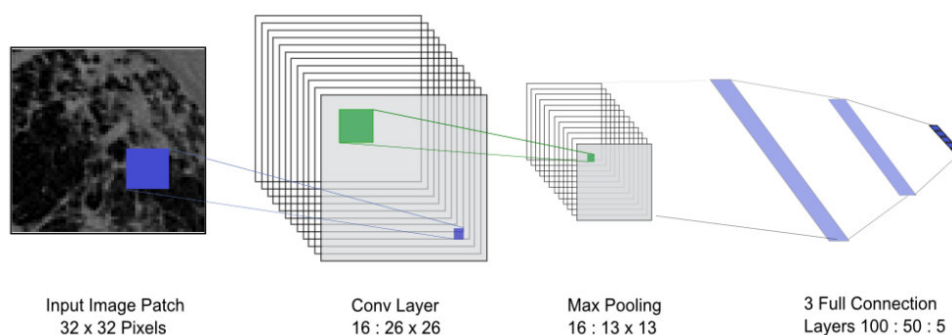


Figura 4 - Camadas de uma CNN [25].

2.1.5 Aprendizagem por transferência

A aprendizagem por transferência é uma técnica crucial em IA que envolve o uso de um modelo pré-treinado em um novo problema relacionado [27]. Esta abordagem aproveita o conhecimento adquirido de uma tarefa para melhorar o processo de aprendizagem e desempenho numa tarefa diferente. Esta técnica é notável pelas suas enúmeras vantagens, nomeadamente:

1. **Eficiência dos recursos:** Treinar redes neurais profundas do zero requer grandes quantidades de dados rotulados e um enorme poder computacional. Esta técnica mitiga essa necessidade ao usar modelos pré-treinados em grandes conjuntos de dados, como por exemplo o *dataset* ImageNet, que contém milhões de imagens rotuladas. Este reuso de modelos pré-treinados reduz significativamente o tempo de treino e o custo computacional [27].
2. **Melhoria no desempenho:** Modelos treinados em grandes e diversos *datasets* aprendem representações de características robustas que possuem uma boa generalização em diferentes tarefas. Ao ajustar esses modelos numa tarefa específica, é possível alcançar um melhor desempenho, comparado ao treino de um modelo do zero com dados limitados [28].
3. **Convergência mais rápida:** Começar com um modelo pré-treinado, a convergência do treino é mais rápida. Como o modelo já aprendeu características úteis, o número de épocas necessárias para alcançar um desempenho ideal na nova tarefa é mais reduzido [29].
4. **Aplicação em Vários Domínios:** A aprendizagem por transferência é amplamente utilizada em várias aplicações de IA, incluindo visão computacional, processamento de linguagem natural e reconhecimento de voz [30].

2.1.6 Reconhecimento Facial

O reconhecimento facial refere-se à deteção e a forma de identificar ou confirmar a identidade de uma pessoa utilizando os seus atributos faciais. Este tópico tem uma ampla gama de perspectivas de aplicação, incluindo estatísticas de tráfego, câmaras digitais e reconhecimento de padrões [31].

Este tipo de tecnologia tem se destacado como uma das mais emergentes devido às suas capacidades de uso em sistemas biométricos como uma alternativa ao reconhecimento individual que tradicionalmente utiliza *passwords*, chaves, PINs e cartões *RFID* [31], [32].

O reconhecimento facial está dividido em três [32] etapas, ilustradas na Figura 5, que são fundamentais para o seu funcionamento, sendo elas:

1. **Deteção Facial** – Esta é a fase inicial, onde é necessário localizar o rosto da pessoa na imagem ou no vídeo. Utilizando algoritmos de deteção, é preciso identificar a região da imagem onde o rosto está presente. Esse processo é fundamental, pois determina as áreas da imagem que devem ser analisadas nas fases subsequentes. A precisão na deteção é crucial para garantir que os algoritmos seguintes foquem apenas na parte relevante da imagem, ou seja, no rosto da pessoa.
2. **Extração de características** – Após a deteção do rosto, o próximo passo é a extração de características específicas da face. Nesta fase, algoritmos de processamento de imagem e *deep learning*, são utilizados para identificar as características únicas do rosto, como a forma dos olhos, nariz, boca e contornos faciais. O sucesso desta fase depende de uma extração eficiente das características mais discriminativas, permitindo que o sistema diferencie um indivíduo de outro.
3. **Classificação** – Com as características extraídas, o sistema compara essas informações com os dados armazenados na base de dados facial. O objetivo desta fase é identificar se o rosto pertence a uma pessoa conhecida ou, caso contrário, marcar como um indivíduo desconhecido. Esta fase é determinante para o sucesso do reconhecimento facial e é a que conclui o processo, determinando o resultado final.

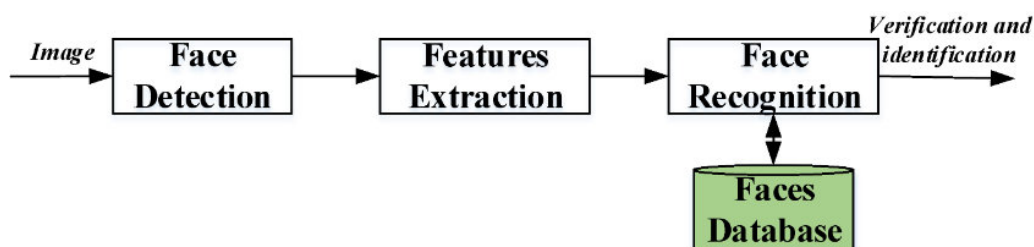


Figura 5 - Etapas do reconhecimento facial [32].

Em cada uma das etapas do reconhecimento facial, nomeadamente deteção facial, extração de características e classificação, existem diferentes métodos e técnicas que desempenham um papel crucial. Assim, os subcapítulos seguintes serão organizados de forma a abordar, detalhadamente, os diferentes modelos e abordagens associados a cada um desses passos. Primeiramente, será explorada as técnicas de deteção facial, seguida pelas técnicas de extração de características e por fim a classificação.

2.1.7 Detecção Facial

Neste subcapítulo, irão ser abordados e detalhados os principais métodos de detecção facial. Estes métodos estão divididos em duas grandes categorias: métodos tradicionais e métodos baseados em *deep learning*. Dentro dos métodos de *deep learning*, há duas subcategorias principais: os *singles stage detectors* e os *dual stage detectors*.

Os métodos tradicionais de detecção facial baseiam-se principalmente em características manuais e algoritmos clássicos de machine learning. Estes métodos utilizam técnicas como a análise de características faciais e a extração de padrões específicos nas imagens. Apesar de serem menos complexos e mais rápidos de implementar, muitas vezes carecem da precisão e robustez necessárias para lidar com uma variedade de condições de iluminação, poses e expressões faciais [33], [34].

Relativamente aos métodos baseados em deep learning, os *single stage detectors* são modelos de *deep learning* que realizam a detecção de objetos numa única etapa. Eles são conhecidos por serem rápidos e eficientes, uma vez que combinam a geração de regiões propostas e a classificação de objetos numa única rede neural [35]. Esses métodos são particularmente úteis em aplicações em tempo real, onde a velocidade é crucial. No entanto, podem apresentar limitações em termos de precisão quando comparados com métodos de duas etapas [35][36].

Por outro lado, os *dual stage detectors* realizam a detecção de objetos em duas etapas. Na primeira etapa, geram-se propostas de regiões onde os objetos podem estar localizados. Na segunda etapa, essas propostas são refinadas e classificadas em categorias específicas [37], [38]. Estes métodos tendem a ser mais precisos do que os *single stage detectors*, pois permitem uma análise mais detalhada e refinada das regiões propostas. No entanto, essa precisão adicional vem com um custo em termos de tempo de processamento, o que os torna menos adequados para aplicações que requerem respostas em tempo real [37], [38].

2.1.7.1 Viola Jones (Haar Cascade)

O método de detecção facial Viola-Jones ou também conhecido como Haar Cascade é um método tradicional, tendo sido o primeiro algoritmo baseado em detecção de objetos que fornecia uma boa taxa de detecção em tempo real [33], [39]. Este algoritmo possui 3 técnicas fundamentais para as operações de detecção, sendo elas o classificador *Haar*, o classificador *Ada Boost* e o classificador *Cascade* [39].

Primeiramente, o classificador *Haar* baseia-se em funções que permitem identificar regiões com diferentes intensidades no rosto, estando estas associadas a estruturas mais claras ou mais escuras. Neste processo são utilizados vários tipos de características, compostas por matrizes binárias com regiões adjacentes verticais ou horizontais de tamanho idêntico que permitem detetar estruturas como limites, linhas horizontais, verticais e oblíquas, ilustradas na Figura 6. Ao aplicar estas características a uma janela da imagem, a intensidade dos pixels das regiões

claras e escuras é somada e a diferença entre regiões é calculada. Valores elevados deste processo indicam uma semelhança entre a característica e a janela da imagem analisada [40].

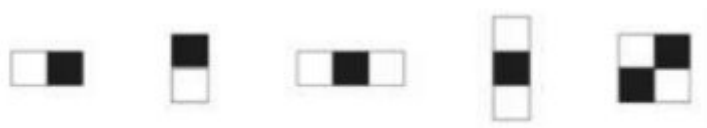


Figura 6 - Classificador Haar [33].

Na etapa de *Ada Boost* é selecionado o melhor subconjunto de características. Cada característica representa um classificador fraco, sendo verificada a sua performance em todas as sub-regiões da imagem. O peso de cada classificador é ajustado de acordo com a sua performance e um classificador forte (*boosted classifier*) é criado, que combina os classificadores fracos com menos erro associado. Este processo torna o algoritmo mais preciso já que permite detectar falsos positivos e verdadeiros negativos presentes nos dados [40].

Por fim, o classificador *Cascade* é utilizado para melhorar a velocidade e precisão do modelo. Cada subjanela das sub-regiões entra num processo iterativo no qual se verifica a presença das características fracas do classificador forte por ordem de importância. A ausência de qualquer uma destas características, leva a que se descarte a presença da estrutura que se pretende detetar na respetiva sub-região da imagem. Desta forma é possível descartar rapidamente as regiões da imagem onde não se encontram estruturas de interesse [40].

2.1.7.2 Histograma de Gradientes Orientados (HOG)

O Histograma de gradientes orientados é uma técnica tradicional baseada, que é utilizada para realizar a deteção de um rosto humano [41]. Este método está dividido em três fases, nomeadamente o cálculo dos gradientes, criação de histogramas e normalização dos blocos [42].

A primeira fase consiste em converter a imagem para escala de cinzentos e normalizá-la de acordo com a iluminação, este processo encontra-se representado na Figura 7. Esta escala pode ser obtida através de processos de deteção de iluminação, de forma horizontal e/ou vertical [42].



Figura 7 – (a) Imagem original, (b) Gradiente Horizontal, (c) Gradiente Vertical, (d) Gradiente Horizontal e Vertical [42].

Em seguida, na fase da criação de histogramas (Figura 8), é feita uma divisão da imagem em pequenas regiões espaciais denominadas de células. Para cada célula será calculado um histograma local das orientações sobre os pixels [42].

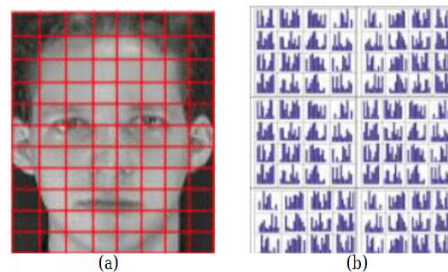


Figura 8 – (a) Divisão da imagem em células, (b) Histograma de cada célula [42].

Por fim, a fase de normalização dos blocos é feita com recurso ao acúmulo de histogramas locais em regiões espaciais um pouco maiores chamadas de blocos. Após a normalização é gerada uma janela de detecção sobre os histogramas gerados, que consiste no *output* do descritor HOG, onde são utilizadas as disposições das orientações dos vetores de gradiente para descrever a forma da imagem analisada [42].

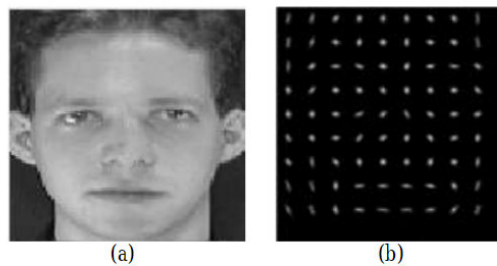


Figura 9 – (a) Imagem original, (b) Características HOG [42].

2.1.7.3 Single Shot Detection (SSD)

O *Single Shot Detection* é um algoritmo de *deep learning* para detecção de objetos baseado em regressão. Possui uma rede neuronal que pode ser utilizada para fazer a previsão da posição de um objeto, determinar a categoria e calcular a confiança de objetos em escalas diferentes [43].

Esta arquitetura encontra-se dividida em duas etapas, sendo a primeira a incorporação de um modelo de *backbone* e a segunda a cabeça do SSD. O modelo de *backbone* geralmente é composto por um modelo pré-treinado, como por exemplo VGG [43], ResNet [44] ou MobileNet [45], treinados no *dataset* ImageNet, na qual a camada final totalmente conectada foi removida. De seguida, a cabeça do SSD adiciona uma ou mais camadas convulsionais a esse *backbone*, e as saídas são interpretadas como caixas delimitadoras e classes de objetos [43].

A arquitetura deste modelo encontra-se ilustrada na Figura 10.

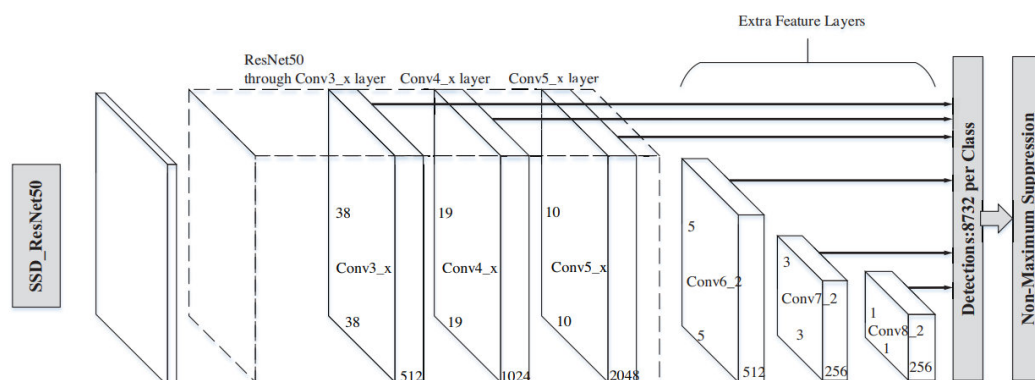


Figura 10 - Estrutura SDD baseada no modelo ResNet50 [43].

Este algoritmo é amplamente utilizado devido à sua alta precisão e velocidade, assim como uma melhor capacidade de detecção de objetivos pequenos em relação a outros detetores, sendo assim possui uma extrema usabilidade na detecção de rostos humanos [44].

2.1.7.4 Multi-Task Cascade Convolutional Neural Network (MTCNN)

A MTCNN é um dual shot detector projetado para correlacionar métodos de detecção e alinhamento facial. Esta rede é capaz de propor simultaneamente caixas delimitadoras, fornecer cinco pontos faciais que correspondem a marcas do rosto e calcular as probabilidades de detecção de cada rosto. A MTCNN divide as tarefas em três estágios: P-Net, R-Net e O-Net [46], representados na Figura 11.

Inicialmente, P-Net produz janelas candidatas utilizando uma SNN (Shallow Neural Network). Em seguida, a R-Net tem como objetivo rejeitar o máximo possível de janelas não faciais. Por fim, a O-Net utiliza uma rede mais complexa para refinar ainda mais a saída obtida na R-Net [46].

O modelo apresenta uma estrutura em cascata com três estágios de profundidade cuidadosamente projetados com redes convolucionais que classificam rostos e pontos de referência. Quando uma MTCNN processa uma imagem, é executada uma operação de redimensionamento para dimensionar a imagem original em diferentes escalas, de forma a gerar uma pirâmide de imagens. Em seguida, as imagens de várias escalas são enviadas para as três sub-redes de treino, de forma a permitir a detecção de diferentes tamanhos de rostos humanos e a detecção de alvos em múltiplas escalas [46].

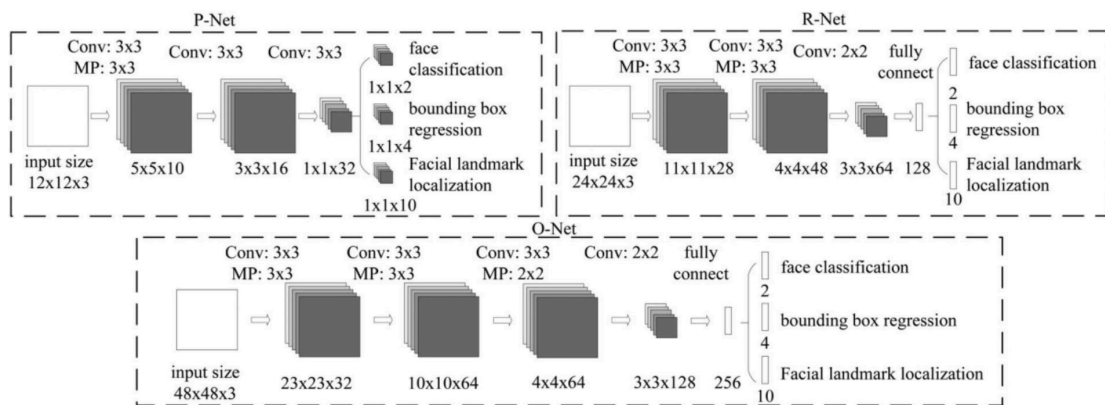


Figura 11 - Arquitetura MTCNN [46].

Além disso, a MTCNN implementa uma estratégia de mineração de amostras difíceis durante o treino, que melhora automaticamente a performance do modelo sem a necessidade de seleção manual de amostras. Esta rede demonstrou um bom desempenho em *benchmarks* desafiadores, demonstrando mantém boa performance em tempo real [46].

2.1.7.5 YOLO (You Only Look Once)

O modelo YOLO é um modelo de *deep learning*, sendo um modelo single shot que se destaca no campo de detecção de objetos, que pode ser ajustado para a realização de detecção facial. Atualmente, é um dos modelos de detecção de objetos mais rápidos e precisos disponíveis, capaz de processar imagens em tempo real.

Ao contrário de outros modelos de detecção de objetos, que aplicam o classificador de imagens para diferentes regiões, o modelo YOLO aplica uma única rede neuronal convolucional para a imagem inteira, dividindo-a em grades e prevendo limites e probabilidades para cada região [35].

A arquitetura do YOLO é baseada numa única rede convolucional que divide a imagem de entrada numa grade $S \times S$. Cada célula da grade prevê um número fixo de caixas delimitadoras e valores de confiança, que refletem a precisão das previsões e a probabilidade de que as caixas contenham certos objetos. A arquitetura da rede é composta por camadas convolucionais seguidas por camadas totalmente conectadas que produzem as previsões finais, fazendo recurso da rede neuronal DarkNet como *backbone* [35].

Este modelo possui diversas vantagens, destacando-se pela sua velocidade, pois processa a imagem inteira com uma única rede, sendo extremamente rápido e adequado para aplicações em tempo real. Além do mais, este modelo possui uma grande precisão na detecção dos objetivos [35].

Desde a sua introdução, várias versões deste modelo foram lançadas, cada uma com melhorias significativas em termos de precisão, velocidade e capacidade de detecção de objetos.

- **YOLOv1:** A primeira versão que introduziu o conceito de detecção em tempo real com uma única rede neural [35].
- **YOLOv2:** Melhorou a precisão e capacidade de detecção de objetos com a introdução de técnicas como ancoragem de caixas (*anchor boxes*) [47].
- **YOLOv3:** Aumentou a precisão e a capacidade de detecção de objetos menores utilizando uma arquitetura de rede mais profunda [48].
- **YOLOv4:** Introduziu várias melhorias como mosaico, *augmix*, *dropblock*, *cspdarknet53*, e outros, de forma a aumentar a precisão e a eficiência [49].
- **YOLOv5, YOLOv6, YOLOv7, YOLOv8:** O modelo teve uma evolução grande em nível de precisão e capacidade de detecção assim como simplicidade de implementação.

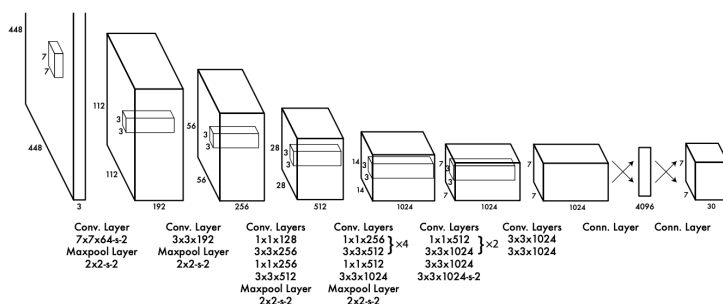


Figura 12 - Arquitetura YOLO [35].

2.1.7.6 RCNN, Faster RCNN e Masked RCNN

RCNN é uma abordagem pioneira que utiliza uma combinação de propostas de regiões e redes neurais convolucionais para a detecção de objetos. O processo (Figura 13) pode ser resumido nas seguintes etapas:

1. **Geração de Propostas de Regiões:** Utilizando algoritmos como o *Selective Search*, são geradas várias propostas de regiões que possivelmente contêm objetos [38].
2. **Extração de Características:** Cada proposta de região é redimensionada e passada através de uma rede neural convolucional para extrair características [38].
3. **Classificação e Regressão:** As características extraídas são alimentadas a uma rede de classificação que determina a presença de um objeto e sua classe, além de uma regressão da caixa de forma a refinar as coordenadas da caixa delimitadora [38].

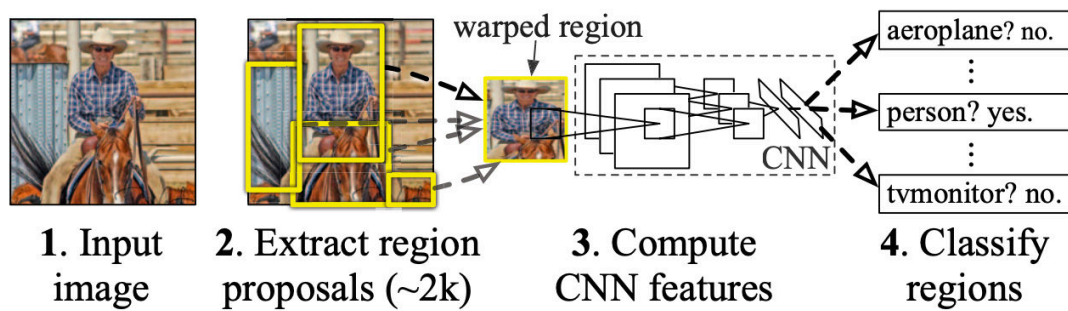


Figura 13 - Funcionamento do modelo RCNN [38].

Embora eficaz, o RCNN tem várias desvantagens, como a necessidade de redimensionar e passar cada proposta de região separadamente pela rede, o que é computacionalmente caro e com um tempo de processamento alto [38].

Sendo assim, o modelo Faster RCNN [37] surge como uma evolução do modelo RCNN que introduz uma Rede de Propostas de Regiões (RPN) para tornar o processo de geração de propostas mais eficiente e integrado.

1. **Geração de Propostas de Regiões com a rede RPN:** A RPN é uma rede neural que gera propostas de regiões de forma rápida e eficiente, eliminando a necessidade de métodos externos como o *Selective Search* [37].
2. **Extração de Características e Classificação:** As propostas de regiões geradas pela RPN são redimensionadas e utilizadas numa rede neural convolucional para extrair características, que são então classificadas e refinadas [37].

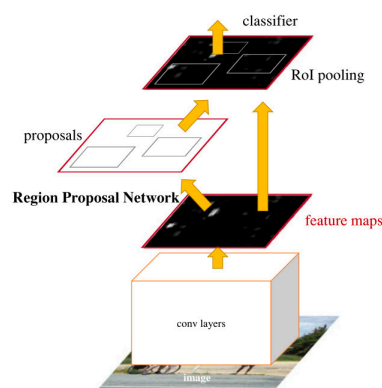


Figura 14 - Funcionamento Faster RCNN [37].

Por fim, o modelo Mask RCNN é uma extensão do Faster RCNN que adiciona uma terceira tarefa de segmentação, além da classificação e regressão da caixa. O funcionamento deste modelo está definido pelas seguintes etapas:

1. **Geração de Propostas de Regiões com RPN:** Semelhante ao Faster RCNN, utiliza-se uma RPN para gerar propostas de regiões [50].
2. **Extração de Características, Classificação e Regressão de Caixa:** As propostas de regiões são processadas para classificação e refinamento das caixas delimitadoras [50].
3. **Segmentação de Máscara:** Adiciona uma rede de segmentação de máscara que gera uma máscara binária para cada objeto detectado, permitindo a segmentação precisa dos objetos [50].

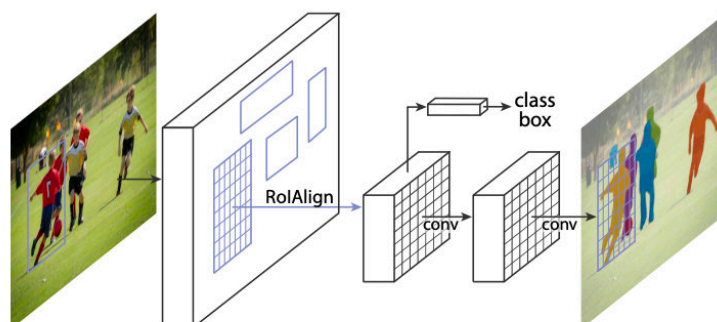


Figura 15 - Funcionamento do modelo Mask RCNN [50].

2.1.8 Extração de características.

Neste subcapítulo, irão ser abordados alguns dos métodos de extração de características na segunda etapa do reconhecimento facial, que assim como na etapa de detecção facial, podem ser divididos em técnicas tradicionais e técnicas baseadas em *deep learning*.

2.1.8.1 Principal Component Analysis (PCA) e Linear Discriminant Analysis (LDA)

A Análise de componentes principais e a análise discriminante linear são duas técnicas tradicionais amplamente utilizadas em ML para redução de dimensionalidade e classificação de imagens [51]. Embora compartilhem alguns objetivos semelhantes, como a simplificação de dados e a melhoria da eficiência computacional [51], estas duas abordagens diferem significativamente.

No que toca à técnica PCA, é um método não supervisionado que visa encontrar as direções (componentes principais) que capturam a maior variação nos dados. É frequentemente utilizada para simplificar conjuntos de dados de alta dimensionalidade, de forma a transformá-los em um espaço de menores dimensões, mantendo o máximo possível da variância original [17] [52].

O processo de implementação do PCA pode ser detalhado nas seguintes etapas [52]:

1. Calcular a média dos dados e centralizar os dados subtraindo a média.
2. Calcular a matriz de covariância dos dados centralizados.
3. Determinar os autovalores e autovetores da matriz de covariância.
4. Ordenar os autovetores por ordem decrescente e selecionar os principais componentes.
5. Projetar os dados originais nos componentes principais selecionados.

Relativamente ao LDA, ao contrário de PCA, é uma técnica supervisionada utilizada para encontrar uma combinação linear de características que melhor separe duas ou mais classes, pois considera as *labels* das classes para maximizar a separação entre diferentes classes nos dados [53]. Este método encontra-se dividido nas seguintes etapas [53]:

1. Calcular as médias das classes e a média geral dos dados.
2. Calcular as matrizes de dispersão intra-classe e entre-classes.
3. Resolver a equação para encontrar os autovalores e autovetores.
4. Ordenar os autovetores pelos autovalores em ordem decrescente e selecionar os discriminantes principais.
5. Projetar os dados originais nos discriminantes principais selecionados.

2.1.8.2 EigenFaces

O método Eigenfaces é uma técnica baseada em PCA explicitamente aplicada ao reconhecimento facial [54]. Este método transforma as imagens dos rostos em um conjunto de componentes principais, denominados de *eigenfaces* (Figura 16), que representam as direções principais das variações dos dados [55].



Figura 16 - Imagens Originais (esquerda) e EigenFaces (direita) [55].

2.1.8.3 FisherFaces

FisherFace é utilizada uma combinação entre os métodos PCA e LDA [54][56]. Esta abordagem usa PCA primeiramente para reduzir a dimensionalidade dos dados e LDA para maximizar a separação entre as classes, tendo como objetivo maximizar a distância de diferentes classes [56]. O processo resulta numa imagem que contém apenas os aspectos mais relevantes da imagem original, como ilustra a Figura 17.



Figura 17 – Imagem Original (esquerda) e FisherFace (direita) [56].

2.1.8.4 Principais Modelos Existentes

Neste subcapítulo irão ser abordados os principais modelos base (*backbones*) que são utilizados no reconhecimento facial assim como os modelos mais avançados de extração de características, que representam o atual estado de arte.

No que toca aos *backbones*, representam as arquiteturas de redes neurais projetadas para a extração de características robustas de imagens faciais. Essas arquiteturas, como ResNet, MobileNet, VGG e Inception, são amplamente utilizadas devido à sua capacidade de capturar e representar características faciais detalhadas e discriminativas. Estes *backbones* servem como blocos de construção fundamentais sobre os quais modelos mais avançados são desenvolvidos.

Redes ResNet

As redes ResNet são conhecidas pela capacidade de treinar redes neurais profundas através do uso de conexões residuais, que mitigam o problema da dissipação do gradiente. Existem diversas versões desta família de redes, nomeadamente, ResNet-50, ResNet-101 e ResNet-152 sendo todas amplamente utilizadas no reconhecimento facial devido à sua profundidade e capacidade de extrair características detalhadas. Esta arquitetura tem sido fundamental em várias aplicações de reconhecimento facial, oferecendo alta precisão e robustez [57], [58]. O uso de blocos residuais permite a criação de redes muito profundas, melhorando a performance em tarefas de classificação e reconhecimento [59].

Redes MobileNet

As redes MobileNet são projetadas para serem leves e eficientes, ideais para dispositivos móveis e sistemas com recursos computacionais limitados [60]. As versões MobileNetV2 e MobileNetV3 introduzem melhorias significativas em termos de eficiência e precisão, utilizando técnicas como convoluções em profundidade e blocos invertidos residuais. Estas redes são frequentemente usadas no reconhecimento facial onde a eficiência é crucial, mantendo um equilíbrio em termos de desempenho e precisão [61].

Redes VGG

As redes VGG, nomeadamente as arquiteturas VGG16 e VGG19, são conhecidas pela sua simplicidade e eficácia. Utilizam pequenas convoluções (3x3) e um grande número de camadas, o que resulta numa arquitetura profunda e poderosa [62]. Embora possuam muitos parâmetros, são populares devido à sua arquitetura intuitiva e desempenho sólido em várias tarefas de visão computacional.

Redes Inception

Por fim, as redes Inception, como a Inceptionv3 e Inception-ResNetv2, são famosas devido as suas arquiteturas complexas que combinam convoluções de diferentes tamanhos para uma extração de características mais detalhada. Este tipo de redes utiliza uma abordagem inovadora para melhorar a eficiência computacional e a precisão [63], [64]. A arquitetura Inception permite capturar diversas informações faciais de maneira eficiente, sendo particularmente útil em aplicações que exigem alta precisão. A combinação de convoluções de vários tamanhos numa única camada ajuda a extrair características em múltiplas escalas [65].

Os modelos avançados construídos sobre estas arquiteturas referidas, incluindo também, uma série de técnicas adicionais de otimização e treino em grandes conjuntos de dados específicos, mais voltados para o domínio de reconhecimento facial. Exemplos de tais modelos incluem VGGFace, ArcFace, CosFace, FaceNet e DeepFace. Estes modelos incorporam inovações que melhoram a precisão e a eficiência do reconhecimento facial, estabelecendo novos padrões de desempenho na área.

FaceNet

O modelo FaceNet, desenvolvido pela Google é um dos mais avançados e eficazes para reconhecimento facial. O principal avanço introduzido por este modelo é o uso de *embeddings* faciais, que são representações vetoriais compactas dos rostos humanos [66]. Este modelo utiliza como *backbone* a arquitetura Inception-ResNet, que combina a profundidade da Inception com a robustez das redes ResNet, de forma a capturar características faciais de maneira eficiente e precisa [66].

Ao invés de utilizar um classificador tradicional, o modelo FaceNet mapeia as imagens faciais diretamente para um espaço de *embeddings*, onde a distância entre os vetores corresponde à similaridade dos diversos rostos, o que permite não apenas a verificação, mas também a identificação e agrupamento de rostos de forma eficaz [66].

Uma característica fundamental deste modelo é o uso da função de perda *triplet loss*, que é essencial para o seu treino. Esta função utiliza triplos de imagens, nomeadamente imagem âncora, imagem positiva e imagem negativa para garantir que os *embeddings* das imagens do mesmo indivíduo (âncora e positiva) estejam mais próximos entre si do que com as de indivíduos diferentes (âncora e negativa). Esse método de treino promove a criação de um espaço de *embeddings* onde rostos semelhantes estão agrupados e diferentes estão bem separados [66].

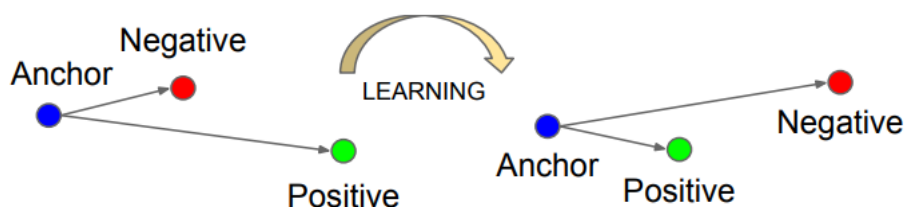


Figura 18 – Treino de *triplets* [66].

DeepFace

O DeepFace é um modelo criado pelo Facebook que identifica rostos humanos em imagens digitais. O programa emprega uma rede neural de nove camadas com mais de 120 milhões de pesos de conexão e foi treinado em quatro milhões de imagens carregadas por utilizadores do Facebook [67].

O principal avanço do DeepFace em relação a sistemas anteriores é a capacidade de criar representações tridimensionais do rosto a partir de uma imagem bidimensional. Este processo envolve o alinhamento do rosto numa pose frontal e normalizada, seguido pela extração de características discriminativas usando uma rede neural profunda. A arquitetura da rede é composta por várias camadas convulsionais que capturam diferentes níveis de abstração das características faciais, culminando numa camada de representação que pode ser usada para comparar rostos de maneira eficaz [67].

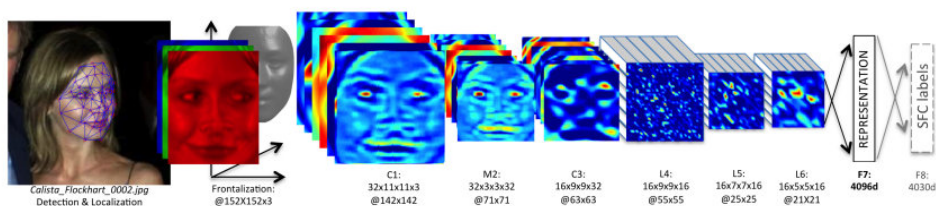


Figura 19 – Arquitetura do modelo DeepFace [67].

VGGFace

VGG-Face é um modelo avançado de reconhecimento facial desenvolvido pelo Visual Geometry Group da Universidade de Oxford. Este modelo é uma adaptação da conhecida arquitetura VGG16, otimizada especificamente para tarefas de reconhecimento facial. O modelo VGGFace demonstrou ser altamente eficaz em diversas aplicações de verificação e identificação facial [68]. Neste modelo, a imagem facial é passada através de uma rede neural profunda baseada na arquitetura VGG16, que consiste em 16 camadas (13 camadas convolucionais e 3 camadas totalmente conectadas). De seguida, a rede extrai características detalhadas do rosto, com recurso a filtros de 3x3 que capturam informações espaciais e texturais, de forma a gerar um vetor de embedding de 4096 dimensões [68].

ArcFace

ArcFace é um modelo avançado de reconhecimento facial que introduz uma nova abordagem para melhorar a discriminação entre os *embeddings* faciais. Este modelo utiliza a adição de uma margem angular à função de perda, conhecida como *Angular Margin Softmax*, para aumentar a separabilidade das representações faciais, sendo assim, maximiza a distância entre *embeddings* de diferentes identidades enquanto minimiza a distância entre *embeddings* da mesma identidade, aumentando significativamente a precisão do reconhecimento facial [69].

Esta função de perda, força o modelo a aprender *embeddings* mais discriminativos, o que melhora tanto a precisão quanto a sua robustez. Especificamente, a margem angular é incorporada na função de perda, o que modifica a forma como as similaridades angulares são calculadas durante o treino [69].

CosFace

CosFace é outro modelo avançado para reconhecimento facial, tem também como objetivo aumentar a separabilidade entre as representações de diferentes classes de rostos. Introduzido como uma solução para as limitações dos métodos anteriores, o modelo CosFace utiliza uma função de perda com margem cosseno (*Large Margin Cosine Loss*), que modifica a função de *softmax* tradicional, onde é adicionada uma margem do cosseno para melhorar a discriminação entre *embeddings* [70]. Este modelo tem demonstrado resultados impressionantes em *benchmarks* de reconhecimento facial, evidenciando sua eficácia e robustez. Comparado com outros modelos, o CosFace oferece uma melhoria significativa na precisão e na capacidade de generalização. Isso o torna uma escolha popular para sistemas de reconhecimento facial em aplicações de segurança, dispositivos móveis e outros contextos que requerem alta precisão [70].

Todos os modelos referidos representam o atual estado da arte no reconhecimento facial, exibindo resultados excepcionais no *dataset* Labeled Faces in the Wild [71]. Este dataset é amplamente utilizado como *benchmark* para avaliar a precisão e robustez dos sistemas de

reconhecimento facial. De seguida, na Tabela 1 é demonstra a performance destes modelos no *dataset* referido.

Tabela 1 - Performance dos modelos no dataset LFW.

Modelo	Acurácia
VGGFace	97.27% [68]
DeepFace	97.35% [67]
FaceNet	99.63% [66]
CosFace	99.73% [70]
ArcFace	99.83%[69]

2.1.9 Métodos de Classificação

Nesse subcapítulo, será abordada a última etapa do reconhecimento facial, nomeadamente as técnicas de classificação que utilizam os resultados das fases anteriores para classificar a imagem, ou seja, identificar ou não o rosto da pessoa.

2.1.9.1 K-nearest Neighbour (KNN)

KNN é um método amplamente utilizado para problemas de classificação, sendo especialmente popular no reconhecimento facial [72], devido ao facto de possuir um bom equilíbrio entre simplicidade e eficácia, o que o torna numa escolha frequente em muitas aplicações de *machine learning*. Este algoritmo classifica um novo padrão de teste com base nos k vizinhos mais próximos desse padrão no espaço de características [73].

O algoritmo KNN segue um processo estruturado em várias etapas, conforme ilustrado na Figura 20. O algoritmo começa com a inicialização e definição do valor de k, que corresponde ao número de vizinhos a serem considerados.

O KNN calcula a distância entre o padrão de teste e todos os pontos de dados do conjunto de treino. Existem várias maneiras de medir essa distância, e a escolha da métrica pode influenciar o desempenho do algoritmo. As métricas de distância mais comuns incluem:

- **Distância de Manhattan:** Calcula a soma das diferenças absolutas entre as coordenadas correspondentes dos pontos no espaço de características.

$$d(x, y) = \sum_{i=1}^k |X_i - Y_i| \quad (2)$$

- **Distância Euclidiana:** Calcula a distância geométrica direta entre dois pontos no espaço multidimensional.

$$d(x, y) = \sqrt{\sum_{i=1}^k (X_i - Y_i)^2} \quad (3)$$

- **Distância de Minkowski:** Uma generalização da distância Euclidiana, onde o valor de q define o tipo de distância.

$$d(x, y) = \left(\sum_{i=1}^k (|X_i - Y_i|)^q \right)^{1/q} \quad (4)$$

Após o cálculo, as distâncias são ordenadas em ordem crescente, de forma a identificar os pontos de treino mais próximos ao padrão de teste. Com as distâncias ordenadas, o algoritmo seleciona os k vizinhos mais próximos [73]. Por fim, para determinar a classe do padrão de teste, o KNN realiza uma votação entre os k vizinhos mais próximos. A classe mais frequente entre esses vizinhos é atribuída ao padrão de teste [73].



Figura 20 – Funcionamento do KNN [72].

2.1.9.2 Support vector machine (SVM)

O SVM é um método amplamente utilizado para problemas de classificação e regressão, sendo especialmente eficaz no reconhecimento facial [74]. A principal vantagem do SVM encontra-se na sua capacidade de encontrar um hiperplano ótimo que separa as classes no espaço de características, maximizando a margem entre as classes distintas. Este algoritmo faz recurso de funções *kernel* para transformar os dados num espaço de alta dimensionalidade onde um hiperplano pode ser utilizado para a separação [75]. Os principais kernels são:

- **Kernel Linear:** Adequado para dados linearmente separáveis.
- **Kernel Polinomial:** Bom para dados onde a relação entre as características é polinomial.
- **Kernel RBF (Radial Basis Function):** Popular para dados que não têm uma separação linear clara, onde é necessário capturar relações não lineares entre características.

Esta técnica procura também um hiperplano que maximize a margem entre as classes de dados, garantindo uma separação clara entre elas, onde essa maximização pode ser feita com recurso a dois tipos diferentes de hiperplanos, nomeadamente:

- **Hiperplano linear:** Utilizado quando as classes de dados podem ser separadas de forma clara por uma linha ou plano [76].
- **Hiperplano não linear:** utilizado quando os dados são muito complexos e não podem ser separados por uma linha reta simples [76].

Para cada um dos hiperplanos existem também tipos de separação, nomeadamente:

- **Separação arbitrária:** Referece a qualquer linha ou plano que separa duas classes de dados no espaço de características, mas sem a garantia de ser a melhor separação possível, podendo não maximizar a margem entre as classes, o que leva a uma classificação menos precisa e mais susceptível a erros [74].
- **Separação ótima:** Refere-se à separação de duas classes, de maneira a que margem (a distância entre o hiperplano e os pontos de dados mais próximos de cada classe) seja maximizada, de forma a fornecer uma maior distância de segurança entre as classes, resultando numa classificação mais robusta e menos susceptível a overfitting [74].

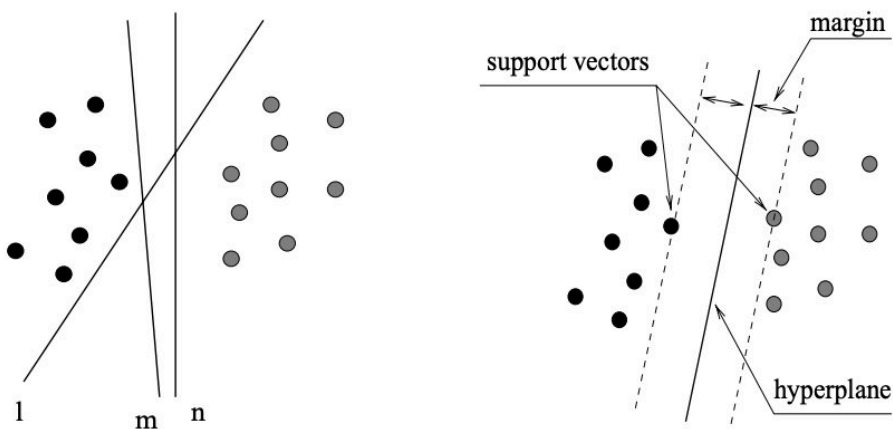


Figura 21 - Tipos de separação [74].

2.2 Revisão Sistemática da Literatura

De forma a recolher informação fundamentada sobre as atuais tecnologias de inteligência artificial na implementação de técnicas de deteção e reconhecimento facial foi utilizada uma metodologia de investigação formal, através de uma revisão sistemática com recurso a uma versão adaptada da metodologia PRISMA [77], de forma a poder responder a três questões de pesquisa.

2.2.1 Questões de pesquisa

Um dos primeiros passos de uma revisão sistemática é a formulação de um conjunto de perguntas que pretende responder ao problema, já apresentado no capítulo 1, sendo assim três questões principais foram escolhidas para o estudo deste trabalho que se encontram representadas na Tabela 2.

Tabela 2 — Questões de pesquisa

Questão de pesquisa	
QP1	Quais são os algoritmos/técnicas necessários para implementar técnicas de reconhecimento facial?
QP2	O quão precisos são os algoritmos/técnicas de reconhecimento facial em situações desfavoráveis?
QP3	Que mecanismos/soluções de vigilância com recurso a reconhecimento facial semelhantes à solução proposta existem?

A primeira questão de pesquisa tem como objetivo compreender quais métodos, técnicas, algoritmos e modelos são utilizados no desenvolvimento de mecanismos de deteção e reconhecimento facial, com o intuito de identificar quais são os mais adequados para a implementação da solução proposta.

A segunda questão de pesquisa permite identificar os principais desafios e problemas enfrentados pelos modelos atuais em condições adversas, como iluminação inadequada ou ângulos desfavoráveis. Através dessa investigação, será possível compreender as limitações desses algoritmos e trabalhar em estratégias para mitigá-las, assim garantindo que a solução final seja capaz de lidar eficazmente com essas dificuldades. Ao estudar esses problemas, o objetivo é implementar um sistema de reconhecimento facial mais robusto, capaz de operar com alta precisão, mesmo em ambientes onde a qualidade das imagens ou vídeos não seja ideal.

Por fim, a terceira questão tem como objetivo questionar que soluções semelhantes com a solução proposta já existem, mas que façam recurso das tecnologias de reconhecimento facial, implementados num sistema completo, semelhante ao proposto. As soluções devem, para além de demonstrar implementações de modelos de AI, demonstrar esses modelos numa aplicação prática, simulando um sistema de vigilância.

2.2.2 Fontes de informação

Após a seleção de um conjunto de perguntas, o próximo passo de uma revisão sistemática refere-se à identificação de fontes de informação, para que esta mesma informação seja tão útil quanto possível para o desenvolvimento do documento e para a obtenção de resultados. Portanto, quatro fontes de informação foram escolhidas, estando representadas na Tabela 3.

Todas estas fontes de informação são bastante conceituadas na procura de artigos relativos ao mundo das tecnologias de informação, sendo este o principal motivo para a sua escolha.

Tabela 3 — Fontes de informação

Fonte	Url
Web of Science	https://www.webofscience.com
ScienceDirect	https://www.webofscience.com
ACM	https://dl.acm.org/
IEEE Xplore	https://ieeexplore.ieee.org

2.2.3 Termos de pesquisa

De forma a facilitar a pesquisa nas fontes de informação foram escolhidas *keywords* específicas (Tabela 4) de forma a obter os artigos mais relevantes para o estudo pretendido.

Tabela 4 — Domínios e *keywords* usadas na seleção dos termos de pesquisa

Domínio	Keywords
Artificial Intelligence	Artificial Intelligence, Machine Learning, AI.
Computer Vision	Computer Vision, Deep Learning, Image Classification.
Face Recognition	Face Detection, Face Recognition, Biometric Identification.

Surveillance	Video Surveillance, Surveillance System, Security Camera, Detection System
--------------	--

2.2.4 Extração de informação

Para efetuar a pesquisa de artigos nas fontes de informação referidas, foram construídas diversas *query strings* (Tabela 5) de modo a obter o conjunto de artigos que possuíam todas as palavras chaves.

Tabela 5 — Query Strings

Fonte	Query String
Web Of Science	(Computer Vision OR Deep learning OR Image Classification) AND (Face Detection OR Face Recognition OR Biometric Identification OR Image Classification) AND (Video Surveillance OR Surveillance System OR Security Camera OR Detection System)
ScienceDirect	(Computer Vision OR Deep learning OR Image Classification) AND (Face Detection OR Face Recognition OR Biometric Identification OR Image Classification) AND (Video Surveillance OR Surveillance System OR Security Camera OR Detection System)
ACM	(Artificial Intelligence OR Machine Learning) AND (Computer Vision OR Deep learning OR Image Classification) AND (Face Detection OR Face Recognition OR Biometric Identification OR Image Classification) AND (Video Surveillance OR Surveillance System OR Security Camera OR Detection System)
IEEE Xplore	(Artificial Intelligence OR Machine Learning) AND (Computer Vision OR Deep learning OR Image Classification) AND (Face Detection OR Face Recognition OR Biometric Identification OR Image Classification) AND (Video Surveillance OR Surveillance System OR Security Camera OR Detection System)

Para mais, foram também criados diversos critérios de inclusão e exclusão de forma a realizar uma filtração após os resultados obtidos da pesquisa. Todos os critérios de inclusão e exclusão encontram se ilustrados na tabela 6 e 7.

Tabela 6 — Critérios de Inclusão

ID	Critério
CI1	A fonte pertence ao domínio da Inteligência Artificial.
CI2	A fonte descreve uma contribuição significativa para os campos de estudo.
CI3	A fonte apresenta soluções ou problemas relacionados com reconhecimento facial.

Tabela 7 — Critérios de Exclusão

ID	Critério
CE1	A fonte foi publicada há mais de 7 anos.
CE2	A fonte não está escrita em inglês.
CE3	A fonte não possui qualquer conteúdo sobre o estudo das tecnologias de reconhecimento facial / detecção de objetos.

Após a construção das devidas *querys strings* para as diferentes fontes de informação foi realizada a pesquisa, onde foram obtidos um total de 566 artigos. Os resultados obtidos por cada fonte de informação encontram se ilustrados na Tabela 8.

Tabela 8 — Resultados por fonte de informação

Fonte	Número de Resultados
Web of Science	111
ScienceDirect	156
ACM	82
IEEE Xplore	217

Como indica a metodologia PRISMA [77] é necessário fazer a filtragem dos diversos artigos das diferentes bases de dados, seguindo todos os passos da metodologia (identificação, remoção de duplicados, triagem, elegibilidade e inclusão). Sendo assim, de forma a economizar o maior tempo possível neste processo, foi utilizada a ferramenta Covidence³.

Como já referido, um total de 566 estudos foram reunidos das diferentes fontes de informação, sendo removidos, um total de 32 artigos duplicados entre estas mesmas fontes. De seguida foi necessário recorrer ao *screening* do título e do *abstract* de cada documento, contabilizando um total de 382 estudos removidos pois não estavam de acordo com os critérios de inclusão e exclusão previamente estabelecidos, resultando apenas 152 para o *screening* do texto completo. Nesta fase, foram excluídos cerca de 109 artigos pois também não se encontravam de acordo com os critérios.

No fim, dos 566 artigos recolhidos inicialmente foram escolhidos 43 estudos que possuíam alta relevância para o estudo a ser efetuado. Toda a sequência de acontecimentos esta ilustrada na Figura 22.



Figura 22 – Diagrama Prisma.

³ Covidence - Better systematic review management - <https://www.covidence.org/>

2.2.5 Resultados

Nesta secção irão ser apresentados os resultados obtidos no fim da pesquisa, fazendo referência a cada artigo e qual das questões de investigação esse artigo ajudou a responder.

2.2.5.1 Quais são os algoritmos/técnicas necessários para implementar técnicas de reconhecimento facial?

Dos 43 artigos seleccionados para o estudo 24 faziam referência aos algoritmos e técnicas mais utilizados no desenvolvimento de tecnologias de reconhecimento facial.

No que toca ao processo de deteção facial, foram recolhidos inúmeros artigos que retratam o uso do algoritmo Viola-Jones/Haar Cascade, nomeadamente [78][79][80][81][82]. Ainda relativamente a métodos tradicionais, o uso de HOG foi abordado também em [82][83].

Relativamente a métodos baseados em *deep learning*, foram obtidos artigos que relatavam o uso do modelo MTCNN [80][82][84][85][86]. Para mais, o uso de modelos SSD foi abordado em [82] e [87]. Por fim, ainda relacionado com modelos de deteção, foram recolhidos artigos que implementam o uso do modelo YOLO [88][89][90] e do modelo *Faster RCNN* [91][92].

Para extração de características, na sua grande maioria os artigos relatavam o uso de CNN [80][85][93][94][95][96][97][98][99][100][101]. Na Tabela 9 encontram-se de forma mais resumida e detalhada as redes *backbone* mais abordadas assim como os modelos pré treinados, relatados ao longo dos diferentes artigos:

Tabela 9 - Modelos de deep learning abordados nos artigos.

Modelo	Artigo(s)
AlexNet	[96][98][99]
ResNet50	[93][94][96][98][99]
MobileNetV2	[94]
VGG16	[94][99][101]
GoogleNet	[97][98]
FaceNet	[80][85][93][95][101]
MobileFaceNet	[86]
InceptionV3	[98]
DeepFace	[95]
VGGFace	[95]
CosFace	[95]

Para além das referidas abordagens com *deep learning*, foram recolhidos artigos com recurso a técnicas tradicionais de extração de características, nomeadamente com uso das PCA em [81] [99][87][101], LDA [99][87][101] e LBP [80][97][101].

Por fim, no que toca as operações de classificação grande parte dos artigos relatava o uso de SVM [80][81][83][96][97][87][101] assim como o uso de KNN [83][100] e classificadores com recurso a uma simples distância euclidiana foram implementados em [85][86]. O classificador MPL foi também abordado em [97] e [87].

2.2.5.2 O quão precisos são os algoritmos/técnicas de reconhecimento facial em situações desfavoráveis?

Diversos problemas podem afetar as precisões dos algoritmos no processo de efetuar o reconhecimento facial, sendo assim, dos 43 artigos 10 artigos foram recolhidos que fazem o estudo desses problemas.

Primeiramente, foi recolhido o artigo [102] no qual é demonstrada a diferença entre os métodos tradicionais e os métodos baseados em *deep learning* em situações não controladas.

O primeiro problema analisado faz referência ao uso de imagens de má qualidade, ou seja, imagem que possuam baixa resolução, este problema encontra se descrito em [103] e [104].

Outro grande problema relacionado é a angulação de deteção do rosto. Nem sempre as imagens que alimentam os modelos de AI vão chegar nas melhores condições de angulação, sendo em [105] é abordado uma solução que soluciona este problema.

A luminosidade também pode ser um fator que irá prejudicar os sistemas de reconhecimento facial, sendo assim 2 artigos que retratam este problema foram também selecionados [106] [107].

O último problemas analisado diz respeito à obstrução do rosto, que tem vindo a ser um problema bastante comum nos dias de hoje devido à situação pandémica que existiu até 2022 e ao consequente uso de uma máscara. Relativamente a este tópico 4 artigos foram selecionados [108] [109] [110] [111].

2.2.5.3 Que mecanismos/soluções de vigilância com recurso a reconhecimento facial semelhantes à solução proposta existem?

Dos 43 artigos recolhidos cerca de 9 possuíam alguma semelhança com a solução proposta.

Relativamente a soluções integradas para um ambiente doméstico, foram recolhidos os artigos [112] [113] [114] [115] [116] [117]. Soluções que utilizam uma simples câmara e reconhecimento artificial na marcação de presenças em estabelecimentos educativos também foram abordadas [118] [119] [120].

2.2.6 Discussão

Neste capítulo irá ser discutido cada um dos artigos selecionados de forma resumida, apenas realçando os pontos mais importante descritos em cada um deles e a sua contribuição para o estudo efetuado.

2.2.6.1 Quais são os algoritmos/técnicas necessários para implementar técnicas de reconhecimento facial?

Para obter o máximo de informações sobre as principais técnicas de reconhecimento facial, os artigos foram analisados detalhadamente, de modo a identificar claramente os mecanismos utilizados em cada estudo. A análise teve como objetivo identificar, pelo menos, um tipo de algoritmo aplicado em uma das fases do processo de reconhecimento facial, nomeadamente nas etapas de deteção facial, extração de características ou classificação, contribuindo assim para o estudo em desenvolvimento.

Nos artigos [78] e [79] é apresentado o algoritmo Viola Jones para a realização de operações que envolvam deteção facial, onde é detalhado o funcionamento do mesmo. Estes artigos foram relevantes para perceber que embora o algoritmo Viola Jones seja antigo, pode ainda ser considerado para implementar um sistema de deteção facial simples, onde a qualidade das imagens, luminosidade e obstrução não são um problema.

Ainda em relação aos mecanismos de deteção facia, é importante perceber a diferença entre performance de métodos tradicionais e métodos baseados em *deep learning*. Sendo assim no artigo [82] é feita uma comparação bastante abrangente onde são apresentados diversos algoritmos e modelos para realizar deteção facial. Neste artigo são apresentados os algoritmos Viola-Jones, HOG, MTCNN, SSD e MMOD, sendo feita uma análise comparativa entre eles. Os resultados desta análise estão ilustrados na Figura 23. O artigo conclui que dos modelos e algoritmos selecionados, o que possui melhor performance é o SSD.

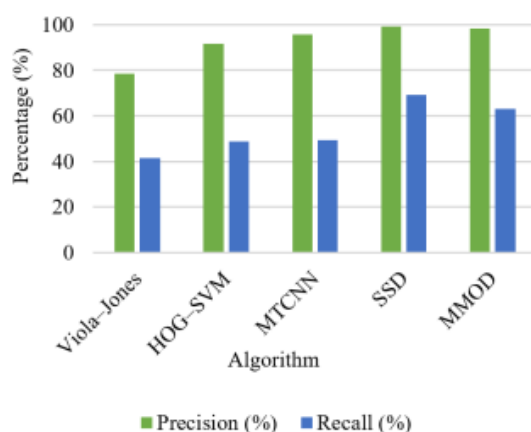


Figura 23 – Comparação entre algoritmos de detecção facial [82].

Em [84] é apresentado uma implementação de um sistema de detecção. Este sistema possui uma rede MTCNN para realizar a operação de detecção facial e uma rede neuronal siamesa para a operação de reconhecimento facial, combinando a extração de características e a classificação no mesmo processo. Foi utilizado um *dataset* com 50 indivíduos e foi obtida uma precisão de 86%.

Nos artigos selecionados, três deles focavam no estudo dos processos de detecção com recurso ao modelo YOLO. Primeiramente em [88] uma análise comparativa de algumas versões deste modelo, nomeadamente YOLOv3, YOLOv4 e YOLOv5. O artigo concluiu que este modelo possui resultados excelentes em operações de detecção facial, mostrando uma evolução gradual do desempenho deste modelo tipo de modelo nas versões mais recentes, nomeadamente YOLOv5 (até a data do artigo). Para mais, no artigo [89] o modelo YoloV5 é abordado, mas na sua versão mais compacta (YOLOv5s) e por fim, em [90] é implementando um poderoso sistema de detecção facial, utilizando como base o modelo YOLOv8. Este modelo sobre uma melhoria onde é aplicado um módulo GMConv, uma variante de convulsão que utiliza a média generalizada para combinar valores de pixels, oferecendo uma maior flexibilidade e robustez, que aumenta consideravelmente a performance do modelo base.

Todos os artigos recolhidos sobre o YOLO, tanto na versão mais recente (v8) como na versão mais antiga (v5) demonstram uma capacidade excepcional deste modelo ao realizar operações de detecção. Sendo assim, este modelo mostra-se uma excelente opção para ser integrada no sistema que é proposto a desenvolver.

Relativamente a mais solução que fazem recurso de *deep learning* nas operações de detecção, no artigo [91] é implementado um sistema de detecção facial com recurso ao modelo Faster RCNN utilizando o uma rede ResNet101 como *backbone*. Este modelo foi treinado com recurso ao dataset WIDERFace. Assim como o YOLO, este modelo revelou ser extremamente poderoso.

O uso do modelo Faster RCNN também é abordado em [92] para detectar rostos em quadros japoneses antigos. Para que tal fosse possível, necessário recorrer ao dataset Kouhon Dataset, que continha milhares de quadrados e as devidas anotações das bounding boxes.

É importante também estudar e compreender os algoritmos de classificação. Um estudo feito em [83] demonstra uma abordagem muito interessante com a conjunção dos algoritmos de classificação SVM e KNN. Neste artigo é implementado um sistema de reconhecimento facial apenas com recurso a HOG e classificadores SVM e KNN. O processo inicia-se com uma imagem de *input*, que irá ter as suas características extraídas com recurso ao método HOG. De seguida essas características irão ser classificadas pelo KNN que irá servir de *input* para o SVM para melhor a precisão do sistema. No fim é concluído que o uso de ambos algoritmos fornece uma melhor precisão do que o seu uso separado (Figura 24).

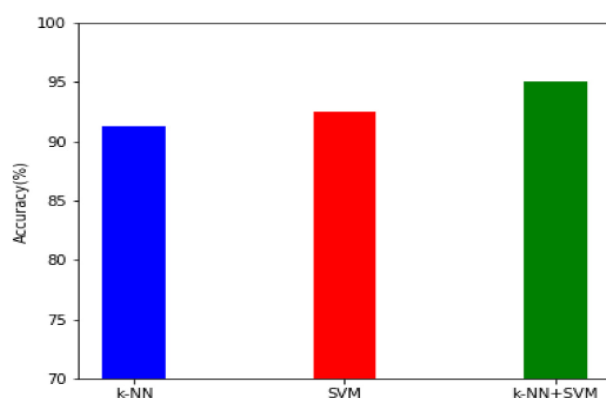


Figura 24 - Comparação entre os métodos de classificação em [83].

Relativamente aos restantes artigos, estes não se limitavam a uma única etapa do reconhecimento facial, mas abordavam o processo como um todo, englobando as fases de deteção, extração de características e classificação. Cada um destes artigos foi essencial para recolher informações valiosas sobre os diferentes métodos e modelos utilizados em cada etapa, proporcionando uma visão mais completa e integrada das técnicas de reconhecimento facial que poderão ser aplicadas no estudo em desenvolvimento.

Primeiramente em [85] é apresentado um sistema de reconhecimento facial denominado de “MiPRE” capaz de proteger a privacidade de menores de idade. A deteção do rosto é feita com recurso a uma MTCNN e o modelo FaceNet é utilizado para fazer a extração de características. A classificação é feita com recurso a distância Euclidiana. Este sistema faz uso um algoritmo de embaralhamento de forma para proteger a identidade de um menor que está a ser detetado.

Em [80] é feita uma comparação entre os 2 tipos de abordagens de técnicas de reconhecimento facial em tempo real. Na primeira abordagem é utilizado novamente o método Viola Jones para fazer a deteção facial, sendo aplicado de seguida o algoritmo LBPH para realizar a extração de características. Seguidamente, esta abordagem é testada com um *dataset* de 300 fotos de 3 indivíduos distintos tendo obtido uma precisão de 50%. Na segunda abordagem foram utilizadas metodologias de *deep learning*, nomeadamente MTCNN para a deteção e o modelo pré treinado FaceNet para realizar a operação de extração, tendo obtido uma precisão de 96%, tendo ambos as abordagens feito recurso do classificador SVM. A análise deste artigo mostrou-se essencial para visualizar a diferença entre métodos tradicionais e métodos de *deep learning* na implementação de um sistema de reconhecimento facial em tempo real.

No artigo [81] é implementado um sistema de reconhecimento facial apenas recorrendo a métodos tradicionais, nomeadamente Viola Jones para deteção facial, PCA combinado com EigenFaces para extração e por fim é aplicado o algoritmo SVM para fazer a devida classificação. Este artigo revelou que estas técnicas tradicionais podem ser utilizadas para um sistema de reconhecimento facial pequeno e simples, não sendo aconselhada para sistemas mais complexos.

No artigo [86] é demonstrado uma implementação de um sistema de vigilância para estações elétricas. Foi utilizado novamente o modelo MTCNN para a realização da detecção facial, em contrapartida ao uso do algoritmo Viola Jones, devido ao facto de possuir maior eficácia nas aplicações no mundo real. Para extração de características foi utilizada a rede MobileFaceNet, uma variação do modelo FaceNet, mais compacto, perfeito para dispositivos moveis com baixo processamento gráfico, procedendo à classificação com recurso à distância euclidiana.

Em [87] é fornecido um estudo comparativa dos resultados do uso de diferentes combinações de algoritmos de extração de características e classificação. Inicialmente o artigo refere o uso de um modelo SSD para a detecção facial e, relativamente aos algoritmos de extração de características o artigo referiu a utilização LDA, PCA e CNN. No que toca à operação de classificação foram usados SVM, MLP e novamente CNN para classificação com uma camada de output softmax. Foram feitos testes de performance com recurso a vídeo em direto. No que to primeiro teste, ilustrado na figura 25, todas as abordagens obtiveram resultados positivos, com precisões acima dos 86%. Por outro lado, em relação aos testes em tempo real a precisão da CNN comparada aos demais foi superior, tendo obtido precisões de 89% contra uma média de 56%, representadas na figura 26.

Sr. No.	Classifier	Feature Extraction technique	Accuracy (%)
1	SVM	PCA	88
		LDA	86
2	MLP	PCA	86
		LDA	87
3	CNN	-	98

Figura 25 – Performance dos algoritmos com uma imagem de input [87].

Sr. No.	Classifier	Feature Extraction technique	Accuracy (%)
1	SVM	PCA	57
		LDA	55
2	MLP	PCA	55
		LDA	57
3	CNN	-	89

Figura 26 – Performance dos algoritmos com imagem em tempo real [87].

Em [93] é feito um pequeno estudo sobre reconhecimento de atributos de um rosto com recurso a *deep learning*. Para isso, implementa um modelo Triplet Loss, simulando a estrutura do modelo FaceNet, com recurso a uma rede ResNet como backbone, obtendo resultados positivos após a implementação.

Em [94] faz o estudo de 3 modelos, nomeadamente VGG16, Resnet50, MobileNetV2, e com recurso a *transfer learning* implementa 3 modelos de reconhecimento facial, sem recorrer a um classificador extra, utilizando apenas uma camada *softmax* de output. Os testes realizados com imagens estáticas demonstraram performances por volta dos 90% de acurácia em todos os modelos.

Em [95] é são propostos dois modelos profundos baseados em árvore para a operação de reconhecimento facial, nomeadamente um modelo de árvore única e a um modelo de árvore em paralelo. Através do resultado no *dataset* LFW, o artigo concluiu que estes modelos propostos possuem performances competitivas comparadas aos modelos atuais modelos de reconhecimento facial.

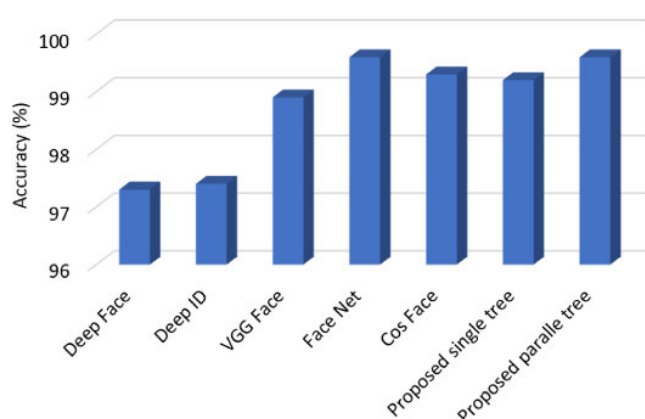


Figura 27 – Comparação da solução com modelos existentes em [95].

No artigo [96] é feito o estudo comparativo entre 2 tipos de abordagens. Primeiramente são utilizados os modelos AlexNet e ResNet50, para a operação de extração de características e o classificador SVM para a classificação. Na segunda abordagem, apenas é utilizado o modelo AlexNet com uma camada *softmax* de output a atuar como classificador. Ambas as abordagens foram avaliadas numa serie de *datasets*, possuindo bons resultados.

No artigo [97] são discutidas 3 possíveis abordagens para a implementação de um mecanismo de reconhecimento facial. A primeira abordagem faz referência ao uso da técnica LBP com MPL, a segunda faz recurso de SVM e LBP e por fim uma abordagem de *deep learning* mais concretamente com o uso do modelo GoogleNet. No fim, o artigo relatava a melhor performance por parte do modelo GoogleNet, seguido pela abordagem que usa SVM com LBP e por fim LBP com MPL. O artigo destaca também a performance, em termos de tempo, do modelo de *deep learning* é inferior as outras duas abordagens, representadas na Figura 26.

	Cpu Run time (mn)	Recognition rate
LBP+MLP	108.06	97.29 %
LBP+SVM	149.20	99.01 %
Googlenet	301.47	99.61 %

Figura 28 - Performance das abordagens em [97].

No artigo [98] é apresentado um modelo de reconhecimento facial denominado “BESDTL-AIFR”. Este modelo faz recurso da rede InceptionV3 como *backbone* para a operação de extração de características e uma DBN para a operação de classificação. O artigo demonstra que sistema obteve bons resultados em comparação a outras metodologias de reconhecimento facial (extração de características + classificação). A comparação encontra-se ilustrada na Figura 29.

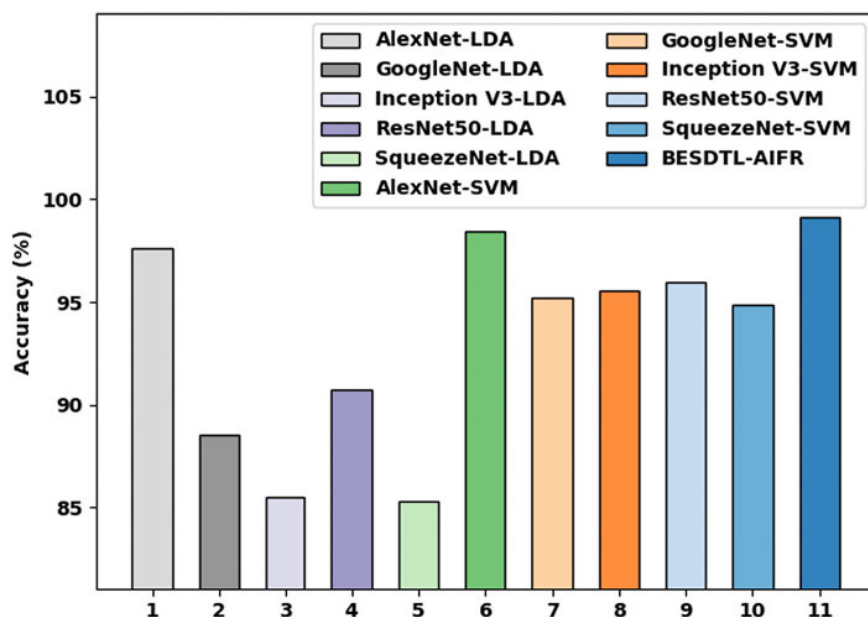


Figura 29 – Comparação de soluções em [98].

No artigo [99] é estudada a performance de 5 tipos de CNNs, nomeadamente VGG16, Alexnet, Resnet50, Resnet18 e Squeezenet, para operação de reconhecimento facial e a classificação é realizada através da camada *softmax* de output nas CNN's referidas. Todos os modelos abordados no artigo atingiram bons resultados, acima dos 95% de acurácia para todos os modelos, tendo sido testados num *dataset* privado de 30 elementos.

Em [100] é demonstrada a implementação de um modelo de reconhecimento facial com o recurso de uma rede neuronal residual, com uma função *triplet loss*, espelhando o modelo FaceNet. Para a classificação o artigo relata o uso do algoritmo KNN.

Relativamente ao último artigo [101], onde é feito um estudo teórico sobre as tecnologias de espectro de imagens em sistemas biométricos, no qual é referido com bastante detalhe o reconhecimento facial. No artigo os autores fornecem uma contextualização teórica sobre métodos utilizados no reconhecimento facial nomeadamente o uso de modelos como VGG e FaceNet, LBP, SVM, LDA e PCA.

Em suma, todos os artigos analisados foram extremamente relevantes para o estudo e compreensão das dezenas de técnicas e modelos que podem ser utilizados nas etapas de deteção facial, extração de características e classificação. Cada estudo trouxe uma abordagem única e contribuiu para uma visão mais abrangente das possibilidades tecnológicas, desde métodos tradicionais até soluções mais modernas baseadas em *deep learning*. Estas análises permitiram entender como os diferentes algoritmos se comportam em contextos variados, proporcionando um ponto de partida sólido para a implementação de um sistema de vigilância.

2.2.6.2 O quão precisos são os algoritmos/técnicas de reconhecimento facial em situações desfavoráveis?

Através da leitura e análise detalhada de todos os 10 artigos recolhidos, é possível afirmar que diversos fatores podem afetar de forma negativa os mecanismos de deteção e reconhecimento facial. Primeiramente é necessário entender que os diferentes tipos de técnicas, nomeadamente as técnicas tradicionais e as técnicas de *deep learning* possuem diferentes comportamentos quando as condições onde os algoritmos/técnicas estão a ser aplicado não são as melhores. Como foi relatado no artigo [102], onde é feito um estudo comparativo entres métodos de tradicionais e modelos de *deep learning* no contexto do reconhecimento facial em situações não controladas, sendo exemplos, ausência de uma boa luminosidade e obstrução parcial do rosto. Neste artigo, numa primeira fase são avaliados 3 algoritmos de extração de características tradicionais, nomeadamente LBP, HOG e SURF com o classificador SVM. Numa segunda fase são utilizados modelos AlexNet, VGG-16, VGG-19 e Inception-V3, também com recurso ao classificador SVM. De seguida, todos os algoritmos são testados no *dataset* EKFD, onde o artigo conclui uma notória vantagem dos modelos de *deep learning* face aos metodos tradicionais (Figura 30 e 31).

Com a análise deste artigo é possível ver uma grande diferença em termos de performance entre metodos tradicionais em comparação aos modelos de deep learning. Sendo assim é necessário garantir que a proposta para este trabalho faça recurso destas tecnicas mais poderosas para o desenvolvimento de um sistema mais poderoso e com mais torença a falhas.

Method	N	P	PO	I	FE	Overall
LBP+SVM	25.00	30.76	23.08	23.08	24.04	25.21
HOG+SVM	34.61	48.05	30.77	36.54	32.69	36.11
SURF+SVM	55.77	25.00	35.86	48.08	44.23	38.68

Figura 30 - Resultados dos métodos tradicionais em[102].

Architecture	N	P	PO	I	FE	Overall
Alexnet	71.15	49.03	57.05	55.77	67.30	58.97
VGG-16	51.92	42.30	41.03	46.15	53.84	45.94
VGG-19	51.92	48.07	44.23	50.00	48.08	47.44
Inception-v3	67.31	60.58	61.54	61.54	72.11	64.32

Figura 31 - Resultado dos métodos de deep learning em [102]

Em [103] é feito um estudo que foca na diferença entre os avanços da tecnologia de reconhecimento facial em ambientes controlados como exemplo redes sociais e o desempenho limitado dessas técnicas em cenários de vigilância, onde as imagens são de baixa qualidade, com resolução reduzida. Sendo assim o artigo propõem um dataset especialmente desenhado para avaliar o desempenho dos modelos de *deep learning* em situações de baixa qualidade de imagem, denominado de SurvFace. Este *dataset* é composto na sua maioria por fotos em condições não favoráveis, nomeadamente com má resolução. De seguida, diversos modelos são *finetuned* neste dataset, nomeadamente CentreFace, VGGFace e SphereFace, onde os resultados foram de 61.3%, 51.0% e 64.0% respetivamente. Este artigo é essencial para perceber que a qualidade que as imagens chegam aos modelos de IA, impactam imenso estes sistemas. Sendo assim, para garantir uma melhor performance as imagens devem ser sempre ser transmitidas melhorar qualidade possível, sendo também necessário garantir que o *dataset* onde os modelos são treinados possuam algumas amostras de má qualidade, de modo a tentar cobrir os cenários onde esta não exista.

Outra possível solução, ainda relativamente à má qualidade de imagem é descrita em [104], onde é proposto um método inovador que utiliza uma combinação de técnicas de reconstrução de imagens e mapeamento de características. Após a melhoria da imagem, essa imagem é então utilizada pelo modelo de reconhecimento facial. Esta solução melhora tanto a qualidade visual quanto a precisão do reconhecimento, preservando as identidades faciais.

Em [105] é feito o estudo da precisão de técnicas de deteção facial em ângulos não favoráveis. Para isso, foi apresentado um *dataset* de treino, denominado de TFD que possuía cerca de 11 124 imagens de 927 pessoas diferentes entre 6 possíveis ângulos. Neste artigo foram abordados 3 modelos de deteção facial, sendo eles MTCNN, YOLOv3 e Faster RCNN. Após o treino tendo em conta o *dataset* criado, os algoritmos foram testados no *dataset* YaleBTilt. Foram obtidos resultados de IoU de cerca de 0.90 para Faster RCNN, 0.86 para o modelo YOLOv3 e 0.76 para a rede MTCNN. O artigo também efetuou testes em tempo real, no qual o modelo YOLOv3 se destacou em comparação aos demais. No fim, o artigo concluiu que os modelos de deteção facial, sendo treinados com amostras onde existe uma grande variedade de ângulos faciais, irão produzir resultados muito satisfatórios e robustos.

No artigo [106] é explorada a aplicação de redes neurais profundas para melhorar imagens com iluminação não uniforme, como as capturadas à noite por câmeras de vigilância. Utilizando uma

rede neural convolucional, pré-treinada com um conjunto de dados que inclui versões bem iluminadas e escurecidas de imagens, o estudo propõe uma metodologia para realçar imagens com iluminação não uniforme. A pesquisa também analisa as características convolucionais extraídas em cada camada da rede neural, identificando-as partes da imagem que ajudam a rede a reconhecer um determinado objeto. O artigo detalha também, os métodos de aumento de dados e a aplicação de mascaramento para simular fontes de luz não uniformes, resultando em uma melhora significativa na precisão da rede neural para detetar objetos em imagens de baixa luminosidade.

Ainda relacionado ao problema da luminosidade em sistemas de reconhecimento facial, o artigo [107] compara a precisão de modelos de reconhecimento facial em condições de iluminação variadas utilizando imagens produzidas por câmeras Kinect em relação a câmeras RGB convencionais. O estudo utilizou a técnica de SVM para classificação e analisou imagens em condições de baixa e alta iluminação. Os resultados mostraram que o uso de imagens produzidas por câmeras Kinect, que incluem imagens em infravermelho, profundidade e 3D, aumentou a precisão da classificação em cerca de 20% em comparação com as câmeras RGB padrão. Em ambientes de iluminação normal, a precisão aumentou cerca de 5%, enquanto em condições de pouca iluminação a precisão melhorou significativamente, variando de 20% a 40%.

Ambos os artigos [106] e [107], fornecem soluções práticas e técnicas para melhorar a eficácia do reconhecimento facial em ambientes com baixa luminosidade, que é uma condição frequente em sistemas de vigilância.

O artigo [108] investiga como o uso de máscaras afeta a eficácia dos sistemas de reconhecimento facial, especialmente durante a pandemia de COVID-19. O estudo realiza uma série de experimentos para comparar o desempenho de diferentes modelos que representam o estado de arte em reconhecer rostos humanos. O artigo realça importância de durante o treino destes modelos, ser necessário incorporar imagens com contenham rostos com máscaras, de modo a melhor a performance dos modelos.

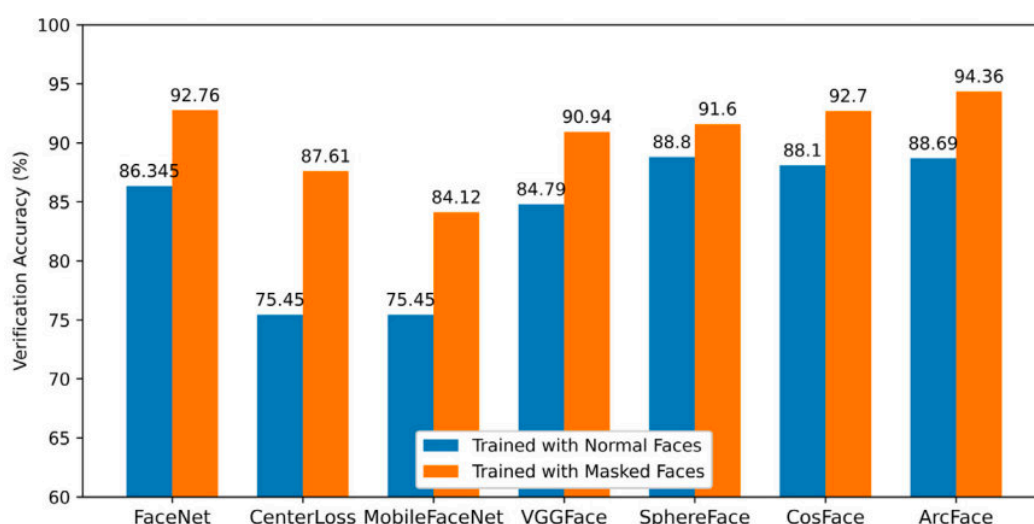


Figura 32 - Performance dos modelos em [108].

No artigo [109], investiga também o desenvolvimento e a eficácia de um sistema de reconhecimento facial para identificar pessoas com máscara, com recurso a *deep learning*.

O artigo [110] revisa várias técnicas de redes neurais convolucionais usadas para o reconhecimento de rostos mascarados, especialmente no contexto da pandemia de COVID-19. O artigo refere o uso de redes RPN, HOG, MTCNN e YOLOv2/v3 para a deteção facial e a aplicação de redes pre-treinadas nomeadamente VGG16 e InceptionV3. Neste artigo os modelos de *deep learning* mostraram um resultado animador nas diversas operações realizadas.

Em [111] é discutido o uso de EigenFaces e técnicas de deep learning para o reconhecimento de rostos mascarados. A combinação de PCA/EigenFaces e deep learning visa aproveitar os pontos fortes de ambas as abordagens. A PCA reduz a dimensionalidade dos dados de entrada, tornando o processamento mais eficiente, enquanto as CNNs aprimoram a capacidade de reconhecimento ao lidar com as variações introduzidas pelo uso de máscaras. O artigo conclui que a combinação de PCA e deep learning é uma abordagem promissora para o reconhecimento de rostos mascarados, oferecendo melhorias em termos de precisão e robustez.

Todos os quatro artigos recolhidos, que abordam o reconhecimento facial com o uso de máscaras, são interessantes para perceber como é que os modelos se adaptam quando os rostos encontram-se parcialmente obstruídos. Embora a solução proposta nesta tese seja um sistema mais direcionado para um ambiente doméstico, onde o uso de máscaras é praticamente nulo, é interessante que o sistema consiga pelo menos realizar as operações de deteção facial de um rosto praticamente obstruído, por exemplo um assaltante. Para que isso aconteça é necessário que o dataset de deteção possua amostras de rostos mascarados e obstruídos, de forma a que os modelos sejam poderosos o suficiente para detetar estes casos.

2.2.6.3 Que mecanismos/soluções de vigilância com recurso a reconhecimento facial semelhantes à solução proposta existem?

Os artigos analisados apresentam semelhanças com a solução proposta nesta tese. Embora alguns estejam mais direcionados para ambientes domésticos e outros para contextos diferentes, como educação e escritórios, continuam a ser relevantes. Mesmo que o foco não seja exclusivamente o ambiente doméstico, a implementação proposta nos artigos pode ser facilmente adaptada a esse contexto, mantendo a essência e a estrutura do sistema semelhante.

Primeiramente em [112] é fornecida uma implementação de um sistema de vigilância com reconhecimento facial para uma casa, através da construção de um mecanismo IoT, onde é utilizada uma Raspberry Pi Zero W, conectada uma câmara NoIR Pi e um sensor de movimento PIR. Numa primeira fase o algoritmo Viola Jones foi utilizado para fazer a deteção facial de um indivíduo, seguido pela utilização do algoritmo LBPH para a extração das características. Por fim, na operação de classificação da imagem foi utilizada a distância euclidiana. O sistema era capaz de notificar o utilizador através de uma notificação enviada para o telemóvel. No fim, o valor da precisão do sistema para a funcionalidade de reconhecimento facial foi de 84%.

No artigo [113] descreve o desenvolvimento de um sistema de segurança de fechaduras inteligente utilizando a tecnologia IoT e reconhecimento facial. Para a deteção facial, o sistema emprega o modelo YOLO numa placa Jetson Nano, de forma a permitir um processamento rápido das imagens capturadas pela câmara Raspberry Pi v2. O sistema utiliza técnicas de deep learning para reconhecer rostos com alta precisão, alcançando uma precisão de até 99% e um tempo de reconhecimento de 0,6 segundos em condições ideais de iluminação e distância. A deteção e o reconhecimento são realizados em tempo real, e os utilizadores podem gerir as fechaduras inteligentes através de uma aplicação movel.

O artigo [114] propõe um sistema de reconhecimento facial que preserva a privacidade das pessoas a ser vigiadas, fazendo recurso de filtros de forma a representar os dados faciais. O método permite que as tarefas analíticas sejam realizadas sobre representações de dados protegidas, mantendo alta confidencialidade dos dados. O sistema transforma imagens faciais em uma representação de filtros, que é armazenada e processada em servidores não confiáveis, protegendo assim a privacidade dos indivíduos. Os dados faciais são inicialmente extraídos utilizando métodos tradicionais de extração de características, nomeadamente o algortimo SVD, recorrendo também ao algoritmo SVM para a classificação.

Em [115] é apresentado o desenvolvimento de um sistema de deteção facial utilizando o Raspberry Pi 4. O sistema é projetado para ser económico e eficiente, empregando técnicas de deep learning. A deteção facial é realizada utilizando o classificador Viola Jones. O artigo mencionado não refere quais os mecanismos empregues na fase de reconhecimento facial e classificação.

O artigo [116] detalha o desenvolvimento de um sistema de segurança inteligente para acesso a uma casa. Para a detecção facial, o sistema emprega o método Viola-Jones e para o reconhecimento facial, o sistema utiliza o método EigenFaces. Uma Raspberry Pi é utilizada como o microprocessador central, garantindo que o sistema seja de baixo custo e compacto. O sistema abre automaticamente a porta para pessoas reconhecidas e envia uma foto dos desconhecidos para um servidor *web*, que notifica o proprietário via e-mail, seguindo uma abordagem extremamente semelhante ao que se pretende implementar.

No artigo [117] é explorada a aplicação de um sistema de vigilância para um escritório. O sistema está empregue para o acesso a um determinado espaço, através de uma porta. Todo o mecanismo de reconhecimento facial, nomeadamente detecção e extração e características é realizado através do modelo Faster RCNN, com a rede VGG16 de backbone. Este sistema tem muitas semelhanças com a solução pretendida neste projeto, nomeadamente com o alerta de notificações e envio de fotos quando um comportamento suspeito é detetado.

Em [118] descrito um sistema de monitoramento de presença baseado em reconhecimento facial, projetado para ser executado em um Raspberry Pi com uma câmera de vigilância. O sistema é modular e pode ser aplicado em salas de aula e laboratórios para vigilância em tempo real ou registro de presença, mesmo sob condições de iluminação precária, devido às técnicas de pré-processamento de imagem utilizadas. O sistema armazena automaticamente os dados de presença num servidor *online*. Este sistema de vigilância utiliza o classificador Haar Cascade para detecção facial e realiza o reconhecimento facial com recurso ao classificador LBPH. No artigo, o autor refere que obteve uma precisão de 74%, referindo que seria possível aumentar a precisão da solução se fossem implementadas técnicas mais poderosas de *deep learning*.

Ainda relacionado ao tópico de sistema de marcação de presença com recurso a video vigilância, em [119] é fornecida uma nova implementação. Nesta solução, o sistema irá marcar automaticamente as presenças numa plataforma online, onde era possível criar um perfil de aluno de forma a guardar o seu rosto numa base de dados. A solução utilizada poderosos modelos de *deep learning* para realizar tanto as operações de detecção como as de reconhecimento facial, com recurso ao modelo YOLOv3 e uma CNN para a extração de características, utilizando também a distância euclidiana como classificação. No fim, esta combinação de algoritmos revelou-se extremamente promissora obtendo resultados superiores a 92.50% de precisão em aulas presenciais e 100% de precisão nas aulas online.

Por fim, um último artigo relacionado com um sistema de marcação automático de faltas foi abordado em [120]. Neste artigo foi implementado um sistema automático de controlo de presenças que utiliza uma combinação de técnicas de *deep learning* para reconhecimento facial. Para a detecção de facial o modelo faz recurso do modelo Tiny YOLOv3, e para o reconhecimento facial, o sistema emprega um conjunto de modelos avançados como VGG-Face, OpenFace, FaceNet e DeepFace, que são combinados para aumentar a precisão do reconhecimento. Esses modelos transformam as características faciais em vetores, que são comparados usando métricas de distância como distancia euclidiãna e distância do cosseno. O

uso de múltiplos modelos e métricas permitiu melhorar a robustez e a acurácia do sistema, atingindo altas taxas de precisão em diferentes condições de iluminação e ângulos.

Embora os artigos recolhidos apresentem soluções interessantes, estas estão longe de se tornarem sistemas que possam ser comercializados de forma eficiente e moderna. Muitos dos estudos utilizam técnicas tradicionais de deteção, como o algoritmo Viola-Jones/Haar Cascade, que, embora relevante no passado, já não é suficiente para dias de hoje, especialmente quando comparado com soluções de deep learning, que oferecem uma precisão muito superior. Além do mais, os sistemas analisados apresentam pouca flexibilidade e personalização por parte dos utilizadores, onde normalmente há apenas um algoritmo para deteção e outro para reconhecimento, limitando assim as possibilidades de ajuste conforme as necessidades específicas de cada cenário.

Outra grande limitação dessas soluções é a ausência de mecanismos de vídeo em direto, essenciais para que o utilizador possa monitorizar o seu espaço em tempo real, além de uma configuração muitas vezes manual e complexa. Em muitos casos, os sistemas dependem da criação de um *dataset* prévio para um número específico de indivíduos, o que significa que a adição de novas pessoas requer um processo manual e ineficiente, sem qualquer dinamismo ou capacidade de adaptação automática.

A solução que se pretende implementar tem como objetivo superar todas estas limitações ao fornecer não só melhores modelos de deteção e reconhecimento, mas também uma infraestrutura mais robusta e completa. O sistema será altamente personalizável, o que permite qualquer utilizador configurar-o de forma simples e eficaz, com uma interface intuitiva e acessível. Além disso, integrará mecanismos de vídeo em direto, permitindo a visualização em tempo real, e oferecerá uma configuração dinâmica e automática para a inclusão de novos indivíduos, tornando-o uma solução muito mais avançada e prática do que as propostas recolhidas nos artigos. Em resumo, o sistema a desenvolver visa combinar as melhores práticas de IA com uma infraestrutura moderna e centrada no utilizador, que proporciona uma solução de vigilância doméstica inteligente que responde às necessidades atuais de segurança e tecnologia.

2.3 Privacidade e proteção dos dados

Quando o assunto é sobre vídeo vigilância com recurso a reconhecimento facial, inevitavelmente irá existir um processamento de dados biométricos dos utilizadores, sendo assim possui um impacto a nível da privacidade. Para lidar com este problema, a união europeia criou o RGPD (Regulamento Geral sobre a Proteção de Dados) que começou a ser aplicada em 25 de maio de 2018 [121].

Como o objetivo do reconhecimento facial é identificar, autenticar e verificar um indivíduo através dos seus dados biométricos, neste caso o rosto, o RGPD aplica-se. Sendo assim é necessário respeitar todas as suas normas, nomeadamente:

1. **Consentimento - Artigo 6 (1.a):** O consentimento deve ser sempre dado pelo utilizador. Assim, o utilizador terá de aceitar ou recusar utilizar o sistema por escolha própria. No contexto da solução proposta, após a inicialização da aplicação, será solicitado ao utilizador o consentimento para o uso dos seus dados no sistema de vigilância.
2. **Transparência – Artigo 12 (1):** O titular dos dados deve ser sempre informado adequadamente sobre o processamento dos mesmos. Esta informação deve ser fornecida no momento da obtenção dos dados pessoais ou antes de ser dado o consentimento.
3. **Limitação de Propósito - Artigo 5 (1.b):** Os dados pessoais dos indivíduos devem ser recolhidos para um propósito específico, explícito e legítimo. Assim, os dados pessoais não poderão ser utilizados para outros fins que não sejam os inicialmente estipulados.
4. **Medidas de Segurança – Artigo 32:** Medidas técnicas e organizativas devem ser incorporadas ao sistema para proteger os dados pessoais e assegurar a sua integridade, confidencialidade e disponibilidade.
5. **Retenção de Informação - Artigo 5 (1.e):** Para garantir que a informação é guardada durante o menor intervalo de tempo possível, é necessário assegurar que dados desnecessários sejam apagados. No caso da solução proposta, sempre que o sistema detectar um comportamento suspeito, ele realizará algumas capturas de imagem para armazenar essa informação, caso seja necessário. Caso contrário, essas informações serão automaticamente apagadas num determinado intervalo de tempo.
6. **Minimização de Informação – Artigo 5 (1.c):** É necessário garantir que a recolha de dados seja relevante e limitada para cumprir o propósito específico. No caso do sistema abordado, apenas informações dos rostos dos utilizadores autenticados serão guardadas, além de informações temporárias quando houver um comportamento suspeito.
7. **Direito de Acesso – Artigo 15:** Os utilizadores têm o direito de aceder aos seus dados pessoais e obter informações sobre como esses dados estão a ser processados. Isso inclui a confirmação de que os dados estão a ser processados, o propósito do processamento, e as categorias de dados pessoais em questão.

8. **Direito ao Apagamento – Artigo 17:** É necessário permitir que os utilizadores solicitem a eliminação dos seus dados pessoais quando não forem mais necessários para os fins para os quais foram recolhidos ou quando o consentimento for retirado.
9. **Direito à Portabilidade dos Dados – Artigo 20:** Os utilizadores têm o direito de receber os seus dados pessoais num formato estruturado, de uso comum e legível, e de transmitir esses dados a outro responsável pelo tratamento sem impedimentos.
10. **Avaliação de Impacto sobre a Proteção de Dados – Artigo 35:** Para sistemas que processam dados pessoais de forma a apresentar um alto risco para os direitos e liberdades dos indivíduos, é necessário realizar uma avaliação sobre a proteção de dados antes de iniciar o processamento.

2.4 Algumas aplicações de videovigilância

Embora a presente dissertação seja focada no desenvolvimento de um sistema de vigilância inteligente para ambiente doméstico, ressalva-se a sua versatilidade, que permite que seja facilmente adaptada para outras áreas. Algumas possíveis aplicações incluem:

1. **Ginásios:** Atualmente, muitos ginásios utilizam cartões *RFID* para controle de acesso. A implementação da solução de reconhecimento facial pode substituir ou complementar este sistema, garantindo que apenas membros autorizados entrem nas instalações, melhorando a segurança e facilitando o acesso sem a necessidade de cartões físicos.
2. **Controle de Acesso em Empresas:** A aplicação pode ser utilizada para controlar o acesso a determinados departamentos dentro de uma empresa, assegurando que apenas funcionários autorizados possam entrar em áreas sensíveis. Isso aumenta a segurança interna e protege informações confidenciais.
3. **Instituições de Ensino:** Em escolas e universidades, a solução pode ser utilizada para monitorar o acesso a edifícios e salas de aula, garantindo a segurança dos alunos e do corpo docente. Pode também ser utilizada para registrar a presença dos alunos de forma automática.
4. **Hospitais e Clínicas:** A implementação da solução em instituições de saúde pode ajudar a monitorar o acesso a áreas restritas, como salas de cirurgia ou farmácias, garantindo que apenas pessoal autorizado entre nessas áreas críticas.
5. **Condomínios Residenciais:** Em grandes complexos residenciais, a solução pode ser utilizada para monitorar entradas e saídas, aumentando a segurança dos moradores ao identificar e registrar visitantes desconhecidos.
6. **Eventos e Conferências:** A aplicação pode ser adaptada para controlar o acesso em eventos e conferências, substituindo os tradicionais crachás de identificação. Isso pode acelerar o processo de *check-in* e aumentar a segurança ao garantir que apenas participantes registrados entrem no evento.

A flexibilidade da solução permite que, com simples ajustes no seu escopo, ela atenda às necessidades de diferentes áreas. Esta versatilidade reflete o potencial de personalização e

a capacidade de integrar-se em diversos ambientes, de forma a garantir uma maior segurança e eficiência no controle de acesso.

2.5 Sumário

Neste capítulo, foi possível adquirir um conhecimento abrangente sobre o estado de arte no domínio do reconhecimento facial e suas diversas técnicas associadas. Através da aplicação da metodologia PRISMA, foram reconhecidos e analisados em detalhe 43 artigos que fornecem informações extremamente relevantes para a implementação da solução proposta.

Diversas técnicas e métodos foram reunidos, os quais são utilizados para implementar soluções de reconhecimento facial nas suas diferentes fases: detecção facial, extração de características e classificação. Cada uma dessas fases possui múltiplos algoritmos que podem ser aplicados. A Figura 33 apresenta um resumo dos principais métodos e algoritmos identificados neste estudo.

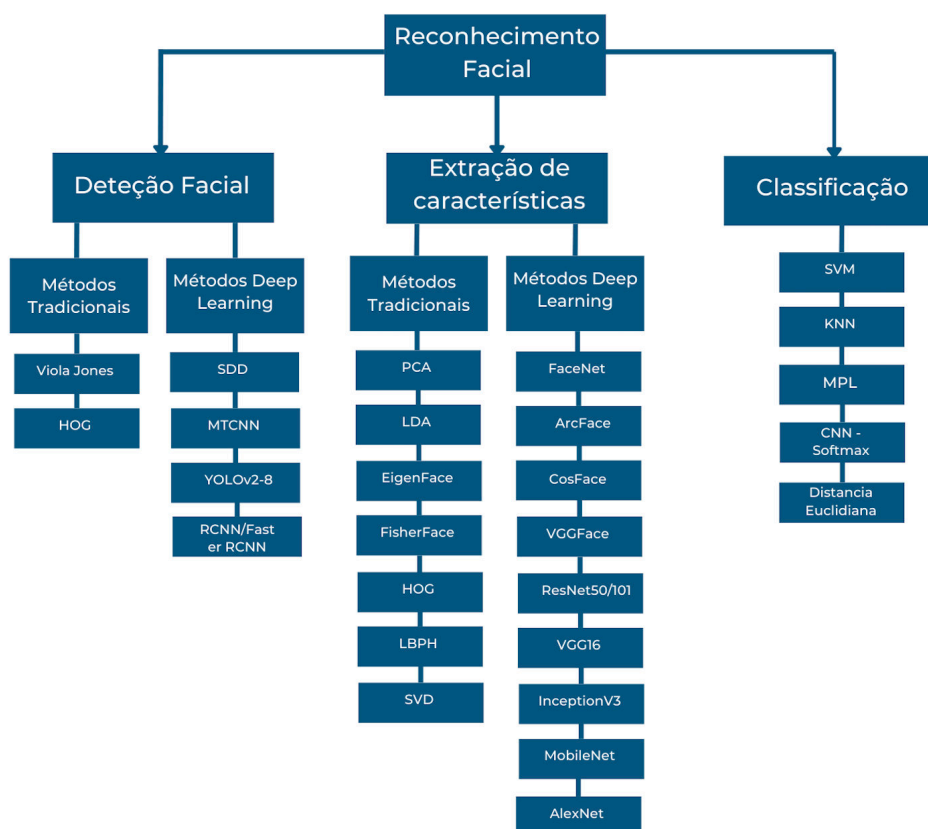


Figura 33 – Resumo do estudo do estado de arte.

Além disso, foi possível compreender como grande parte desses métodos e algoritmos se comporta em situações desfavoráveis, um problema crucial no desenvolvimento de um sistema

de vigilância inteligente, onde frequentemente as imagens não chegam nas melhores condições. Os artigos analisados forneceram *insights* valiosos sobre a diferença entre métodos tradicionais e de *deep learning* nessas condições mencionadas. Concluiu-se que o segredo para elevar a performance do sistema proposto é fornecer uma grande quantidade de dados variados e de alta qualidade.

Notas importantes foram retiradas sobre a necessidade de criar um *dataset* robusto, tanto para detecção quanto para reconhecimento facial. Este *dataset* deve incluir imagens com um elevado nível de rotações, obstruções, diferentes qualidades de resolução e variações de luminosidade, de forma a produzir um sistema extremamente robusto. Isso garantirá que o sistema de vigilância inteligente possa operar eficazmente em diversos cenários, melhorando significativamente sua precisão e confiabilidade. Essa análise detalhada e o reconhecimento da importância de um dataset robusto reforçam a necessidade de investir em um processo de colheita de dados abrangente e diversificado. Com isso, será possível desenvolver um sistema de reconhecimento facial que mantenha alta performance mesmo em condições não ideais, garantindo a eficácia e a eficiência necessárias para aplicações de vigilância e segurança.

Além disso, foi possível reunir artigos que apresentavam abordagens semelhantes e inovadoras, altamente relevantes para o presente trabalho, focando na aplicação de tecnologias de Inteligência Artificial para implementar sistemas de vigilância inteligentes na prática. Embora esses artigos apresentem soluções interessantes, eles estão longe de atender às necessidades que esta tese pretende suprir. Os sistemas discutidos possuem um nível extremamente baixo de personalização, onde oferecem ao utilizador poucas opções de configuração. Na maioria dos casos, são aplicados apenas um método para detecção e outro para reconhecimento facial, sem permitir a escolha ou combinação de diferentes abordagens que possam melhorar a performance em diferentes cenários. Além disso, esses sistemas dependem fortemente de configurações manuais, o que limita a sua escalabilidade e flexibilidade. Falta também qualquer forma de dinamismo, como a geração automática de datasets ou a capacidade de incorporar novos dados de maneira eficiente, dificultando a sua adaptação às necessidades específicas de cada utilizador ou ambiente. A solução proposta nesta dissertação visa superar essas limitações, oferecendo um sistema mais adaptável, automatizado e centrado no utilizador, com a capacidade de personalizar métodos, gerar datasets dinâmicos e reduzir o trabalho manual envolvido na configuração.

No final do capítulo, foram discutidos aspectos cruciais relacionados à privacidade e proteção de dados. Esforços foram feitos para mapear as normas mais importantes do Regulamento Geral sobre a Proteção de Dados (RGPD) em relação à solução proposta.

3 Método e Implementação

Este capítulo tem como objetivo detalhar todo o processo de desenvolvimento e implementação da solução. Encontra-se dividido em três subcapítulos principais, começando por descrever o método utilizado, que aborda as estratégias consideradas, o funcionamento principal da solução e as principais tecnologias e ferramentas utilizadas. De seguida, é apresentada toda a arquitetura de alto nível da solução, nomeadamente a apresentação dos diferentes componentes que interligam o sistema e que são fundamentais para o seu funcionamento. Seguidamente, cada componente é analisado em detalhe, desde as técnicas de Inteligência Artificial até à interface do utilizador. Por fim, é demonstrado um sumário do que foi implementado.

3.1 Método

O projeto consiste no desenvolvimento de uma aplicação móvel que faz recurso de modelos de inteligência artificial para realizar tarefas de deteção facial e reconhecimento de indivíduos, visando criar um sistema de vigilância inteligente. Para tal, foi necessário recorrer a diferentes tipos de modelos de IA para efetuar as operações de deteção e reconhecimento.

Com o estudo realizado no capítulo 2, foi possível concluir que existem inúmeras técnicas de deteção facial, algumas utilizando as mais modernas abordagens de *deep learning*, sendo outras mais primitivas, recorrendo a técnicas tradicionais. Assim, para a implementação deste trabalho, foram escolhidos três diferentes tipos de métodos para abordar o problema de deteção, permitindo uma comparação abrangente entre eles.

Primeiramente, foi selecionado o método tradicional Viola Jones/Haar Cascade. A escolha desta técnica deve-se ao facto de ter sido a técnica tradicional mais abordada nos artigos recolhidos, mostrando ser bastante útil para uma solução prática.

A segunda metodologia emprega o modelo Faster R-CNN, um modelo também abordado na revisão bibliográfica devido à sua arquitetura robusta e desempenho preciso em tarefas de deteção. Em complemento às tecnologias de *deep learning*, a metodologia final utiliza a versão mais recente do modelo YOLO, especificamente o YOLOv8 (vigente até a data de redação deste documento). A escolha destas duas abordagens de *deep learning* deve-se à possibilidade de implementar e analisar de forma detalhada a maneira como ambos os modelos se comportam na solução proposta. Isso permite realizar diversas comparações entre os modelos, uma vez que o Faster R-CNN é uma arquitetura de duas etapas (*dual-stage*) enquanto o YOLOv8 é uma arquitetura de etapa única (*single-stage*). Tal escolha não só foi influenciada pelos artigos revistos, mas também pela oportunidade de conduzir um estudo comparativo sobre as estas diferentes arquiteturas de *deep learning*.

A seleção das técnicas de reconhecimento, espelha a mesma abordagem que foi utilizada para as técnicas de detecção, sendo novamente escolhida uma técnica tradicional e duas técnicas com recurso a *deep learning*. A revisão de literatura evidenciou a preponderância da técnica EigenFaces para a extração de características, bem como a utilização do classificador SVM. Sendo assim, essa combinação de algoritmos foi adotada como a primeira técnica de reconhecimento facial. Em continuidade, e com base em evidências de artigos científicos recolhidos, optou-se pela implementação de um modelo com uma camada *softmax* de *output* que iria realizar a classificação. Por fim, na terceira abordagem, será utilizada uma técnica de extração de características baseada em *triplet loss*, simulando o comportamento do modelo FaceNet[66], que demonstrou possuir bons resultados. Para esta terceira abordagem, a classificação será feita com recurso à distância euclidiana.

É importante salientar que ambos os modelos de *deep learning*, irão fazer curso da rede pré-treinada ResNet50 que irá servir como *backbone*, exigindo ajustes nas camadas de saída para se adequar às necessidades específicas do projeto.

Para fazer recursos destes modelos de IA, foi projetada uma aplicação móvel que irá atuar como interface do utilizador, denominada de iDetect, onde o sistema de vigilância poderá ser configurado e adaptado às necessidades do utilizador. Esta aplicação móvel visa interagir com o sistema inteligente, alertando o utilizador quando existe alguma detecção de um elemento desconhecido, com recurso a um sistema de notificações. A aplicação possui altos níveis de personalização, permitindo a seleção e alteração dos modelos dos diferentes modelos de AI, em tempo real, utilizados para cada uma das tarefas (detecção e reconhecimento). Esta flexibilidade garante que o sistema possa ser adaptado para diferentes cenários de uso e otimizado para um melhor desempenho conforme as necessidades evoluem.

O alerta de um indivíduo desconhecido é gerado com base nos perfis de pessoas que o utilizador autenticou previamente no sistema. Para isso, o utilizador utiliza a aplicação iDetect para adicionar as pessoas ao seu sistema, criando uma base de dados de indivíduos que podem ser reconhecidos e classificados de forma eficiente.

Além disso, o sistema oferece a capacidade de treinar modelos e gerar *datasets* remotamente, facilitando tanto a manutenção quanto a escalabilidade da solução.

O processo de funcionamento da solução, desde a configuração inicial até à detecção e notificação de indivíduos, pode ser visualizado no Fluxograma da solução (Figura 34). Este fluxograma ilustra o fluxo de todo o sistema, começando pela configuração dos diferentes modelos de detecção e reconhecimento, pela conexão da câmara e pela análise das imagens capturadas, até à eventual emissão de alertas caso sejam detetados rostos não autenticados no sistema.

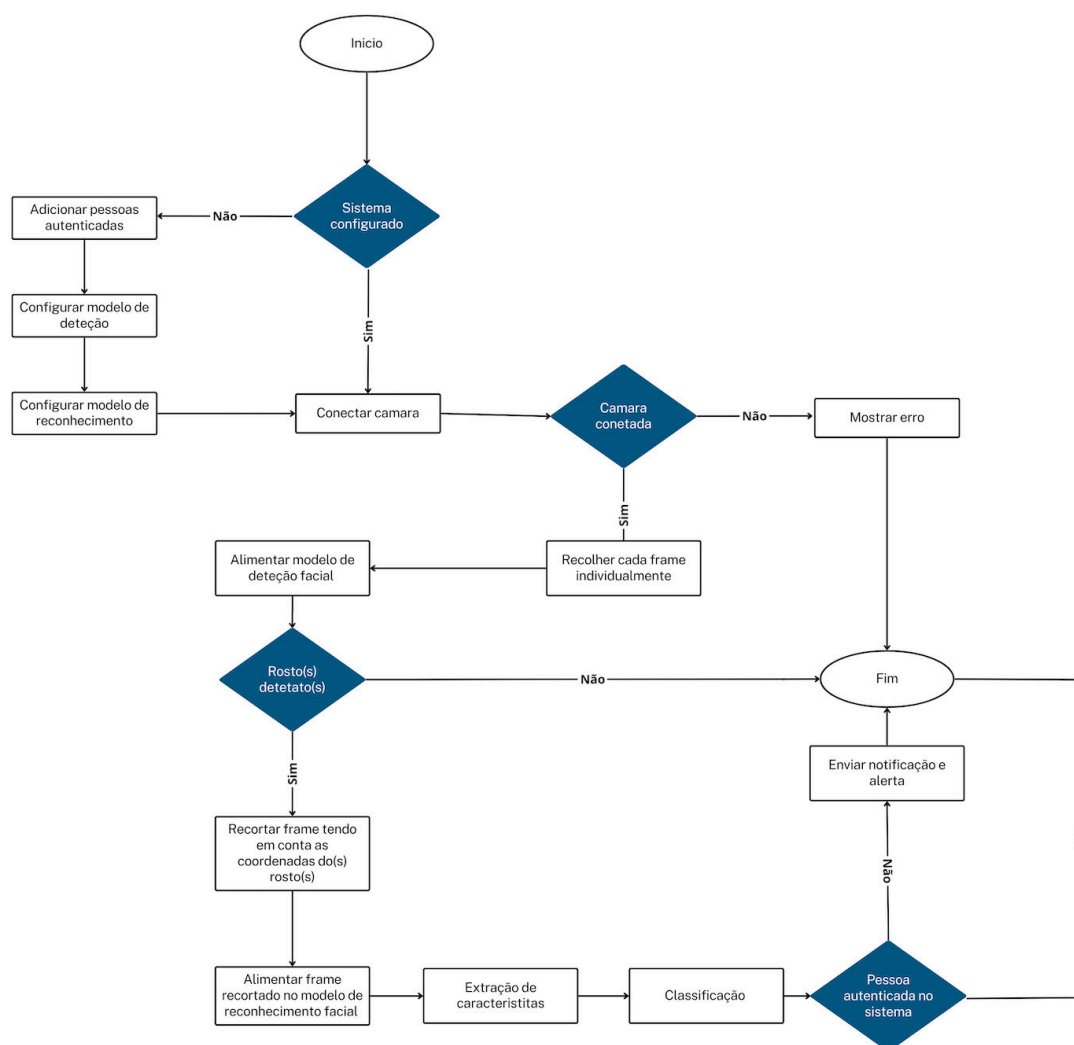


Figura 34 - Fluxograma da solução.

A Figura 34 ilustra de forma clara e sequencial o funcionamento do sistema, desde a configuração inicial até à emissão de alertas, caso sejam detetados rostos não autenticados.

O sistema começa com a etapa de configuração, onde o utilizador precisa de adicionar as pessoas autenticadas e configurar os modelos de deteção e reconhecimento facial. Uma vez que o sistema está configurado, ele tenta estabelecer a conexão com a câmara. Se a câmara não estiver conectada ou falhar na conexão, o sistema apresenta uma mensagem de erro, indicando ao utilizador que algo está errado.

Com a câmara conectada, o sistema começa a recolher *frames* do vídeo, analisando cada *frame* individualmente. O primeiro passo no processamento do frame é alimentar o modelo de deteção facial, que verifica se há rostos presentes na imagem. Se nenhum rosto for detetado, o sistema termina o processamento daquele *frame* e passa ao próximo.

Se um ou mais rostos forem detetados, o sistema recorta o frame com base nas coordenadas do(s) rosto(s). Este recorte é essencial para que a imagem do rosto possa ser analisada de forma mais eficiente no próximo passo.

De seguida, a imagem recortada é então alimentada no modelo de reconhecimento facial. Este modelo extrai características únicas de um determinado rosto, características essas que o sistema irá utilizar para realizar a classificação para determinar se o rosto pertence a uma pessoa autenticada no sistema. Se a pessoa for autenticada, o processo termina e não são gerados alertas, caso contrario, o sistema gera uma notificação e um alerta, informando o utilizador sobre a presença de uma pessoa não autorizada.

Para que o desenvolvimento deste sistema fosse possível, foi necessário recorrer a inúmeras tecnologias e ferramentas, sendo as principais:

Python

Python⁴ é uma linguagem de programação de alto nível, notável pela sua clareza e simplicidade sintática, que facilita a leitura e a escrita do código. Devido a essas características, Python é escolhida como a linguagem predominante para este projeto, destacando-se no desenvolvimento de inteligência artificial, grande parte graças ao seu rico ecossistema de bibliotecas. As principais bibliotecas utilizadas no projeto são:

- **OpenCv**

OpenCV⁵ é a principal biblioteca *open source* para a visão computacional. Através desta biblioteca foi possível implementar os diversos algoritmos e técnicas para realizar algumas das operações de processamento de imagem, nomeadamente deteção e reconhecimento facial.

- **Scikit-Learn**

Scikit-learn⁶ é uma biblioteca que fornece uma coleção de algoritmos para processamento de imagem. Esta biblioteca irá ser utilizada em conjunto com a já mencionada OpenCV para fornecer os algoritmos e métodos necessários para implementar a solução.

- **Jupyter Notebook**

Jupyter Notebook⁷ é uma aplicação *web* que tem como funcionalidade criar e partilhar documentos computacionais. Possui a vantagem de oferecer uma experiência simples, e centrada em documentos. No que toca à solução, todo o código relacionado aos modelos de AI, nomeadamente pré-processamento, treino e testes serão documentados nesta aplicação.

⁴ Welcome to Python.org - <https://www.python.org/>

⁵ OpenCV - Open Computer Vision Library - <https://opencv.org/>

⁶ Scikit-Learn: Machine Learning in Python - <https://scikit-learn.org/stable/>

⁷ Project Jupyter | Home - <https://jupyter.org/>

- **Flask**

No contexto da solução, a biblioteca Flask⁸ irá atuar como API para fornecer o serviço de vídeo em direto. Esta biblioteca terá também o papel de aplicar os diferentes modelos de AI nas imagens que serão recolhidas pelo sistema de vigilância.

- **Tensorflow**

TensorFlow⁹ é uma biblioteca *open source* criada especializada para desenvolvimento de modelos de *deep learning*, computação numérica, entre outras tarefas. Foi desenvolvida pela Google em 2015 e rapidamente se tornou uma das principais ferramentas para *machine learning* e *deep learning* [122]. Esta biblioteca foi essencial para realizar o treino dos diversos modelos associados com os mecanismos de reconhecimento facial.

- **Pytorch**

PyTorch¹⁰ é outra biblioteca *open source* semelhante ao TensorFlow. Desenvolvida inicialmente pelo Laboratório de Inteligência Artificial do Facebook, sendo lançada em 2016 onde rapidamente ganhou uma grande popularidade na comunidade de inteligência artificial [123]. Foi necessário recorrer a esta biblioteca para treinar e implementar os modelos relacionados com o mecanismo de deteção facial.

Flutter

Criado pela Google, Flutter¹¹ é uma *Framework* para o desenvolvimento de aplicações móveis para Android e iOS de forma híbrida. Esta *framework* irá ser utilizada para elaborar a aplicação móvel, ou seja, a interface e utilizador à qual o sistema de vigilância inteligente estará interligado.

.Net

.Net¹² é uma plataforma *open source* para criar aplicações de *desktop*, web e móveis que podem ser executados nativamente em qualquer sistema operacional. Na solução a implementar, esta plataforma irá servir para construir o serviço responsável por gerir todas as informações relativas ao utilizador, nomeadamente dados de autenticação, configurações do sistema e alertas.

Firebase

⁸ Welcome to Flask -

<https://flask.palletsprojects.com/en/3.0.x/>

⁹ Tensorflow - <https://www.tensorflow.org/>

¹⁰ Pytorch - <https://pytorch.org/>

¹¹ Flutter Build apps for any screen -

<https://flutter.dev/>

¹² .Net | Build. Test. Deploy. -

<https://dotnet.microsoft.com/en-us/>

Firebase¹³ é uma plataforma desenvolvida pela Google para a criação de aplicações *web* e móveis. Ela fornece uma variedade de ferramentas e serviços projetados para ajudar desenvolvedores a construir aplicações de alta qualidade. Esta plataforma irá ser fundamental para implementar o mecanismo de notificações, sempre que for necessário enviar uma notificação para o utilizador.

RaspberryPi 4

Para implementar o próprio *hardware* de vigilância será necessário simular uma câmara de vigilância. Essa simulação foi feita com recurso a uma Raspberry Pi¹⁴ 4 com 8 GB de memória RAM.

RaspberryPi High Quality Câmera

De forma a conseguir captar vídeo é necessário utilizar uma câmara que possua boa resolução, neste caso uma câmara de 12.3 MP que fornece imagem a 1080p30, 720p60 ou 640x480p60/90.

GitHub

GitHub¹⁵ é uma plataforma de desenvolvimento colaborativo que aloja projetos na nuvem utilizando o sistema de controle de versões chamado Git. Neste projeto, esta plataforma foi extremamente importante para organizar e gerir o código fonte, garantindo que todas as versões do *software* estavam acessíveis, facilitando a colaboração e o desenvolvimento.

Canva

O Canva¹⁶ é uma ferramenta de design gráfico *online* que foi essencial para a criação dos diagramas de planeamento e arquitetura da solução. Esta ferramenta foi utilizada para visualizar e organizar a estrutura do sistema antes da implementação, de forma a mapear o fluxo de dados e a interação entre os componentes de forma clara.

Computador com alto processamento gráfico

Para que esta implementação fosse possível foi necessário recorrer a um computador com alta performance e processamento gráfico de forma a que se tornasse viável treinar os diferentes modelos em tempo aceitável devido ao facto de os *datasets* de treino possuírem até centenas de milhares de imagens. Sendo assim, o presente trabalho foi realizado numa máquina com as seguintes especificações: I9 13900K, RTX 4080 16GB e 64 GB RAM DDR5.

¹³ Firebase - <https://firebase.google.com/?hl=pt-br>

¹⁴ Raspberry Pi - <https://www.raspberrypi.com/>

¹⁵ GitHub - <https://github.com/>

¹⁶ Canva - <https://www.canva.com/>

3.2 Arquitetura da solução

Para implementar a solução, foi necessário criar diversos componentes separados que funcionavam de forma conjunta para obter os devidos resultados. Sendo assim, foi elaborado um sistema que estava dividido em 8 componentes separados em 5 módulos, nomeadamente:

1. **iDetectApp:** Este componente atua como interface de utilizador da solução. Aqui é feito todo o tipo de iterações com o utilizador, nomeadamente todo o tipo de configurações do sistema, desde adição de pessoas, configuração dos algoritmos, visualização da câmara em tempo real, *logins* e registos.
2. **Raspberry Pi Camera:** Representa o componente de *hardware* do sistema, onde apenas é utilizada uma Raspberry Pi HQ Camera conectada a uma Raspberry Pi 4 transmitindo vídeo em direto via servidor RTPS ¹⁷.
3. **Serviço de informação do utilizador:** Este componente serve como API para os dados do utilizador, nomeadamente para fornecer os serviços de *login*, as configurações do sistema e dispositivos necessários para enviar as notificações e alertas.
4. **Base de dados:** Responsável por armazenar os dados do utilizador, nomeadamente dados de registo, fotos, vídeos, configurações, ids, etc.
5. **Serviço de *Machine Learning*:** Componente central do sistema que fornece o vídeo em tempo real do sistema de vigilância, assim como a aplicação dos diferentes modelos de ML. Responsável também pelo envio das notificações e configurações dos modelos, nomeadamente a geração do *dataset* e do treino de forma remota.
6. **Modulo de Detecção:** Componente que armazena todos os modelos de deteção facial, assim como a *pipeline* de pré-processamento e de treino.
7. **Modelos de Reconhecimento:** Componente que armazena todos os modelos de reconhecimento facial, nomeadamente as técnicas de extração de características e classificação, assim como a *pipeline* de pré-processamento e de treino.
8. **Serviço de notificações:** Componente externo que apenas irá ser utilizado quando é necessário enviar uma notificação para um determinado utilizador com recurso aos dispositivos onde a aplicação está instalada.

Todo o sistema, as suas relações assim como as principais tecnologias e ferramentas utilizadas no desenvolvimento de cada componente estão ilustradas na Figura 35.

¹⁷ Real Time Streaming Protocol - <https://www.ietf.org/rfc/rfc2326.txt>

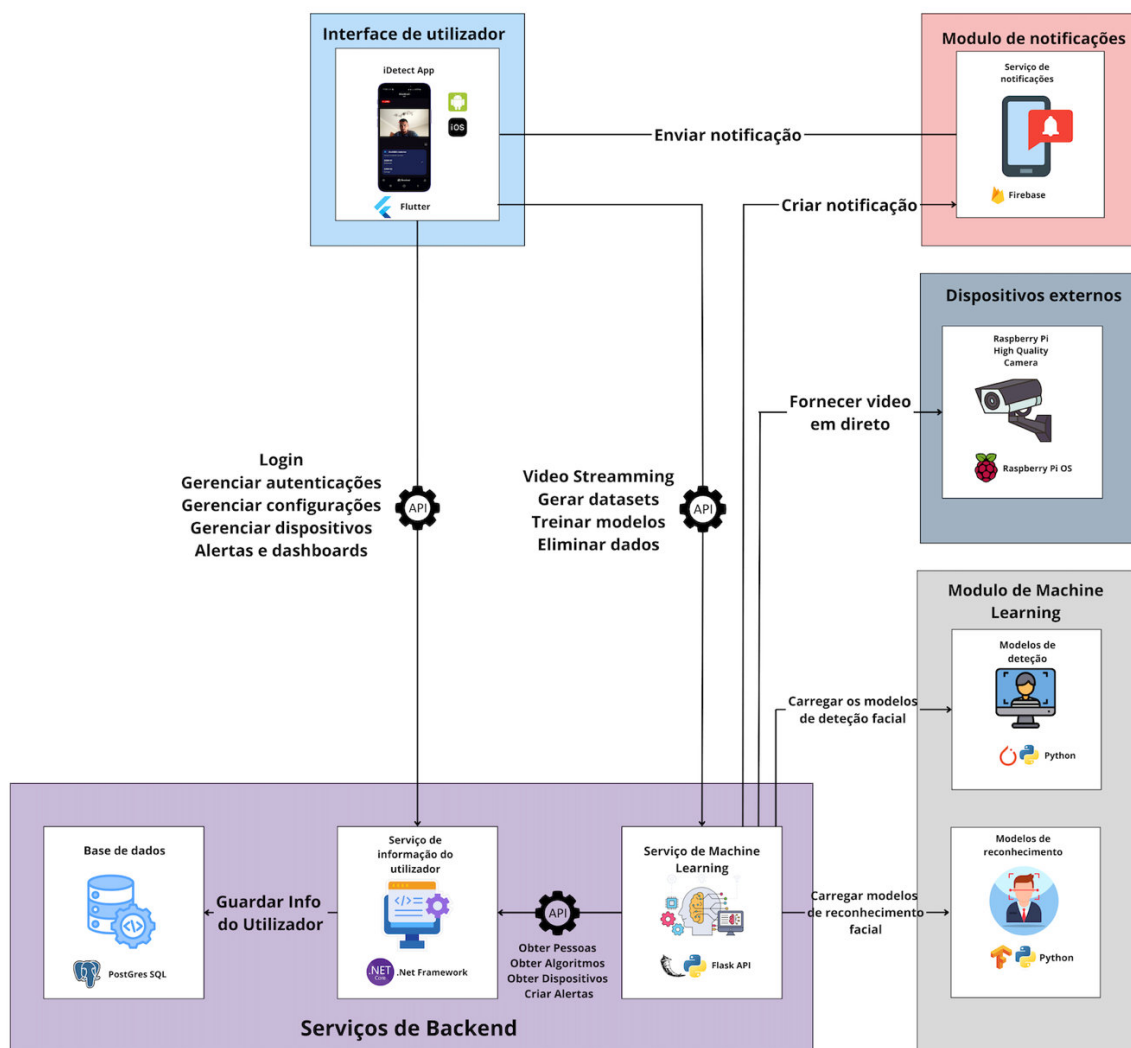


Figura 35 - Arquitetura do Sistema

3.3 Implementação da solução

Neste subcapítulo, será apresentada a implementação detalhada da solução, abordando cada módulo mencionado anteriormente e os seus respetivos componentes. O subcapítulo começa pelo módulo de *machine learning*, onde serão explicados os *datasets* escolhidos, o pré-processamento realizado e a implementação prática dos modelos de deteção e reconhecimento. Em seguida, será abordado o módulo de notificações, seguido do módulo dos serviços de *backend*, onde serão demonstradas as implementações dos dois serviços principais: o serviço do utilizador e o serviço de *machine learning*, incluindo as suas interações. Também será discutido o módulo de *hardware*, com foco na montagem da câmara de vigilância. Por fim, será apresentada a interface do utilizador e todas as suas funcionalidades.

3.3.1 Módulo de Machine Learning

O Módulo de *Machine Learning* desempenha um papel fundamental no sistema, sendo responsável por todas as operações relacionadas com a manipulação de dados e o treino dos diferentes modelos. Neste módulo, os *datasets* são armazenados e submetidos a um rigoroso processo de pré-processamento, garantindo que os dados estão em condições ideais.

É também neste módulo que os modelos de *deep learning* são treinados e, após o treino, armazenados de forma a ficarem disponíveis para serem utilizados pelo sistema em tempo real.

3.3.1.1 Datasets

A solução implementada emprega diversos *datasets* essenciais para o sucesso das operações de deteção e reconhecimento facial. No que toca aos modelos de deteção facial, são necessários *datasets* robustos que contenham uma ampla variedade de rostos em diferentes condições de iluminação, ângulos e expressões, garantindo que o sistema seja capaz de identificar rostos com elevada precisão em situações reais. Da mesma forma, os modelos de reconhecimento facial exigem *datasets* específicos que incluam múltiplas imagens dos mesmos indivíduos, de forma a treinar o sistema para reconhecer e diferenciar identidades com elevada acurácia. A combinação destes *datasets* é fundamental para assegurar que os modelos podem operar de forma eficaz e fiável, atendendo às necessidades complexas da solução proposta.

Sendo assim, para o *dataset* de deteção facial, foi necessário recolher um grande número de imagens de rostos anotados, ou seja, com as coordenadas anotadas de cada rosto, como amostras positivas e um conjunto de fotos sem rostos para compor as amostras negativas.

O *dataset* escolhido para representar as amostras positivas foi o dataset WIDERFace [124], também mencionado no capítulo 2. Este *dataset* contém cerca de 12 000 imagens de rostos disponíveis para treino, abrangendo uma ampla variedade de condições, como diferentes idades, etnias, expressões faciais, poses e condições de iluminação.

O WIDERFace não apenas oferece uma extensa coleção de imagens, mas também fornece um conjunto detalhado de anotações, mais concretamente caixas delimitadoras (bounding boxes), que especificam as coordenadas de cada rosto presente nas imagens.

Estas coordenadas são descritas da seguinte forma:

- **X:** Ponto inicial do eixo dos x;
- **Y:** Ponto inicial do eixo dos y;
- **Width:** Largura da caixa;
- **Height:** Altura da caixa.

A escolha deste dataset foi feita devido à sua abrangência e riqueza de detalhes, que permitem treinar e validar os modelos de *deep learning* de forma eficaz e precisa. As Figura 36 e 37 demonstram visualmente os tipos de dados brutos contidos no *dataset*.



0_Parade_marchingband_1_849.jpg 5_Car_Accident_Accident_5_829.jpg

Figura 36 - Foto com 1 rosto (esquerda) e foto com 3 rostos (direita).

```
0--Parade/0_Parade_marchingband_1_849.jpg
1
449 330 122 149 0 0 0 0 0
5--Car_Accident/5_Car_Accident_Accident_5_829.jpg
3
399 530 39 44 1 0 0 0 0
2 608 18 22 2 0 0 0 1
104 595 13 22 2 0 0 0 1
```

Figura 37 - Exemplo de caixa delimitadoras.

Relativamente às amostras negativas, foram utilizados um conjunto de diferentes tipos de *datasets* que continham imagens sem a presença de rostos humanos. Este conjunto estava dividido pelas seguintes categorias:

- **Animais Domésticos** - Inclui imagens de vários animais de estimação, como cães, gatos e pássaros;
- **Compartimentos exteriores** - Compreende paisagens, exteriores de casas, jardins e outras cenas ao ar livre;
- **Compartimentos interiores** - Inclui imagens de diferentes compartimentos internos, como cozinhas, salas de estar e casas de banho;
- **Objetos** - Contém imagens de uma variedade de objetos, incluindo flores, frutas, alimentos, utensílios e outros itens do quotidiano.

Esta categorização abrangente das amostras negativas é essencial para assegurar que os modelos de reconhecimento facial sejam treinados para distinguir de forma eficaz entre rostos humanos e outros tipos de imagens, aumentando assim a robustez e a precisão do sistema. Com a junção de todos os *sub-datasets* foi possível construir um *dataset* principal composto por cerca de 12000 imagens negativas. Todos os *datasets* de amostras negativas referidos foram recolhidos da plataforma Kaggle¹⁸.

¹⁸ Kaggle: Your Machine Learning and Data Science Community - <https://www.kaggle.com/>

Após garantir a robustez dos modelos de detecção facial com os *datasets* mencionados, a atenção foi direcionada para a construção de *dataset* de reconhecimento facial, que exigiu a utilização de dois tipos distintos destes: um *dataset* público e outro privado, essenciais para o treino e validação dos modelos implementados.

Relativamente ao *dataset* privado, seria necessário recolher rostos das pessoas que seriam identificadas pelo sistema. Sendo assim, este *dataset* seria reduzido podendo auferir de um tamanho variado (num contexto doméstico). O tamanho do *dataset* seria proporcional ao número de pessoas autenticadas no sistema, portanto, quanto maior o número de pessoas autenticadas, maior o *dataset*. Este *dataset* era gerado a partir de um vídeo que era carregado pelo utilizador, no momento da adição de uma pessoa ao sistema, na aplicação movel. O vídeo tinha de ter aproximadamente 15 segundos de duração, onde a pessoa a ser autenticada deveria fazer o maior número de rostos e rotação possíveis nesse determinado espaço de tempo, de modo a gerar o maior número de variações possíveis para essa mesma pessoa. Para além do vídeo, seria necessário identificar o indivíduo para mais tarde o sistema aprender a identificar essa pessoa. Após o *upload* destes dados, um mecanismo de geração de *dataset* é ativado, no qual eram geradas cerca de 100 imagens, correspondendo a 100 *frames* do vídeo anteriormente referido. Nesse mecanismo um modelo de detecção facial era acionado, de maneira a delimitar de forma precisa os rostos, excluindo assim o *background* das imagens que não eram relevantes, com o propósito de evitar um desempenho mais fraco dos modelos de reconhecimento.

No fim, é gerado um *dataset* com 100 imagens distintas de um indivíduo autenticado, organizado por pastas, cada pasta fazendo referência ao nome e sobrenome da pessoa.

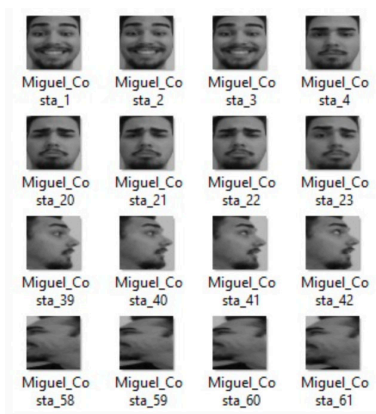


Figura 38 – Dataset privado de reconhecimento facial.

Além do *dataset* privado dos indivíduos a serem identificados, foi necessário recorrer a um *dataset* público de rostos diversos. Este *dataset* público exigia um volume significativamente maior de dados para treinar adequadamente os modelos de *deep learning*. Portanto, foi escolhido o *dataset* público VGGFace2 [125] para esse propósito. O dataset VGGFace2, é composto por 3,3 milhões de imagens divididas entre 9131 indivíduos, oferecendo uma ampla variedade de representações faciais. É importante realçar que não foi utilizado o *dataset*

original, mas sim uma versão melhorada¹⁹, na qual os rostos já se encontravam recortados, excluindo o *background* das imagens que não eram relevantes para o treino.

Para alcançar um equilíbrio entre a eficácia do treino e a viabilidade computacional, o *dataset* não foi utilizado na sua totalidade, devido aos imensos recursos computacionais necessários para processar 3,3 milhões de imagens. Assim sendo, foi necessário reduzir significativamente o tamanho do *dataset*, sendo necessário selecionar um subconjunto do *dataset*. Especificamente, foram escolhidos 6500 indivíduos de forma aleatória, cada um contendo um conjunto de 25 imagens, representando um total de 162000 imagens. Esta redução permitiu manter a diversidade necessária para o treino dos modelos enquanto se reduzia o tempo de treino e a complexidade computacional.

3.3.1.2 Pré-processamento de dados

O pré-processamento de dados é uma etapa crucial para assegurar a qualidade dos modelos de detecção e reconhecimento facial. Este processo envolve uma série de transformações e ajustes que preparam os dados brutos iniciais, recolhidos nos *datasets* mencionados anteriormente, para serem utilizados no treino dos modelos. Começando pelo *dataset* de detecção, foram aplicadas técnicas específicas para garantir que as imagens estivessem em condições ideais para o processo de aprendizagem, permitindo assim que os modelos fossem treinados com dados de alta qualidade. Inicialmente, foram recolhidos aproximadamente 24 000 dados brutos, divididos igualmente entre 12 000 amostras positivas e 12 000 amostras negativas. Em seguida, procedeu-se à divisão destes *datasets*, reservando 80% dos dados para treino e os restantes 20% para validação.

Após a leitura de todas as imagens, foi necessário aplicar técnicas de aumento de dados para aumentar a variedade das imagens, de forma a consolidar a robustez dos modelos de *deep learning*. Sendo assim, tanto no *dataset* de treino como de validação foram aplicados métodos de aumento de dados, como rotações, zooms, escala, ajustes de luminosidade, entre outros, visando aumentar ainda mais a diversidade do dataset. Para cada imagem original, foram criadas 3 versões aumentadas, conforme se verifica na Figura 39.

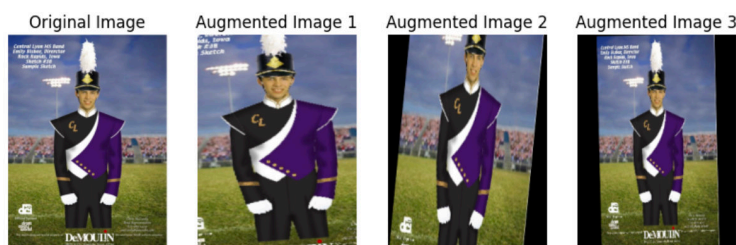


Figura 39 –Imagem original e variações aumentadas para detecção.

¹⁹ VGGFace2 Cropped - <https://www.kaggle.com/datasets/ansarisaqib/vgg-face-2-cropped>

Após aplicar as técnicas de aumento de dados, todas as imagens foram redimensionadas para o tamanho de 640 x 640. Para mais, foram normalizados cada *pixel* individualmente e as respectivas caixas delimitadoras associadas a cada imagem, para valores entre 0 e 1. Além das coordenadas e tamanhos das caixas delimitadoras, foi atribuído um rótulo binário 1 para indicar a presença de um rosto. Para as imagens que não continham rostos, foi utilizado um identificador binário 0, além de coordenadas e valores *placeholder* [0, 0, 0, 0], como mostra a figura 40.

```
Caixa delimitadora normalizada: Amostra Positiva
[0.3671196869441441, 0.15304349710385257, 0.6227391106741769, 0.3690654279324955, 1]
Caixa delimitadora normalizada: Amostra Negativa
[0, 0, 0, 0, 0]
```

Figura 40 - *Bounding Boxes* normalizadas.

Por fim, após realizar todos os processos de pré-processamento, foi criado um *dataset* de treino que continha cerca de 80 000 entradas (10 000 amostras positivas originais + 30 000 amostras positivas aumentadas + 10 000 amostras negativas originais + 30 000 amostras negativas aumentadas), pronto para ser utilizado no treino dos diferentes modelos de detecção. Após todo o processo de pré-processamento eram obtidos os resultados demonstrados na Figura 41.

```
Normalized Bounding Boxes:
[0.005859375, 0.267578125, 0.07421875, 0.38671875, 1]
[0.24609375, 0.25, 0.32421875, 0.376953125, 1]
[0.484375, 0.251953125, 0.568359375, 0.3828125, 1]
[0.720703125, 0.201171875, 0.814453125, 0.33203125, 1]
[0.91796875, 0.166015625, 0.998046875, 0.314453125, 1]
```

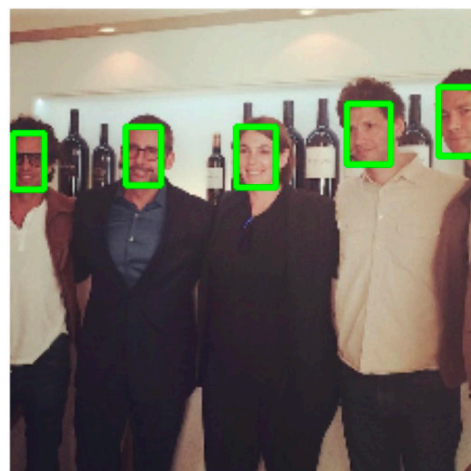


Figura 41 - Resultado da junção da imagem com as respectivas *bboxes*.

A Figura 42 ilustra de forma detalhada todo o fluxo de pré-processamento realizado, desde a recolha dos dados brutos até à criação do *dataset* final de treino, incluindo as etapas de aumento de dados e a geração das *bounding boxes* associadas.

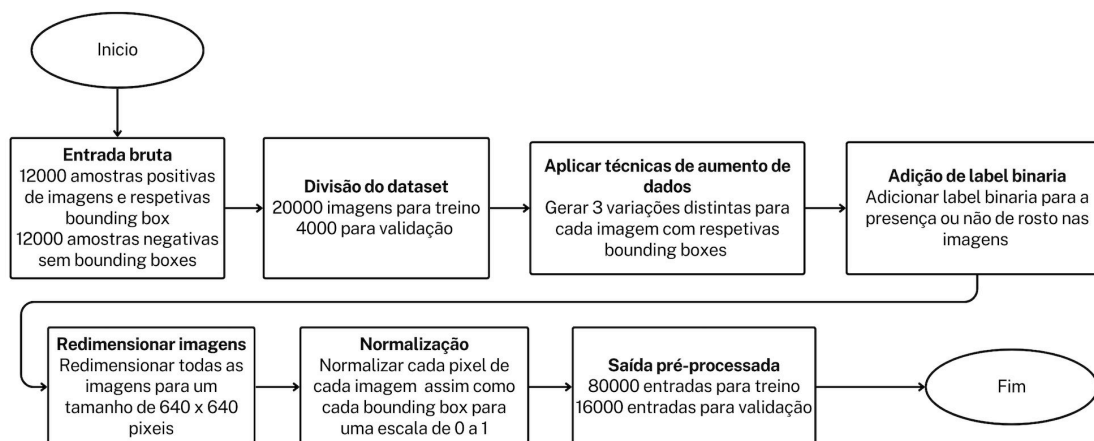


Figura 42 - Pipeline de pré-processamento do dataset detecção.

No que diz respeito ao pré-processamento dos dados para os *datasets* de reconhecimento facial, as metodologias adotadas foram análogas às utilizadas nos *datasets* de detecção facial. As imagens foram também redimensionadas, mas desta vez para dimensões uniformes de 224x224 *pixels*, e todos os *pixels* foram normalizados para um intervalo de 0 a 1, garantindo consistência nos dados de entrada dos modelos.

Quanto às técnicas de aumento de dados, estas foram apenas aplicadas ao *dataset* privado de indivíduos pois o *dataset* publico já possuía uma ampla diversidade nos seus dados. Sendo assim, foi necessária a implementação de abordagens mais agressivas devido à necessidade de gerar uma variedade substancial de amostras a partir de um conjunto limitado, originário de um único vídeo. Logo, foram geradas 5 imagens adicionais, onde foram aplicadas diversas transformações tais como rotações, ajustes de saturação, zoom in, zoom out e esticamentos. Estas manipulações visam enriquecer o *dataset*, aumentando a diversidade e complexidade das amostras, o que contribui significativamente para a robustez dos modelos de reconhecimento facial (Figura 43).

Adicionalmente, foi essencial processar os dados de identificação associados a cada imagem. Os nomes das pessoas identificadas nas imagens foram convertidos para etiquetas codificadas em formato *one-hot*. Este formato é crucial para facilitar o treino dos modelos, permitindo que eles associem de forma eficaz os padrões faciais às respectivas identidades.

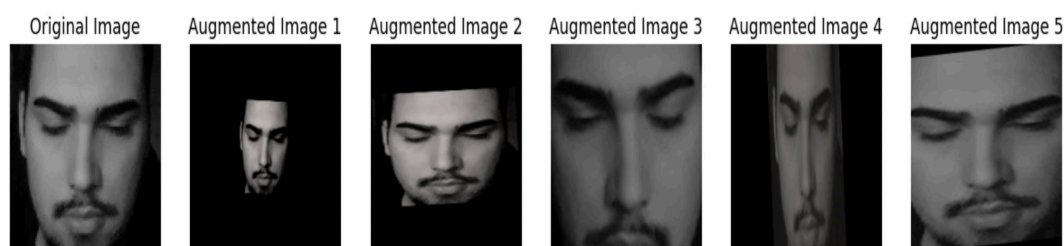


Figura 43 - Versões aumentadas no *dataset* de reconhecimento privado.

Os processos de pré processamento de ambos os *datasets* de reconhecimento encontram se ilustrados nas figuras 44 e 45.

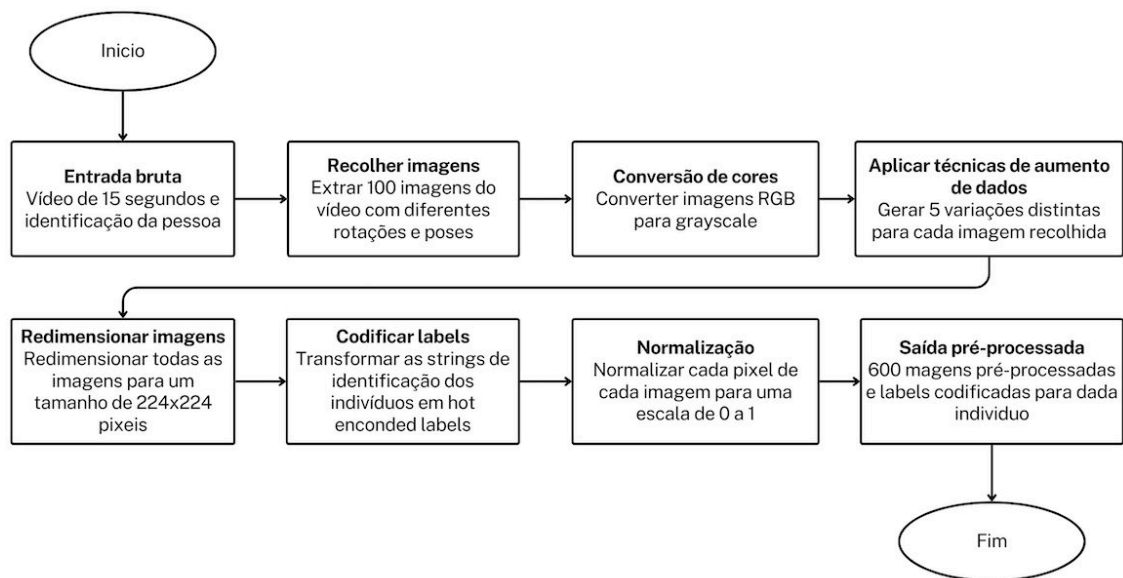


Figura 44 - Pipeline de pré-processamento do *dataset* de reconhecimento privado.

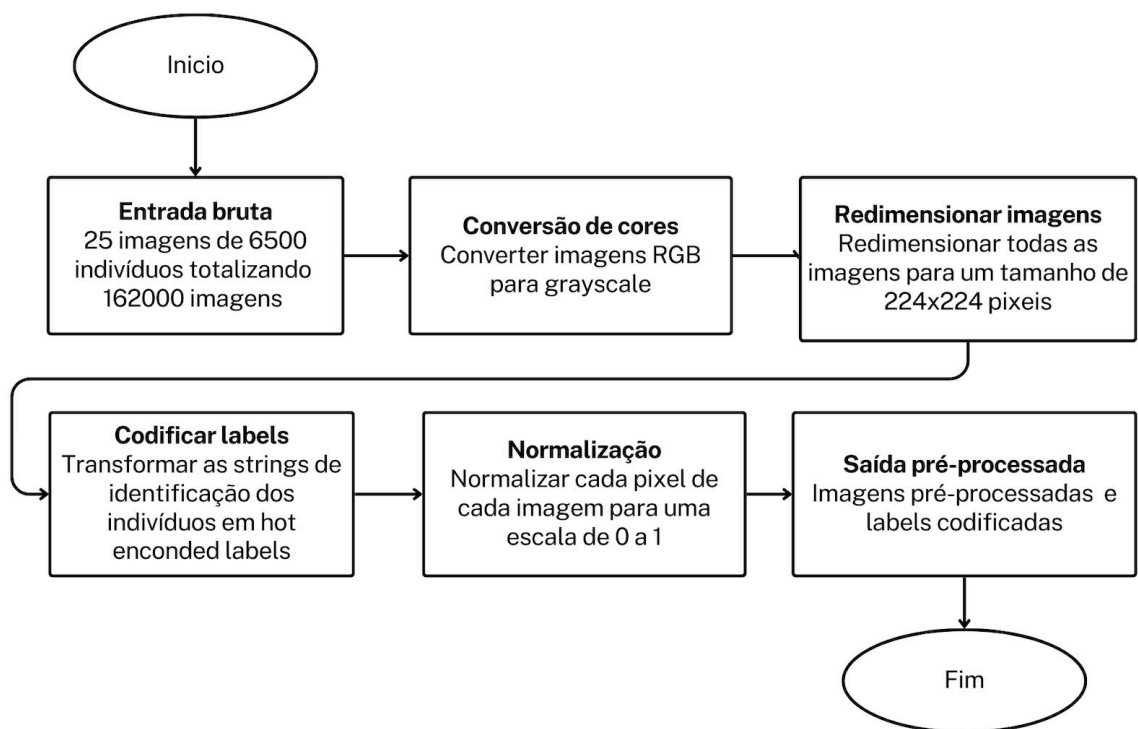


Figura 45 - Pipeline de pré-processamento do *dataset* de reconhecimento publico.

3.3.1.3 Modelos de detecção facial

O uso da técnica Haar Cascade foi relativamente simples e direto de implementar, onde não foi necessário treinar nenhum modelo, pois esta técnica é baseada em classificadores pré-treinados que se encontram disponíveis na biblioteca *OpenCV*. A implementação consistiu simplesmente em utilizar a biblioteca e aplicar os classificadores às imagens, o que facilitou o processo e reduziu o tempo necessário para a configuração inicial.

Os resultados obtidos com o Haar Cascade foram bastante razoáveis em condições ideais, como em imagens com boa iluminação e onde os rostos estavam bem alinhados e de frente para a câmara. No entanto, a performance da técnica diminuiu significativamente em condições mais complexas. Por exemplo, quando as imagens apresentavam variações de iluminação, ângulos de rotação dos rostos ou quando os rostos estavam parcialmente ocultos, este classificador mostrou-se menos eficaz. Nessas situações, por norma, a técnica falhou na deteção dos rostos ou gerou um número elevado de falsos positivos e falsos negativos, gerando BBOXS em posições que não possuíam um rosto.

No que diz respeito aos modelos de *deep learning* utilizados, empregando as suas versões pré-treinadas, foi necessário realizar o *fine-tuning* dos mesmos. Isso deve-se ao facto de ambos os modelos, Faster R-CNN e YOLOv8, não serem originalmente pré-treinados para reconhecer rostos. Assim, o *fine-tuning* foi crucial para adaptar os modelos às necessidades específicas do projeto, permitindo que eles fossem treinados para detetar rostos.

Ambos os modelos funcionavam de formas semelhantes, tendo como *input* uma imagem e como *output* uma ou mais *bounding boxes*, cada uma representando as coordenadas de um rosto humano. A arquitetura desta solução encontra-se ilustrada na figura 46.

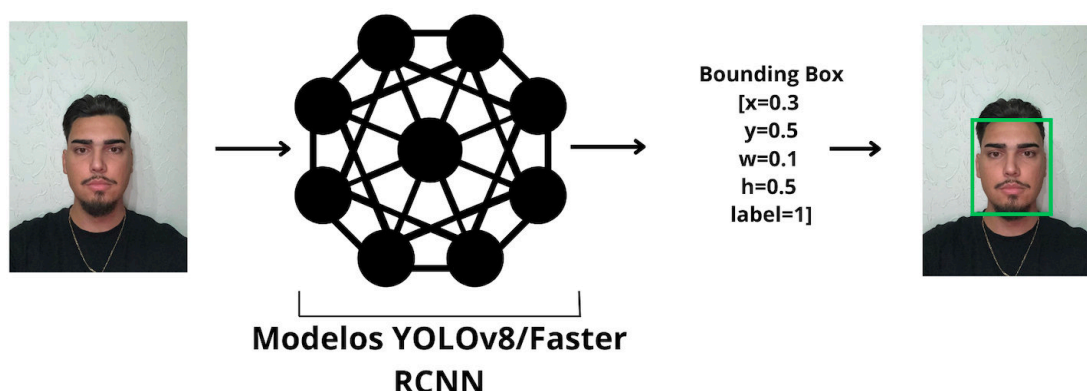


Figura 46 - Arquitetura dos modelos de deteção.

O treino do modelo YOLOv8 foi feito com recurso a biblioteca ultralytics²⁰, na qual estão disponíveis as versões mais recentes deste modelo de deteção.

Para o treino foi necessário ajustar todos os ficheiros correspondentes as imagens e as suas *bounding boxes* no formato específico do modelo YOLO, sendo seleccionadas os seguintes hiper-parâmetros:

- **Batch size:** 64
- **Optimizer:** SGD
- **Learning Rate:** 0.001
- **Nºde épocas:** 75

A *learning rate* foi ajustado automaticamente pela biblioteca ao longo do treino, de forma a garantir melhor convergência do modelo. Adicionalmente, foi implementado um sistema de *callbacks* para guardar o modelo ao final de cada época, permitindo a recuperação e avaliação dos melhores pesos obtidos durante o processo de treino.

Por outro lado, o treino do modelo Faster R-CNN mostrou-se ser mais complexo em comparação com o do YOLOv8, devido à sua arquitetura e requisitos de processamento. Com recurso à biblioteca *TorchVision*, o modelo Faster R-CNN foi carregado com o *backbone* ResNet50. Esta escolha deve-se ao facto de ser um dos *backbones* mais utilizados para este tipo de problema. Para este modelo foram escolhidos os seguintes hiper-parâmetros:

- **Batch size:** 16
- **Optimizer:** SGD
- **Learning Rate:** 0.005
- **Nºde épocas:** 35

Comparativamente ao modelo YOLOv8 o processo de treino foi bem mais pesado, pois todos os dados teriam de ser carregados de forma dinâmica para a memória, com recurso a *data loaders*, sendo assim, o processo de treino extremamente lento comparado ao modelo YOLO.

Os *data loaders* desempenharam um papel crucial na eficiência do treino, permitindo que as imagens fossem carregadas e processadas durante o treino do modelo.

A complexidade adicional do treino do Faster R-CNN se deve à sua arquitetura de duas etapas, onde a primeira etapa envolve a geração de propostas de regiões e a segunda etapa realiza a classificação e refinamento dessas propostas. Esse processo requer mais recursos computacionais e uma gestão eficiente da memória, o que foi alcançado com o uso de *data loaders* e implementação de estratégias de otimização no código.

²⁰ Home - Ultralytics YOLO - <https://docs.ultralytics.com/>

Uma das métricas mais importante neste tipo de implementações é o tempo de treino, este que pode ser observado na tabela 10, que demonstra o tempo de treino das diferentes abordagens escolhidas.

Tabela 10 - Tempo de treino dos modelos de detecção.

Arquitetura	Épocas	Tempo
Haar Cascades	-	-
YOLOv8	75	4h 35m
Faster RCNN	35	26h 43m

É possível destacar uma diferença significativa na eficiência de cada abordagem. Enquanto o YOLOv8 completou 75 épocas em apenas 4 horas e 35 minutos, o treino do modelo Faster R-CNN exigiu 16 horas para completar apenas 35 épocas, mesmo utilizando uma GPU de alta performance, como a RTX 4080 com 16 GB de memória.

Essa disparidade evidencia que o modelo Faster R-CNN é substancialmente mais lento para treinar, devido à sua arquitetura de duas etapas e aos complexos cálculos envolvidos na geração e refinamento das propostas de regiões. O tempo de treino prolongado pode ser um fator limitante em aplicações que requerem ajustes rápidos ou treino frequente com novos dados. Em contrapartida, o YOLOv8, com a sua abordagem de detecção numa única etapa, oferece um treino muito mais rápido, tornando-se uma opção mais viável para cenários que exigem agilidade e eficiência. Este problema não será muito relevante para a solução atual, pois o treino dos modelos de detecção não será feito com muita recorrência em comparação aos modelos de reconhecimento que serão abordados no decorrer deste documento.

A implementação dos modelos YOLO v8 e Faster R-CNN demonstrou a eficácia das duas abordagens distintas nas operações de detecção facial. Enquanto o YOLO v8 se destacou pela simplicidade e rapidez no treino, o modelo Faster R-CNN mostrou a sua robustez e precisão através de uma abordagem mais detalhada e complexa. A combinação dessas metodologias proporcionou uma análise abrangente das capacidades e limitações de cada modelo, contribuindo para a escolha de técnicas mais adequadas conforme os requisitos específicos da aplicação em estudo.

3.3.1.4 Modelos de Reconhecimento Facial

A implementação do método de EigenFaces foi com recurso à técnica de Análise de Componentes Principais, de forma a reduzir a dimensionalidade das imagens faciais, mantendo as características mais relevantes, seguindo os seguintes passos:

1. **Centralização dos Dados:** As imagens foram centralizadas subtraindo-se a média de cada pixel;

2. **Cálculo da Covariância:** Foi calculada a matriz de covariância dos dados;
3. **Eigenfaces:** Os autovetores eigenfaces da matriz de covariância foram computados, representando as direções principais de variação nas faces;
4. **Projeção:** As imagens foram projetadas no espaço formado pelas eigenfaces, resultando numa redução significativa na dimensionalidade dos dados.

Após a extração das características com recurso às EigenFaces, a classificação foi realizada utilizando SVM, devido à sua capacidade de lidar com dados de alta dimensionalidade e encontrar o hiperplano ótimo para separar as diferentes classes.

A implementação desta combinação de métodos provou ser relativamente simples e minimamente eficaz para o reconhecimento facial. Este método não só reduziu a dimensionalidade dos dados, como também manteve as características essenciais para uma futura classificação.

Relativamente ao modelo de *triplet loss*, o seu treino foi dividido em quatro partes principais:

1. Seleção dos *triplets* no *dataset* público;
2. Treino do modelo com os triplets gerados;
3. Geração de vetores *embeddings* do *dataset* privado.

Primeiramente, com recurso ao *dataset* público, foi realizada a seleção de *triplets*, representando conjuntos formados por um exemplo âncora, um exemplo positivo (da mesma classe que a âncora) e um exemplo negativo (de uma classe diferente), demonstrado na figura 47. Esse processo garante que a rede aprenda a aproximar o *embedding* da âncora ao *embedding* positivo, enquanto afasta o *embedding* negativo. Este passo inicial é crucial para definir a base de treino do modelo, pois a qualidade da seleção dos *triplets* impacta diretamente a eficácia da aprendizagem.



Figura 47 – Screenshot de um conjunto de *triplets*.

Com os *triplets* selecionados, estes são então utilizados para alimentar a rede neural. Durante o processo de treino, a rede neural foi ajustada de forma que a distância entre o *embeddings* da âncora e do *triplet* positivo fosse menor do que a distância entre o *embedding* da âncora e do *triplet* negativo. Este processo iterativo, onde a rede ajusta os seus pesos com base na função de perda de *triplet*, é fundamental para a aprendizagem de representações discriminativas robustas.

A rede de *backbone* escolhida para a extração de características foi a ResNet50, reconhecida pela sua eficiência e profundidade na extração de características robustas. Foi retirada a camada de topo, sendo substituída por uma camada de saída com 128 unidades para obter *embeddings* compactos e discriminativos, semelhante ao modelo FaceNet, a Figura 48 ilustra a arquitetura utilizada na solução.

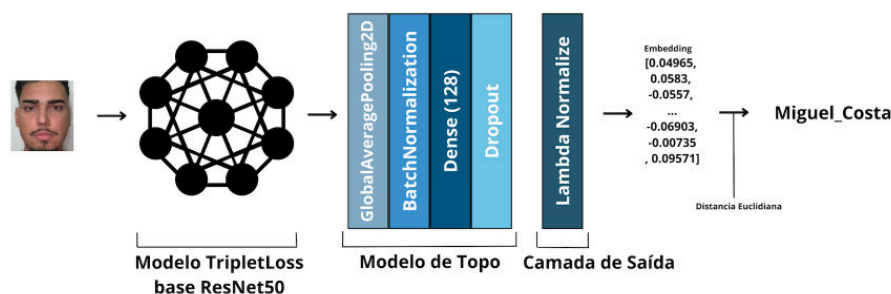


Figura 48 - Arquitetura do modelo *Triplet Loss*.

O treino do modelo foi realizado com a ajuda de *callbacks* que monitoram a *loss* de validação de forma a evitar *overfitting* e garantindo um melhor desempenho do modelo.

Após extensas tentativas, foram escolhidos os melhores hiper-parâmetros para este tipo de modelo, nomeadamente:

- **Batch size:** 64
- **Optimizer:** Adam
- **Learning Rate:** 0.0001
- **Nºde épocas:** 45

Após o treino do modelo, o próximo passo foi gerar os *embeddings* para o *dataset* privado, que continha os rostos das pessoas a serem reconhecidas. Cada imagem do *dataset* foi processada pelo modelo treinado para obter o seu *embedding* correspondente. Esses *embeddings* foram armazenados para uso posterior em tarefas de classificação e reconhecimento.

Como o *output* do modelo *triplet loss* é precisamente um *embedding* que representa a imagem de forma compacta, a classificação consiste em comparar o *embedding* da imagem a ser analisada com os *embeddings* armazenados, utilizando a distância euclidiana como método de classificação.

Um *threshold* de 0,5 foi utilizado para determinar se uma imagem pertence à mesma classe que outra, baseando-se na proximidade dos seus *embeddings*. Se a distância euclidiana entre os *embeddings* for inferior a este valor, as imagens são consideradas da mesma classe, caso contrário, são classificadas como desconhecidos.

Este método de classificação demonstrou ser eficaz na identificação e distinção de indivíduos, tanto no *dataset* privado como na identificação de indivíduos desconhecidos. A escolha do *threshold* de 0.5 foi crucial para balancear a sensibilidade e a especificidade do modelo, o que garantiu que as imagens de indivíduos da mesma classe fossem corretamente agrupadas, enquanto aquelas de classes diferentes fossem adequadamente separadas. Esse processo permitiu a criação de um sistema robusto de reconhecimento, capaz de operar com alta precisão em cenários reais.

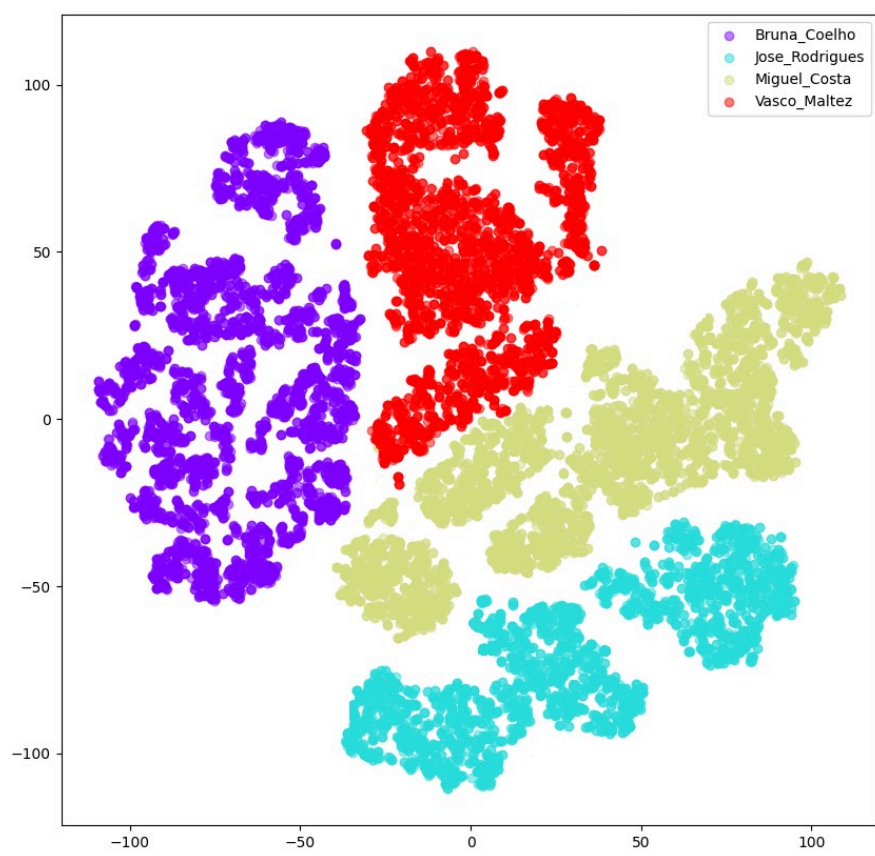


Figura 49 - *Embeddings* gerados pelo *dataset* privado com 4 elementos.

Como é possível observar na Figura 49, que representa um teste com um *dataset* de 4 pessoas, este modelo mostrou-se extremamente preciso na geração de *clusters* de *embeddings*, facilitando a identificação de indivíduos autenticados de indivíduos desconhecidos. A visualização dos *embeddings* gerados revela que os exemplos pertencentes à mesma pessoa estão agrupados próximos uns dos outros, enquanto os exemplos de diferentes pessoas estão bem separados.

Por fim, no que toca ao terceiro e último modelo de reconhecimento facial escolhido, a utilização de um modelo ResNet50 com uma camada de classificação *softmax* mostrou ser uma abordagem mais simples de implementar do que a anterior. Para o treino deste modelo, utilizou-se apenas os dados do *dataset* privado, adicionando uma classe extra para representar indivíduos desconhecidos. Esta classe foi preenchida com uma seleção aleatória de amostras do *dataset* público.

Para adaptar o modelo ResNet50 à tarefa específica, as camadas do topo foram novamente removidas e um novo modelo foi construído sobre o modelo base, com recurso à técnica de aprendizagem por transferência. Este novo modelo incluiu uma camada de saída softmax, responsável por realizar a classificação das imagens. Toda a arquitetura encontra-se ilustrada na Figura 50.

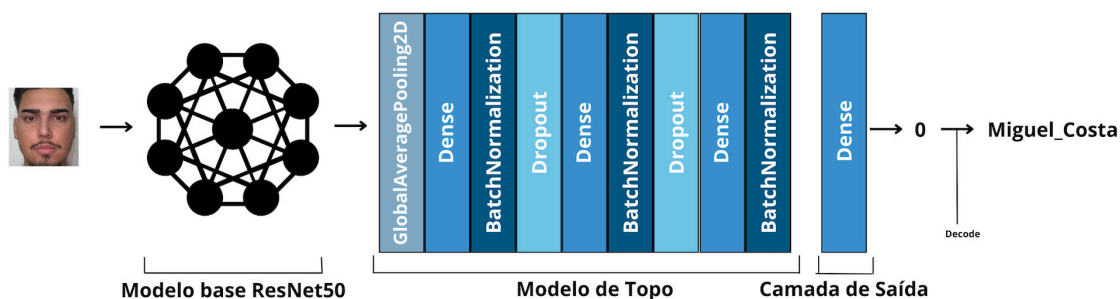


Figura 50 - Arquitetura modelo *Softmax*.

Devido à sua simplicidade, os hiper-parâmetros deferiram do modelo *triplet loss*, sendo eles nomeadamente:

- **Batch size:** 128
- **Optimizer:** Adam
- **Learning Rate:** 0.0001
- **Nº de épocas:** 20 (Para um dataset de 5 classes distintas)

De maneira semelhante aos modelos anteriores, foi implementado um sistema de *callback* para interromper o treino quando a acurácia no *dataset* de validação não melhorasse por 5 épocas consecutivas. Este método garantiu a eficiência do treino, de forma a evitar *overfitting* e economizar o tempo de treino.

Relativamente ao tempo de treino, este é um fator muito importante na solução proposta, pois os diversos modelos tendem a sofrer alterações ao longo do tempo com a adição e remoção de pessoas no sistema. É importante implementar um mecanismo que atue e atualize em tempo considerável e que seja totalmente dinâmico. Na tabela 11 encontra-se os tempos de treino inicial do modelo, assim como o tempo de retreino, ou seja, sempre que é efetuada uma adição de uma pessoa ao sistema de vigilância.

Tabela 11 - Tempo de treino e retreino dos modelos de reconhecimento.

Modelo	Épocas	Tempo treino	Tempo de retreino
Softmax Model	20	>8m	>8m
Triplet Model	45	35h e 28m	<1m

A análise dos tempos de treino e retreino revela uma diferença significativa entre os dois modelos. O modelo *Softmax*, após 20 épocas, requer mais de 8 minutos tanto para o treino inicial quanto para o retreino, isto para um *dataset* com cerca de 5 classes, nomeadamente 4 pessoas reconhecidas e a classe extra que representa as pessoas desconhecidas. Em contraste, o modelo *Triplet Loss*, embora necessite de 45 épocas e um tempo substancial de 35 horas e 28 minutos para o treino inicial, consegue realizar o retreino em menos de 1 minuto.

A principal razão para essa diferença está na abordagem fundamental que cada modelo adota, pois o modelo *Softmax* classifica as imagens diretamente numa das classes predefinidas (pessoas conhecidas) e sempre que uma nova pessoa é adicionada, toda a estrutura de classificação precisa ser ajustada, o que resulta num tempo de retreino considerável, comparável ao tempo de treino inicial. Por outro lado, o modelo *Triplet Loss* aprende a mapear as imagens faciais para um espaço de *embeddings*, onde a proximidade de pontos reflete a similaridade entre os rostos. O treino inicial deste modelo é intensivo e demorado, pois precisa aprender um espaço de representação robusto que generalize bem para novas entradas. No entanto, uma vez que o espaço de *embeddings* está estabelecido, adicionar uma nova pessoa ao sistema é significativamente mais eficiente. O modelo simplesmente calcula e armazena o *embedding* do novo rosto, sem necessidade de reprocessar todo o conjunto de dados, o que explica o tempo de retreino muito reduzido.

Em suma, embora o modelo *Triplet Loss* tenha um custo inicial de treino mais elevado, ele oferece uma vantagem operacional significativa em termos de flexibilidade e eficiência ao lidar com a adição de novas pessoas ao sistema. Esta característica é particularmente valiosa em aplicações práticas, onde a capacidade de integrar rapidamente novos indivíduos é crucial. Portanto, a escolha entre os dois modelos deve considerar o balanço entre o tempo de treino inicial e a eficiência de retreino, com o modelo *Triplet Loss* a destacar-se em cenários dinâmicos e escaláveis.

3.3.2 Módulo de Notificações

O Módulo de Notificações é o componente mais simples da solução, sendo responsável apenas por enviar notificações diretamente para os utilizadores. Este módulo utiliza um serviço externo, o Firebase Cloud Messaging da Google, para distribuir notificações para os diferentes dispositivos móveis registados no sistema. A sua principal função é assegurar que os utilizadores

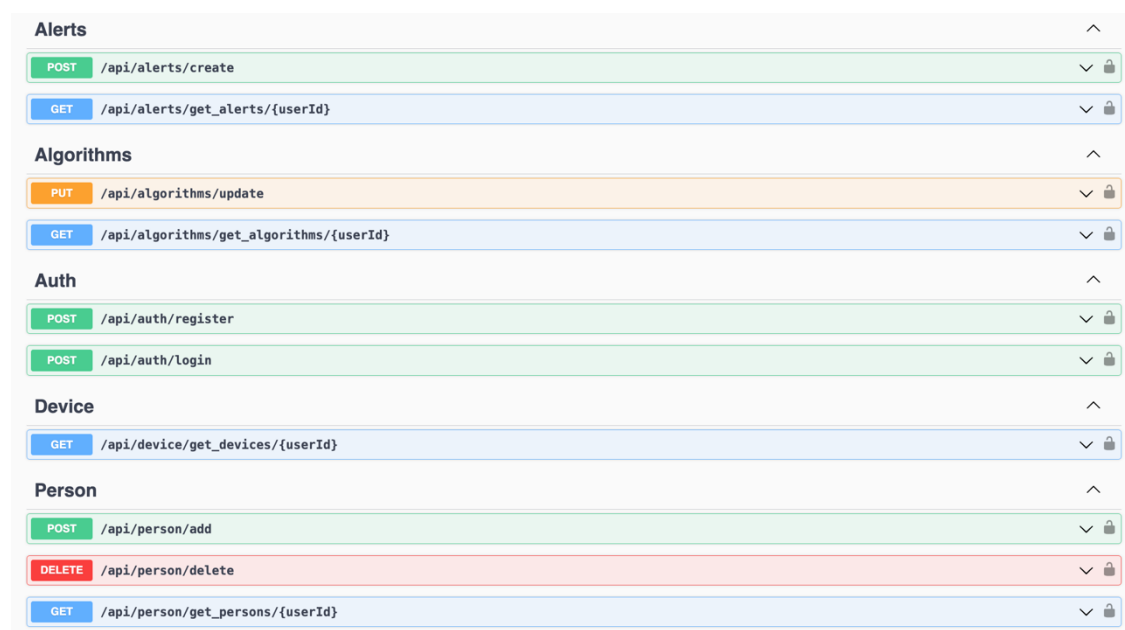
sejam prontamente informados sobre eventos importantes, como a deteção de um desconhecido pelo sistema de vigilância.

3.3.3 Módulo dos Serviços de Backend

O Módulo dos Serviços de *Backend* é a componente central do sistema que integra e coordena todos os serviços *backend* necessários para o funcionamento da aplicação. Este módulo agrega os diversos serviços que comunicam entre si para fornecer uma solução completa. Nele estão incluídos o serviço de dados do utilizador, o serviço de *machine learning* e a base de dados.

3.3.3.1 Serviço de informação do utilizador e base de dados

O serviço de informação do utilizador foi projetado para gerir os dados dos utilizadores, de forma a garantir a segurança, a personalização e o correto funcionamento do sistema de vigilância. Este serviço encontra-se dividido em 5 categorias de *endpoints*, ilustrados na figura 51.



Alerts		^
POST	/api/alerts/create	▼ 🔒
GET	/api/alerts/get_alerts/{userId}	▼ 🔒
Algorithms		^
PUT	/api/algorithms/update	▼ 🔒
GET	/api/algorithms/get_algorithms/{userId}	▼ 🔒
Auth		^
POST	/api/auth/register	▼ 🔒
POST	/api/auth/login	▼ 🔒
Device		^
GET	/api/device/get_devices/{userId}	▼ 🔒
Person		^
POST	/api/person/add	▼ 🔒
DELETE	/api/person/delete	▼ 🔒
GET	/api/person/get_persons/{userId}	▼ 🔒

Figura 51 - *Endpoints* do serviço de informação do utilizador.

1. **Autenticação (Auth):** Esta secção é responsável pela gestão do registo e *login* dos utilizadores na aplicação móvel. É utilizado um sistema de autenticação robusto, baseado em *JSON Web Tokens*²¹, para assegurar que apenas utilizadores autorizados tenham acesso às funcionalidades do sistema. O *endpoint* [POST] /api/auth/register permite o registo de novos utilizadores, enquanto o *endpoint* [POST] /api/auth/login

²¹ Json Web Tokens - <https://jwt.io/>

autêntica os utilizadores existentes, gerando um *token* JWT que será utilizado em todas as interações com a API.

2. **Dispositivos (*Devices*):** Este conjunto de *endpoints* gere os dispositivos associados a cada utilizador, identificando-os através de IDs únicos. Quando é efetuado um login, é acionado um mecanismo que verifica se o ID daquele dispositivo encontra-se na base de dados ou não, caso não exista é criada uma entrada que representa a relação entre o dispositivo e o utilizador.

O *endpoint* [GET] `/api/device/get_devices/{user_id}` é utilizado para obter a lista de dispositivos registados para um determinado utilizador. Os ids dos dispositivos tornam-se essenciais para a gestão de todo o sistema de notificações.

3. **Pessoas (*Persons*):** Neste grupo de endpoints são geridas as pessoas que se encontram autenticadas no sistema de vigilância de cada utilizador. Estas são as pessoas que o sistema de vigilância está programado para reconhecer utilizando os diferentes modelos de reconhecimento. O utilizador poderá adicionar pessoas ao seu sistema, identificando a pessoa em questão, associando também um vídeo de 15 segundos que irá ser utilizado no treino dos modelos de reconhecimento, através do *endpoint* [POST] `/api/person/add`.

O utilizador poderá também, em qualquer momento, remover uma pessoa que já não deseja estar associada ao seu sistema de vigilância através do *endpoint* [DELETE] `/api/person/delete`. Por fim, o utilizador tem acesso à sua lista de pessoas autenticadas com recurso ao *endpoint* [GET] `/api/get_persons/{userId}`.

4. **Algoritmos (*Algorithms*):** Permite ao utilizador configurar as preferências do seu sistema, nomeadamente quais os modelos de deteção e reconhecimento facial a serem utilizados.

O *endpoint* [PUT] `/api/algorithms/update` atualiza as configurações escolhidas pelo utilizador em tempo real, enquanto [GET] `/api/algorithms/get_algorithms/{userId}` recupera as configurações atuais. Estas definições são cruciais para personalizar a operação do sistema de acordo com as necessidades específicas de cada utilizador.

5. **Alertas (*Alerts*):** Por fim, os *endpoints* relacionados com alertas geram todos os alertas após as notificações geradas pelo sistema, nomeadamente quando é detetado um desconhecido. O *endpoint* [POST] `/api/alerts/create` cria um novo alerta na base de dados, que inclui uma foto do momento da deteção e a data a que foi feito o alerta. Por sua vez, o *endpoint* [GET] `/api/alerts/get_alerts/{userId}` permite ao utilizador consultar os alertas gerados na aplicação móvel. Esta funcionalidade é essencial para monitorizar e rever possíveis intrusões ou atividades suspeitas capturadas pelo sistema de vigilância.

3.3.3.2 Serviço de Machine Learning

O Serviço de *Machine Learning* é o núcleo do sistema, encarregado de processar os dados da câmera de videovigilância, aplicar modelos de inteligência artificial para detecção e reconhecimento facial e tomar decisões em tempo real com base nos resultados obtidos. Este serviço, utiliza os algoritmos previamente configurados para analisar as imagens de forma a identificar os rostos conhecidos e desconhecidos. Além disso, este serviço está integrado com outros módulos do sistema, como o Serviço de Notificações e o Serviço de Informação do Utilizador, para garantir que qualquer evento relevante, como a detecção de um intruso, seja comunicado de forma imediata. Na Figura 52 é possível observar a sequência de eventos, desde o início da transmissão do vídeo em direto até à notificação de um indivíduo desconhecido por parte do sistema.

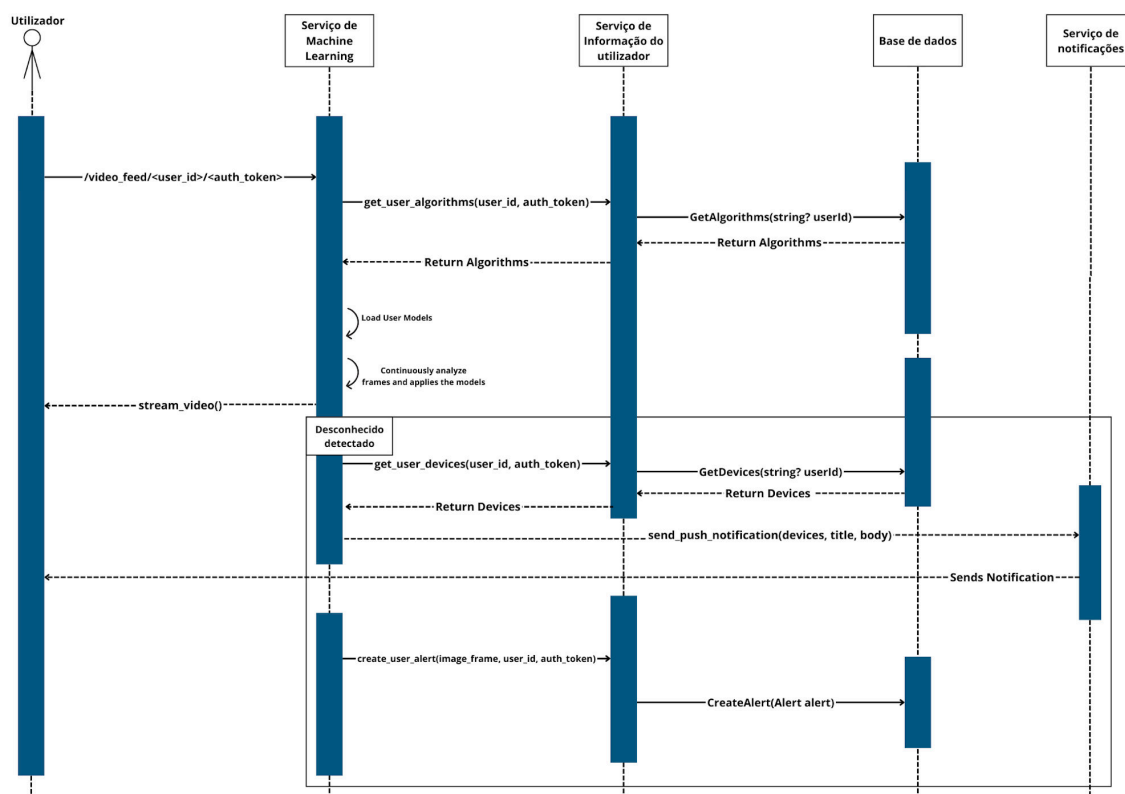


Figura 52 - Diagrama de sequência do processo de notificação.

Para que o utilizador seja notificado, é necessário uma série de mecanismos de comunicação entre diferentes componentes do sistema. Primeiramente, é iniciada a transmissão de vídeo. Esta transmissão é então capturada pelo Serviço de *Machine Learning*, que imediatamente solicita ao Serviço de Informação do Utilizador os algoritmos configurados para aquele utilizador que, por sua vez, consulta a base de dados.

Com os algoritmos obtidos, o Serviço de *Machine Learning* carrega os modelos associados ao utilizador, que está identificado por um *userId* e, de seguida, começa a analisar os *frames* do vídeo em tempo real. Se durante essa análise for detetado um rosto desconhecido, este serviço faz uma nova solicitação ao Serviço de Informação do Utilizador para obter os dispositivos registados do utilizador. Após receber a lista de dispositivos, o Serviço de *Machine Learning* aciona o Serviço de Notificações para enviar uma notificação para esses dispositivos, alertando o utilizador sobre a deteção.

Simultaneamente, o Serviço de *Machine Learning* cria um alerta no sistema, que é armazenado na base de dados. Este alerta inclui informações como a imagem do *frame* onde o desconhecido foi detetado, garantindo que o utilizador tenha acesso a todos os detalhes relevantes do evento. Esse fluxo de comunicação e processamento assegura que o sistema funcione de maneira integrada, o que permite que o utilizador seja notificado de forma rápida sempre que uma potencial ameaça for identificada.

O Serviço de *Machine Learning* possui mais funcionalidades, nomeadamente a geração do *dataset* e treino do modelo de forma automática e remota, garantindo que o sistema se adapta de forma dinâmica e eficiente às mudanças realizadas pelo utilizador. Esse fluxo pode ser observado na Figura 53.

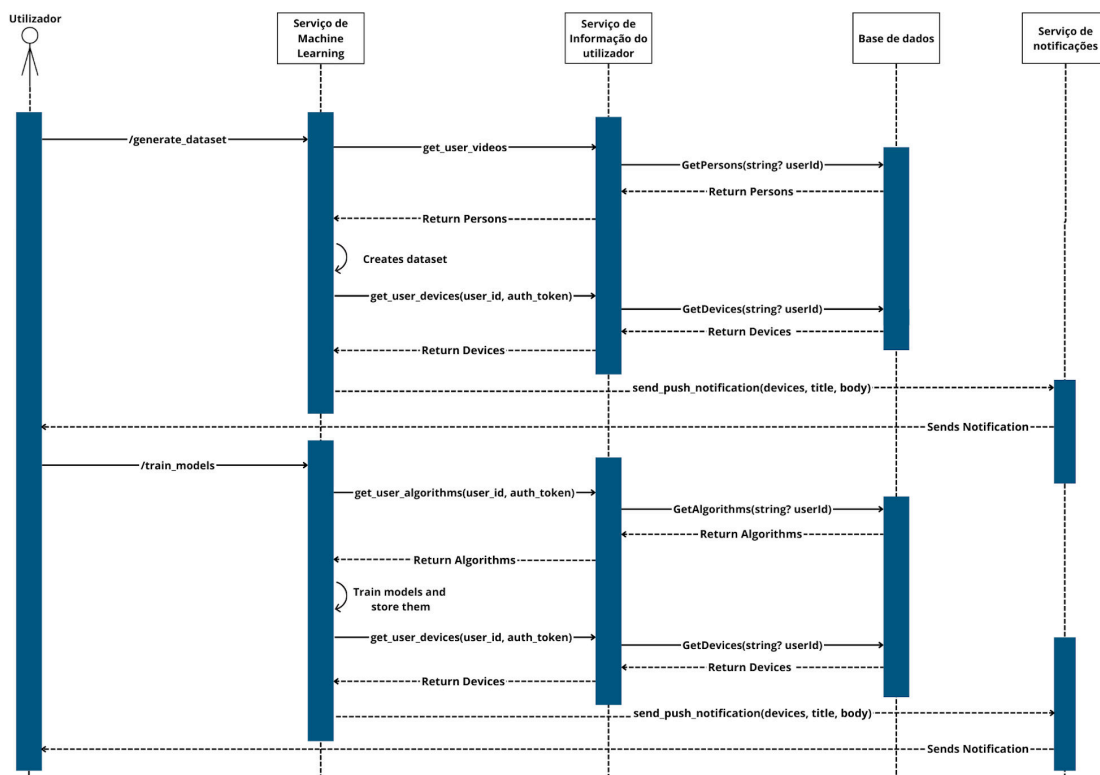


Figura 53 - Diagrama de sequência dos comandos remotos do sistema.

Quando o utilizador adiciona uma nova pessoa ao sistema, o Serviço de *Machine Learning* inicia a geração do *dataset*, comunicando com o Serviço de Informação do Utilizador para obter os vídeos associados à nova pessoa. O Serviço de Informação do Utilizador, por sua vez, consulta a Base de Dados para recuperar as informações necessárias retornando-as ao Serviço de *Machine Learning*, que processa os dados para criar um *dataset* apropriado.

Após a criação do *dataset*, o Serviço de *Machine Learning* comunica novamente com o Serviço de Informação do Utilizador para obter os dispositivos registados pelo utilizador. Esses dispositivos são então notificados através do Serviço de Notificações, informando o utilizador sobre a conclusão da geração do *dataset*. Subsequentemente, o Serviço de *Machine Learning* procede ao treino automático dos modelos com base no *dataset* recém-criado. Novamente, o Serviço de *Machine Learning* comunica com o Serviço de Informação do Utilizador para obter os algoritmos configurados pelo utilizador e, após o treino dos modelos, repete o processo de consulta dos dispositivos registados. Uma vez treinados e armazenados os modelos, o Serviço de *Machine Learning* aciona o Serviço de Notificações para informar o utilizador sobre a conclusão do processo de treino.

Este fluxo automatizado assegura que o sistema se adapta dinamicamente às alterações realizadas pelo utilizador, como a adição/remoção de novas pessoas, otimizando continuamente os modelos de *machine learning* para garantir um reconhecimento facial preciso e eficiente.

3.3.4 Dispositivos externos

Para a captura de vídeo, foi utilizada uma câmara Raspberry Pi HQ, que oferece alta qualidade de imagem, especialmente em ambientes com pouca luz, graças ao seu sensor avançado. Esta câmara está conectada a uma Raspberry Pi com 8GB de RAM, representado na Figura 54. A configuração foi projetada para capturar e transmitir vídeo em tempo real, utilizando o protocolo RTSP. A *stream* RTSP pode ser acedida a partir de qualquer dispositivo na mesma rede local através do endereço `rtsp://<IP-PRIVADO-RASPBERRYPI>:8554/unicast`.

A escolha deste *hardware* permite uma integração eficiente com o Serviço de *Machine Learning*, garantindo que os dados visuais são capturados e processados com a mínima latência, com recurso a biblioteca OpenCv.

Para esta solução, foi utilizada apenas uma câmara devido a limitações de disponibilidade de *hardware*, uma vez que estava disponível apenas uma Raspberry Pi. No entanto, a solução poderia suportar outros tipos de câmaras, desde que estas fossem capazes de transmitir via RTSP ou um protocolo semelhante.

Dado o elevado nível de complexidade do sistema, e considerando as limitações mencionadas, até à data de escrita deste documento, não foi implementado um sistema dinâmico de câmaras que permitisse ao utilizador adicionar múltiplas câmaras e aceder a diferentes compartimentos da sua casa.



Figura 54 - Câmera de videovigilância com recurso à Raspberry Pi.

3.3.5 Interface do utilizador

No que toca à interface do utilizador, este módulo é feito com recurso à aplicação móvel denominada de iDetect ²², que facilita a interação do utilizador com o sistema de vigilância, de forma a permitir a gestão e monitorização das operações do sistema de maneira intuitiva e centralizada.

Inicialmente, o utilizador terá de criar uma conta através de um processo de registo e *login*. Após a autenticação, o utilizador poderá adicionar pessoas ao sistema, geralmente membros da família ou colegas de casa. O processo de adição de pessoas envolve a inserção de informações como nome, sobrenome, uma fotografia e um vídeo de 15 segundos. Este vídeo tem um papel fundamental, pois é utilizado para treinar os modelos de reconhecimento facial. Para mais, a aplicação permite que o utilizador ligue ou desligue o sistema de vigilância com um simples botão, proporcionando um controle fácil e rápido sobre a ativação do sistema. Esta funcionalidade é importante para situações onde o utilizador deseja desativar temporariamente o sistema, como durante reuniões familiares ou eventos sociais.

Ainda relativamente às configurações, a aplicação ainda fornece a capacidade de personalizar a deteção e o reconhecimento facial através da escolha entre diferentes tipos de modelos de inteligência artificial. Existem três opções de modelos tanto para a deteção facial, já referidos, HaarCascades, FasterRCNN e YOLOv8, assim como para o reconhecimento facial, nomeadamente EigenFaces & SVM, *Softmax Model*, *TripletLoss Model*. Esta flexibilidade permite que o utilizador selecione o modelo que melhor se adapta às suas necessidades específicas e ao ambiente monitorado, seja ele uma residência, escritório ou outro tipo de instalação. Os ecrãs que fazem referência à configuração do sistema encontram-se ilustrados na Figura 55.

²² iDetect – Video Surveillance System Powered by AI -
https://www.youtube.com/watch?v=zQdV_a3H8_Q&t=14s

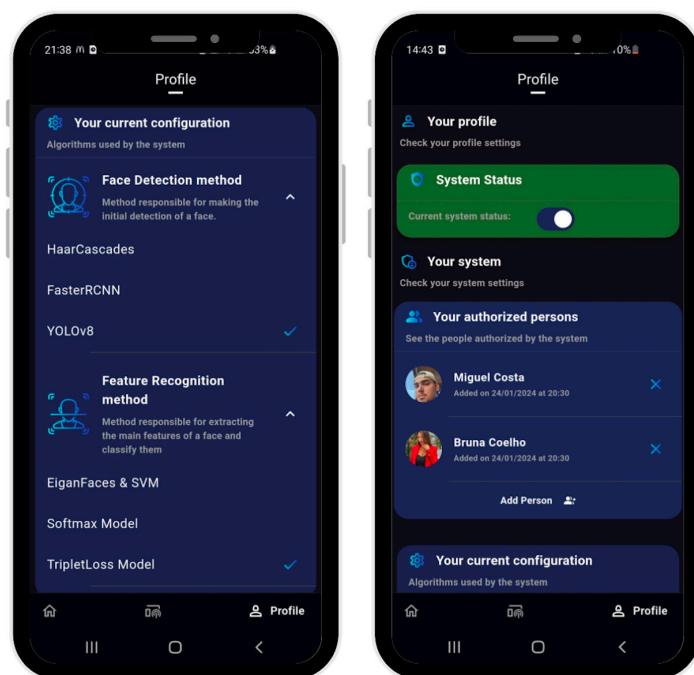


Figura 55 - Configuração do sistema.

Para além da configuração, a aplicação oferece outras funcionalidades avançadas (Figura 56), nomeadamente a realização de várias operações de forma remota. Isso inclui a geração de um *dataset* a partir dos vídeos carregados dos indivíduos autenticados, o treino do modelo de reconhecimento facial de forma remota, assim como a possibilidade de remover todos os dados dos serviços, de forma a respeitar as normas de privacidade dos dados.

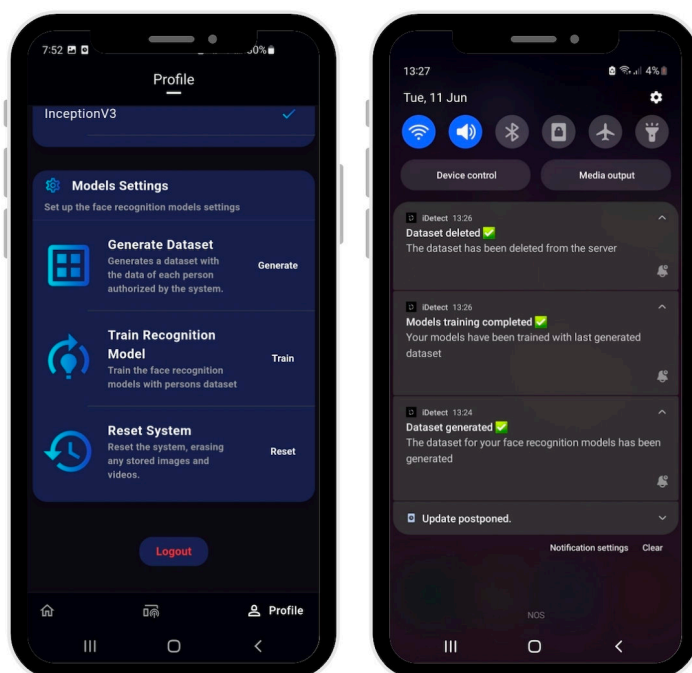


Figura 56 - Configurações avançadas do sistema.

Todas as operações mencionadas contam também com um sistema de *feedback* via notificações, de forma que o utilizador seja notificado que a sua ação foi completada. Este sistema de notificações é essencial, pois todas as operações necessitam de algum tempo de processamento computacional que pode vir a ser demorado.

Relativamente à funcionalidade mais importante, o acesso ao vídeo da câmara em tempo real, esta permite ao utilizador não apenas visualizar as imagens capturadas pelas câmaras, mas também ver os resultados dos *frames* processados pelos diferentes modelos de inteligência artificial integrados no sistema.

Durante este processamento, o sistema realiza a deteção e reconhecimento facial, identificando rostos conhecidos e desconhecidos. Rostos conhecidos, previamente cadastrados no sistema, são destacados em verde, como mostra a Figura 57, permitindo que o utilizador identifique rapidamente indivíduos familiares no campo de visão da câmara. Por outro lado, rostos desconhecidos são destacados a vermelho, de forma a proporcionar um alerta visual imediato sobre possíveis intrusos ou indivíduos não autorizados no ambiente monitorado, também demonstrado na Figura 57.

Além de visualizar o vídeo em tempo real, a aplicação também oferece funcionalidades interativas que melhoram a experiência do utilizador e a eficácia do monitoramento. Entre estas funcionalidades, destaca-se a capacidade de fazer *zoom* nas imagens de vídeo, funcionalidade essa que permite ao utilizador aumentar a visibilidade de áreas específicas da imagem de forma a facilitar a identificação detalhada de rostos de interesse. Esta função é particularmente útil em situações onde é necessário observar de perto determinados detalhes de forma a verificar uma identidade ou monitorar um comportamento específico.

Um dos principais avanços deste sistema é a total integração entre os modelos de deteção e reconhecimento facial e a interface do utilizador, possibilitando que o processamento efetuado pelos modelos seja mostrado em tempo real no ecrã da aplicação. Esta funcionalidade proporciona ao utilizador uma visão clara e instantânea do que está a ser processado pelo sistema, demonstrando diretamente como os diversos modelos interpretam as imagens obtidas. O utilizador tem a capacidade de visualizar em tempo real o instante preciso em que um rosto é detectado e se é identificado ou não, com os resultados apresentados diretamente no vídeo em tempo real.

Esta integração em tempo real é fundamental, na medida em que permite uma experiência de vigilância muito mais envolvente e informativa. O utilizador não precisa de aguardar por relatórios ou estudos futuros para entender o que está a ocorrer no ambiente monitorado, pois todas as informações sobre deteções e reconhecimentos são disponibilizadas de forma imediata. Isso não apenas aprimora a reação a potenciais ameaças, mas também fortalece a confiança do utilizador no sistema, visto que pode acompanhar como os modelos de ML estão a funcionar e a tomar decisões no exato instante dos eventos.

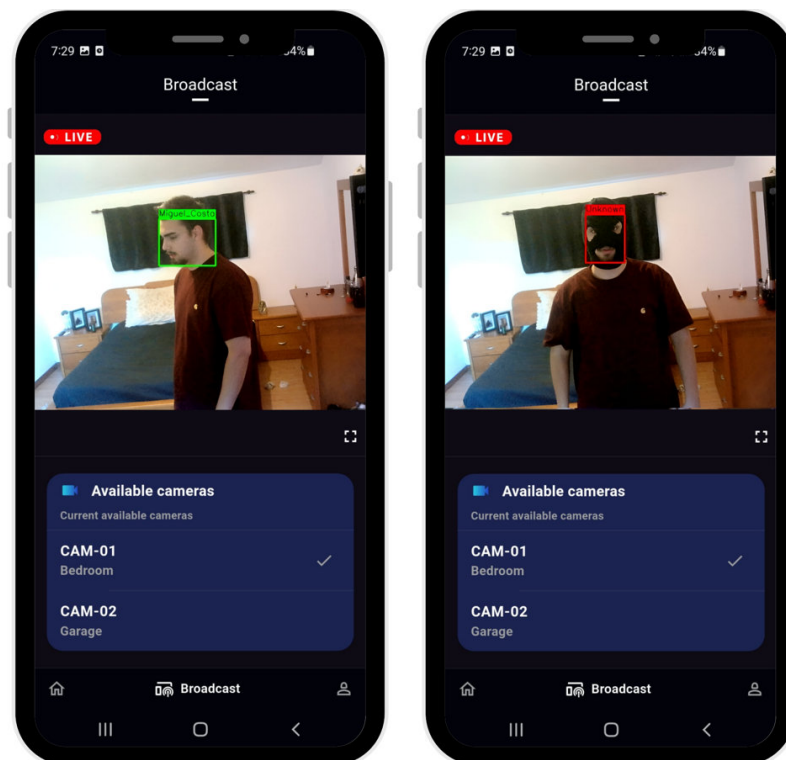


Figura 57 - Sistema de vídeo em direto.

No contexto da deteção de um indivíduo não autenticado, quando este cenário ocorre, o sistema aciona uma série de mecanismos para garantir que o utilizador seja imediatamente informado e possa tomar as devidas decisões de forma rápida e eficiente.

A primeira resposta do sistema é enviar uma notificação (Figura 58) para o utilizador, em todos os dispositivos que possuem a conta do sistema. Este alerta instantâneo é crucial para garantir que o utilizador seja informado em tempo real sobre qualquer possível ameaça.

Simultaneamente ao envio da notificação, o sistema gera um alerta no *dashboard* da aplicação. Este *dashboard* é a central de monitorização onde o utilizador pode visualizar todos os eventos de segurança. Os alertas são organizados cronologicamente e separados por dia, permitindo ao utilizador voltar a ver os eventos da última semana de forma estruturada. Esta organização facilita a análise de padrões e a identificação de incidentes recorrentes.

Além dos alertas textuais, o sistema captura e armazena uma imagem do momento exato da deteção. Estas imagens são guardadas numa galeria dentro da aplicação, também organizada por dia. Esta funcionalidade não só fornece uma evidência visual do evento, mas também permite ao utilizador visualizar as imagens posteriormente para uma análise detalhada. A galeria de imagens serve como um arquivo de referência que pode ser crucial em investigações futuras.

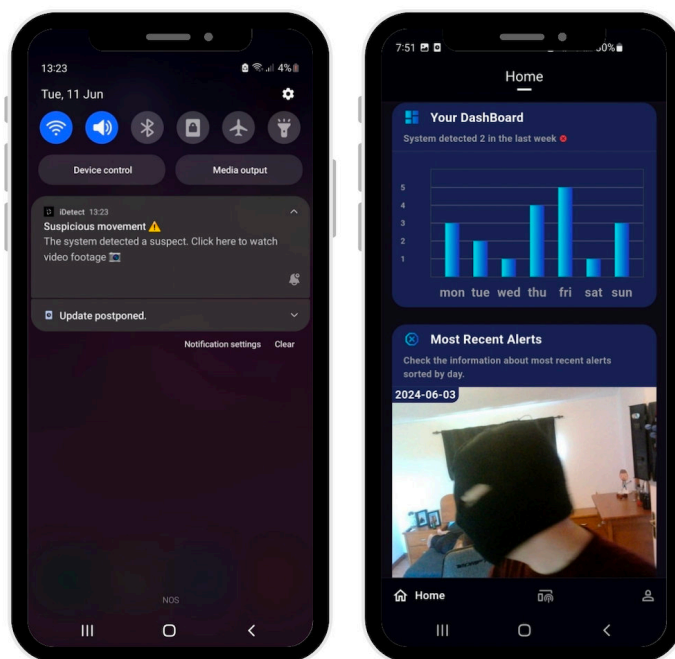


Figura 58 - Notificação de alerta e *dashboard*.

3.4 Sumário

Neste capítulo, foi possível detalhar a implementação do sistema de vigilância inteligente proposto nesta dissertação, composto por diversos componentes interligados, cada um a desempenhar um papel fundamental no funcionamento global do sistema. Desde módulos mais complexos, como o módulo de *Machine Learning*, até partes mais simples, como o módulo de notificações, todas as camadas do sistema foram descritas, onde foram destacadas as comunicações os componentes. Esta arquitetura modular permitiu construir uma solução robusta, capaz de realizar várias funções em simultâneo, oferecendo uma experiência de vigilância eficaz e em tempo real.

A integração dos modelos de inteligência artificial foi a principal conquista desta implementação, que permitiu a aplicação prática de diversas técnicas avançadas de detecção e reconhecimento facial. A implementação demonstrou como estes modelos podem ser utilizados de forma eficiente numa solução real, que processa os dados em tempo real.

Além disso, a criação de uma aplicação móvel foi essencial para tornar a experiência do utilizador mais acessível e interativa. Através desta aplicação, foi possível visualizar o funcionamento dos modelos de IA em tempo real, oferecendo uma interface intuitiva que exhibe os resultados diretamente num dispositivo móvel. Essa funcionalidade não só reforçou a utilidade prática do sistema, mas também mostrou como a IA pode ser aplicada de forma prática e eficiente, melhorando significativamente os sistemas de videovigilância tradicionais.

4 Avaliação e Experimentação

Neste capítulo, irá ser abordada a metodologia de avaliação e experimentação dos algoritmos utilizados na implementação. O capítulo será dividido na metodologia empregada para efetuar a avaliação, que se encontra estruturada entre os mecanismos de detecção e reconhecimento facial. No final, os resultados serão analisados e as respectivas conclusões serão retiradas.

4.1 Metodologia

Para testar a solução como um todo, é necessário dividir a experimentação e avaliação dos modelos em diferentes tópicos. O primeiro tópico refere-se à avaliação dos modelos e mecanismos de detecção facial, e o segundo aos modelos e mecanismos de reconhecimento facial.

Para avaliar as diferentes técnicas de detecção facial implementadas neste projeto, foram novamente escolhidas amostras do *dataset* WIDERFace, composto por cerca de 3600 imagens que não foram utilizadas durante o processo de treino. A escolha deve-se ao facto deste *dataset* possuir um elevado nível de diversidade e principalmente dificuldade. Este conjunto de dados foi o ideal para testar a robustez e eficácia dos algoritmos de detecção facial.

Para a avaliação dos modelos de reconhecimento facial, foi utilizado um *dataset* de teste personalizado. Este processo envolveu a recolha de vídeos adicionais, com cerca de 15 segundos cada, de 4 indivíduos (os mesmos indivíduos mencionados no capítulo anterior que foram utilizados no treino dos modelos), de forma a gerar, desta vez, 200 frames cada um, totalizando cerca de 800 imagens de pessoas conhecidas. Além das 800 amostras positivas, foram também utilizadas 800 amostras negativas do *dataset* VGGFACE2 (que não foram utilizadas no treino), contabilizando um total de 1600 imagens de teste. A inclusão de amostras negativas visa testar a capacidade do modelo em distinguir entre faces conhecidas e desconhecidas, avaliando assim a precisão e a taxa de falsos positivos.

Além dos *datasets* de teste, foram realizados testes extensivos com vídeo em direto. Este aspeto é particularmente relevante, uma vez que o principal foco da tese é avaliar como estes algoritmos reagem em cenários de vídeo em direto e em termos práticos. Estes testes permitem observar o desempenho dos algoritmos em condições dinâmicas e não controladas, proporcionando uma avaliação mais realista da sua eficácia em aplicações do mundo real. Durante os testes com vídeo em direto, foram considerados vários fatores, como a latência, a precisão da detecção e reconhecimento em tempo real, e a capacidade do sistema em lidar com variações de iluminação e movimento, assim como a própria qualidade do vídeo, nomeadamente a taxa de quadros por segundos (FPS).

4.2 Detecção Facial

Na avaliação de modelos de deteção facial, várias métricas são utilizadas para quantificar a precisão e a eficácia do modelo na tarefa de identificar e localizar rostos em imagens. Para a presente solução foram utilizadas as seguintes métricas:

1. **Precision:** A precisão mede a proporção de verdadeiros positivos (deteções corretas de rostos) em relação ao total de detecções feitas pelo modelo (verdadeiros positivos + falsos positivos).

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

2. **Recall:** Mede a proporção de verdadeiros positivos em relação ao total de instâncias reais de rostos (verdadeiros positivos + falsos negativos), indicando a capacidade do modelo encontrar todos os rostos presentes nas diversas imagens.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

3. **Intersection over Union:** Métrica responsável por medir a sobreposição entre a caixa prevista e a caixa real. A IoU foi escolhida por proporcionar resultados mais precisos e robustos na avaliação da performance dos modelos.

$$IoU = \frac{area(B_{original} \cap B_{predicted})}{area(B_{original} \cup B_{predicted})}$$

4. **mAp@0.5:** A Precisão Média é uma métrica que avalia a performance do modelo em diferentes limiares de IoU. A mAP@0.5, por exemplo, calcula a média da precisão em um limiar de IoU de 0.5, proporcionando uma visão mais ampla da eficácia do modelo em detetar rostos com diferentes níveis de sobreposição.

5. **Tempo:** O tempo de execução mede a eficiência do modelo em termos de tempo necessário para processar uma imagem ou um conjunto de imagens. Esta métrica é crucial em aplicações em tempo real, onde a rapidez na detecção é essencial.

As escolhas das métricas mencionadas foram fundamentadas em artigos reconhecidos e discutidos no capítulo da bibliografia desta dissertação. Estes artigos fornecem uma base sólida para a avaliação de modelos de deteção facial, garantindo que as técnicas utilizadas são as mais eficazes e validadas pela comunidade científica.

4.2.1 Validação e experimentação dos modelos de detecção.

Após serem realizados os testes no *dataset* de teste, foram obtidos os resultados presentes na Tabela 12.

Tabela 12 - Performance dos modelos de detecção.

Arquitetura	Precision	Recall	IoU	mAP@0.5	Time (s)
Haar Cascades	0.48	0.37	0.46	0.37	-
YOLOv8	0.91	0.86	0.78	0.87	45.66
Faster RCNN	0.93	0.88	0.81	0.89	140.95

Para mais, foi também feito um estudo mais detalhado de como estes métodos reagem para a mesma imagem, de forma a ter uma ideia mais visual do potencial de cada um e comparar os devidos resultados. Na Figura 59 é possível observar o comportamento de cada método em 2 imagens de testes, nomeadamente uma imagem que possui rostos de extrema dificuldade de detecção, onde grande parte destes estão em posições de rotação difíceis de detetar, e outra imagem na qual são apresentados rostos de forma frontal, representando um cenário mais simples de detecção.



Figura 59 - Testes dos modelos de detecção facial.

Após a realização dos testes dos diferentes modelos de detecção, é possível afirmar que cada um apresenta resultados distintos que merecem uma análise detalhada.

Os resultados inferiores do HaarCascade já eram esperados com base na revisão de literatura e no estudo detalhado realizado no capítulo 2 desta investigação. As principais limitações deste método incluem:

- **Precisão:** Menor precisão na detecção facial, especialmente em condições de iluminação variada, ângulos diferentes e rotações faciais adversas;
- **Robustez:** Menor robustez contra oclusões e expressões faciais variadas.

No entanto, é importante destacar alguns aspectos positivos do HaarCascade:

- **Facilidade de Implementação:** Este método é relativamente fácil de implementar, sendo ideal para a criação de Provas de Conceito (POC) que necessitem de reconhecimento facial básico;
- **Leveza Computacional:** Extremamente leve em termos de recursos computacionais, podendo ser executado eficientemente em máquinas com apenas uma CPU, sem necessidade de GPU.

Na análise dos modelos de *deep learning* utilizados para a detecção facial, ambos os modelos, YOLOv8 e Faster RCNN, apresentaram resultados notavelmente bons, com uma leve vantagem do modelo Faster RCNN na maioria das métricas de teste.

Os testes revelaram que ambos os modelos são capazes de reconhecer rostos parciais ou totalmente obstruídos. Isso significa que, embora não seja possível reconhecer um rosto completamente coberto, é possível efetuar a sua detecção. Este fator é extremamente importante para o funcionamento e segurança do sistema pois, sendo assim, o sistema é capaz de detetar rostos mesmo quando estão completamente tapados, resultando numa classificação como pessoa desconhecida, alertando futuramente o utilizador de um comportamento suspeito.

O modelo Fast RCNN destacou-se em situações desfavoráveis para a detecção facial, como rostos extremamente longe, com baixa qualidade e luminosidade. Apesar da sua superioridade em precisão e *recall*, o modelo Faster RCNN é consideravelmente mais lento, levando aproximadamente três vezes mais tempo para processar as imagens comparado ao modelo YOLOv8.

O modelo YOLOv8, apesar de apresentar métricas ligeiramente inferiores em comparação ao Faster RCNN, ainda demonstra uma elevada precisão nas operações de detecção facial, identificando a grande maioria dos rostos nas imagens, exceto aqueles em condições extremamente desfavoráveis. Num contexto doméstico, onde o sistema de vigilância estará instalado em posições estratégicas, este modelo não enfrentará dificuldades para realizar o reconhecimento, uma vez que as condições extremas testadas no subcapítulo anterior não se aplicarão na prática.

Durante os testes práticos com a aplicação completa, ficou evidente a velocidade mais lenta do modelo Faster RCNN, o que comprometeu a performance da transmissão de vídeo. Isso resultou numa baixa taxa de quadros por segundo e, conseqüentemente, numa experiência de utilizador inferior.

Em suma, a escolha do modelo deve ser feita com base no contexto da aplicação. Para sistemas de vigilância domésticos, onde a velocidade e a fluidez são cruciais, o YOLOv8 é recomendado. Já para aplicações que requerem alta precisão e robustez, mesmo em condições adversas, o Faster RCNN pode ser mais adequado, apesar da sua maior necessidade por recursos computacionais. Por último, o algoritmo HaarCascade pode ser utilizado em aplicações onde existam limitações a nível de *hardware* e os cenários de deteção não são complexos, apresentando ambientes favoráveis.

4.3 Reconhecimento Facial

Para avaliar os diversos modelos de reconhecimento facial foram utilizadas algumas métricas já abordadas no capítulo 4.1, nomeadamente a métrica de precisão, *recall* e tempo, sendo também utilizadas métricas tais como:

- **Acurácia:** A acurácia mede a proporção de todas as previsões corretas (verdadeiros positivos + verdadeiros negativos) em relação ao total de previsões (verdadeiros positivos + verdadeiros negativos + falsos positivos + falsos negativos).

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}$$

- **F1-Score:** O F1 Score é a média harmônica de precisão e *recall*. Esta métrica é útil para avaliar a performance de um modelo quando se procura um equilíbrio entre precisão e sensibilidade.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4.3.1 Validação dos modelos de reconhecimento

Após os testes realizados com o *dataset* referido, o resultado da performance dos diferentes modelos encontra-se disponível na Tabela 13. Também, é possível observar o resultado dos diferentes modelos para a mesma imagem na Figura 60, na qual inclui 2 indivíduos autenticados (Miguel Costa e Bruna Coelho), rosto totalmente obstruído e um indivíduo não autenticado.

Tabela 13 - Performance dos modelos de reconhecimento.

Arquitetura	Accuracy	Precision	Recall	F1- Score	Time (s)
EiganFaces+SVM	0.59	0.63	0.65	0.64	1.64
Softmax model	0.93	0.90	0.92	0.91	3.03
TripletLoss Model	0.97	0.94	0.96	0.95	4.34

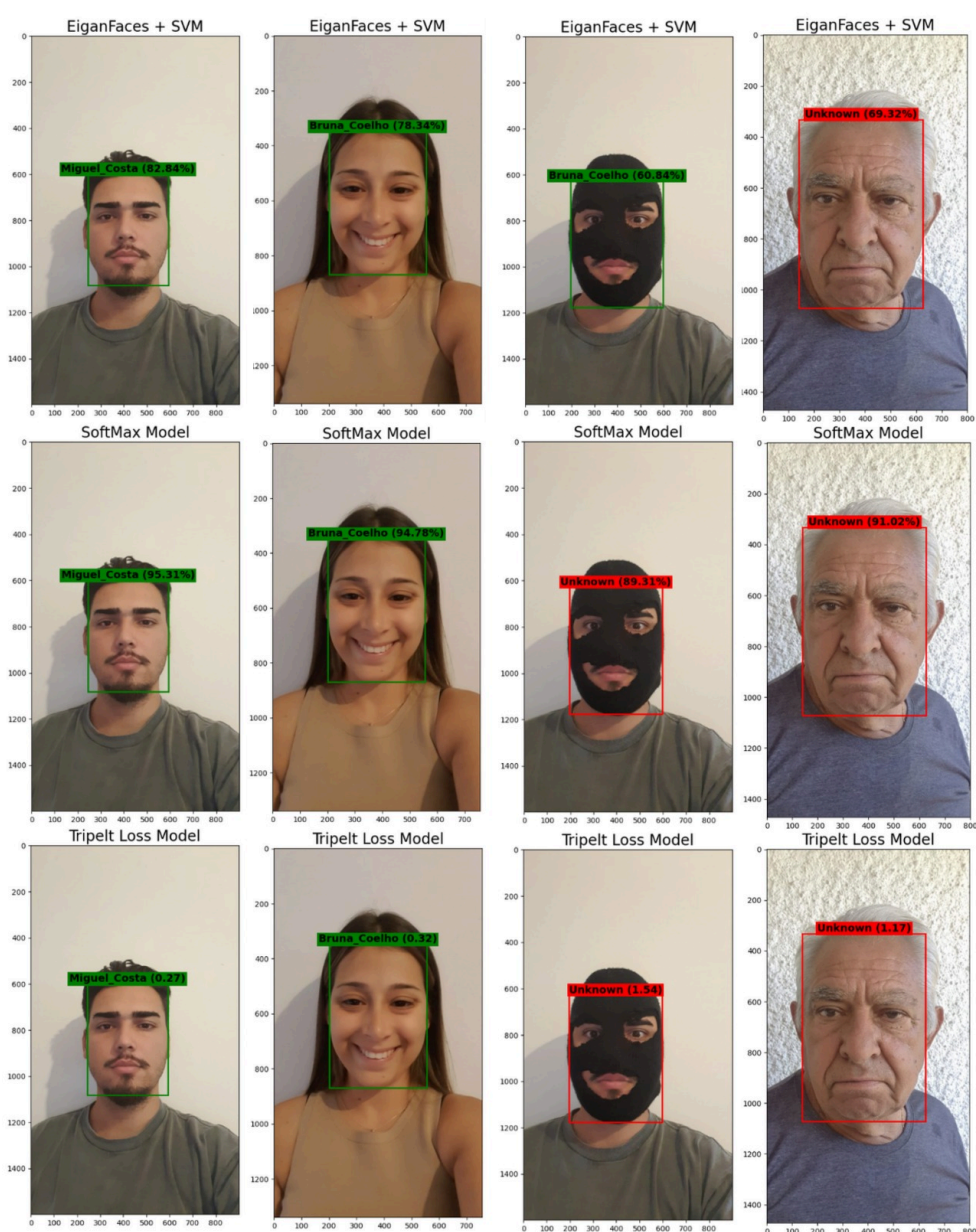


Figura 60 - Testes dos modelos de reconhecimento facial.

A técnica EigenFaces combinada com SVM apresentou uma acurácia de 0,69, precisão de 0,73, *recall* de 0,75 e um F1 Score de 0,74. Esses resultados, embora demonstrem a capacidade do modelo em realizar reconhecimento facial, ficaram abaixo do desempenho observado em modelos baseados em *deep learning*. A principal vantagem da abordagem EigenFaces+SVM é a sua leveza computacional e rapidez no treino, o que a torna uma opção viável em cenários com recursos computacionais limitados. Contudo, a precisão relativamente baixa sugere que este modelo pode não ser adequado para aplicações que exigem alta confiabilidade e robustez, como sistemas de vigilância. Esta limitação é evidenciada na Figura 60, onde é possível observar um falso positivo num dos casos analisados (rosto obstruído).

O modelo *Softmax* apresentou uma acurácia de 0.93, precisão de 0.90, *recall* de 0.92 e um F1 Score de 0.91. Estes resultados demonstram que o *Softmax* é muito mais eficaz que o EigenFaces+SVM, sendo capaz de reconhecer faces com maior precisão. Entretanto, uma desvantagem notável deste modelo é a necessidade de ser novamente treinado sempre que um novo utilizador é adicionado ao sistema. Esta característica torna-se um desafio para a atual implementação do sistema de vigilância, pois será sempre necessário ao longo do tempo ir adicionando/removendo novas entidades.

Relativamente ao modelo *Triplet Loss*, que obteve os melhores resultados do estudo feito, pode destacar-se uma acurácia de 0.97, precisão de 0.94, *recall* de 0.96 e um F1 Score de 0.95. Estes valores indicam uma performance superior em comparação aos outros dois modelos. Para mais, este modelo possui a clara vantagem de adicionar novos indivíduos rapidamente, sem a necessidade de fazer um novo treino completo do modelo, onde apenas é necessário gerar os *embeddings* para a nova entrada e armazená-los para futuro uso. Esta funcionalidade é essencial para o projeto, onde é necessário constantemente atualizar o sistema de vigilância com novas pessoas. Apesar de ser ligeiramente mais lento, esta diferença não é significativa a ponto de comprometer a eficiência em testes de vídeo real. O modelo *Triplet Loss* levou claramente a vantagem nestes testes, sendo capaz de lidar com a variabilidade e complexidade das faces em ambientes dinâmicos.

Ao comparar os três métodos, é evidente que os modelos baseados em *deep learning* (*Softmax* e *Triplet Loss*) superam a técnica EigenFaces+SVM em termos de precisão, *recall* e F1 Score. O EigenFaces+SVM, embora eficiente em termos de recursos computacionais, não atinge o nível de precisão necessário para aplicações críticas. O modelo *Softmax*, embora eficaz, apresenta a desvantagem de precisar ser treinado novamente sempre que é feita a adição/remoção de uma pessoa ao sistema, o que pode ser impraticável em sistemas que necessitam de atualizações frequentes e rápidas, tendo também um desempenho pior do que o modelo *Triplet Loss* em todas as métricas analisadas. Por outro lado, o modelo *Triplet Loss* oferece uma combinação ideal de alta precisão e flexibilidade na adição de novas pessoas. Esta capacidade de lidar com a variabilidade das faces em tempo real torna este modelo particularmente adequado para sistemas de vigilância, onde a atualização contínua da base de dados de utilizadores é uma necessidade. A leve diferença no tempo de processamento em relação aos outros modelos é compensada pela sua robustez e precisão superiores.

Para mais, o modelo *Triplet Loss* possui maior escalabilidade do que o modelo *SoftMax*. Embora essa diferença não possa ser grande num contexto doméstico, quando adaptado a um contexto onde é necessário fazer o reconhecimento facial de um maior número de pessoas, sendo exemplos num contexto empresarial, onde o número de pessoas a autenticar supera as centenas, o modelo *SoftMax* não será o mais aconselhado, pois tende a ter uma menor performance quanto maior o número de pessoas. Além disso, diversos ajustes são necessários ao *dataset* de treino do modelo *SoftMax*, sendo necessário introduzir cada vez mais classes de amostras negativas de forma a equilibrar os dados.

Sendo assim, é possível concluir que o modelo *Triplet Loss* atende de maneira mais completa aos requisitos do sistema de vigilância proposto, garantindo a precisão e flexibilidade necessárias para um desempenho eficaz num ambiente dinâmico.

4.4 Sumário

A fase de validação e experimentação dos modelos de detecção e reconhecimento facial foi essencial para garantir a eficiência e precisão do sistema implementado. Diversos testes foram realizados, utilizando diferentes arquiteturas dos diversos modelos implementados, com o objetivo de identificar quais eram as melhores soluções para a detecção e reconhecimento de rostos em tempo real. Os resultados dos testes permitiram uma comparação clara entre as arquiteturas, avaliando parâmetros como precisão, *recall*, IoU e o tempo de processamento.

No que diz respeito à detecção facial, os modelos baseados em *deep learning*, como o YOLOv8 e o Faster RCNN, apresentaram os melhores resultados, com precisões de 91% e 93% respetivamente. Esses modelos demonstraram uma clara superioridade em termos de desempenho em comparação aos métodos tradicionais, como o Haar Cascades, que apresentou uma precisão significativamente inferior. Contudo, apesar da diferença de performance, os métodos tradicionais ainda podem ser úteis em situações onde os recursos computacionais são limitados, dado que exigem menos capacidade de processamento.

Para a reconhecimento facial, os resultados seguiram uma tendência semelhante, com os modelos mais avançados, como o modelo *Triplet Loss* e o modelo *Softmax*, alcançando valores de acurácia de até 97% e 93% respetivamente. Esses algoritmos mostraram-se altamente eficazes em identificar rostos com alta precisão, especialmente em cenários com maior necessidade de segurança. Por outro lado, a técnica Eigenfaces + SVM apresentou resultados mais modestos, reforçando a ideia de que, para sistemas de vigilância mais robustos, as abordagens modernas de *deep learning* são as mais indicadas.

Os modelos de *deep learning* provaram oferecer uma performance muito superior, especialmente em termos de precisão e *recall*, ao custo de maior poder computacional. Métodos tradicionais, apesar de menos precisos, podem ainda ser utilizados em casos muito específicos onde o processamento rápido é mais importante que a exatidão, como por exemplo,

ambientes com baixa variação de rostos ou condições controladas. Assim, a escolha do modelo ideal depende do balanço entre as necessidades de precisão e os recursos disponíveis.

Os testes realizados demonstraram de forma clara a robustez e eficácia do sistema desenvolvido. Os resultados alcançados com os modelos de detecção e reconhecimento facial, especialmente os baseados em *deep learning*, comprovam que o sistema é altamente preciso e confiável. Esta avaliação mostrou que a solução pode ser aplicada em contextos reais de vigilância, oferecendo uma combinação eficiente de precisão e capacidade de resposta. Assim, conclui-se que o sistema tem o potencial de ser utilizado em ambientes reais, atendendo aos requisitos de segurança e monitoramento com grande robustez e adaptabilidade.

5 Conclusão

Neste capítulo final, a dissertação é revista como um todo, começando por examinar se os objetivos estabelecidos no início foram alcançados. Em seguida, são feitos comentários sobre a eventual continuação deste projeto e da pesquisa apresentada, seja para corrigir quaisquer defeitos no resultado ou para realizar melhorias. Por fim, são apresentadas algumas observações e considerações pessoais sobre os resultados obtidos e reflexões sobre o trabalho realizado.

5.1 Objetivos alcançados

Este subcapítulo tem como objetivo detalhar cada um dos objetivos estabelecidos no início do projeto, demonstrando como foram atingidos e destacando as contribuições significativas de cada etapa para o desenvolvimento do sistema completo.

O objetivo principal deste trabalho foi desenvolver e implementar um sistema inteligente de videovigilância, utilizando técnicas avançadas de detecção e reconhecimento facial, com o propósito de aumentar a segurança em ambientes domésticos. A questão de investigação que orientou esta dissertação foi:

Como incorporar mecanismos de inteligência artificial para melhorar os tradicionais sistemas de videovigilância, de forma a criar um sistema de vigilância completamente inteligente?

Esta questão, levou ao desenvolvimento de um sistema de vigilância robusto, que utiliza algoritmos de ML para realizar a detecção e reconhecimento facial de maneira eficiente e em tempo real.

Os principais resultados obtidos ao longo do desenvolvimento do sistema indicam que a integração de algoritmos de IA, juntamente com técnicas de detecção e reconhecimento facial, permitiu aumentar significativamente a precisão e a eficiência dos sistemas de videovigilância. Esta eficiência foi validada através de testes rigorosos, que demonstraram que o sistema é capaz de operar com elevado desempenho, alcançando uma precisão de até **93%** para detecção facial e uma acurácia de até **97%** para reconhecimento facial. Esses resultados confirmam a robustez da solução proposta, garantindo uma maior segurança para os utilizadores em ambientes domésticos.

5.1.1 Objetivo 1 – Estado de Arte

O objetivo de realizar uma análise abrangente do estado da arte foi concluído com sucesso, o que permitiu um desenvolvimento robusto do projeto. Nesta etapa, foram recolhidas e estudadas inúmeras técnicas de reconhecimento facial e detecção facial que se mostraram essenciais para a implementação do sistema desenvolvido. Além disso, a metodologia PRISMA foi utilizada para verificar um conjunto de questões de pesquisa que, por sua vez, permitiu uma revisão sistemática e rigorosa da literatura existente, garantindo que muitas publicações relevantes fossem consideradas e avaliadas.

Todo este conhecimento foi diretamente aplicado no desenvolvimento do projeto, o que resultou numa implementação que se baseou nas melhores práticas e nos avanços mais recentes da área, assim como também nos mais antigos, de forma a poder fazer uma comparação entre ambos. Em resumo, o objetivo de realizar uma análise detalhada do estado da arte foi alcançado com êxito.

5.1.2 Objetivo 2 e 3 – Datasets

Foi possível reunir diversos *datasets* para ambas as operações de detecção e reconhecimento facial, garantindo que os dados utilizados fossem os mais avançados e adequados para essas práticas nos dias de hoje. Para a detecção facial, foram utilizados os *datasets* WIDERFace, amplamente reconhecidos pela sua abrangência e qualidade, que contém uma grande variedade de imagens com diferentes condições de iluminação, ângulos e expressões faciais, e um *dataset* de fotos sem rosto, que complementa o WIDERFace ao fornecer imagens que não contêm rostos, essenciais para treinar modelos capazes de distinguir entre rostos e *background*. Para o reconhecimento facial, os *datasets* selecionados foram o VGGFACE2, mais propriamente um *subdataset* que continha 162 mil imagens, conhecido pela sua diversidade em termos de idade, sexo e etnia dos indivíduos, assim como fotos com boa e má qualidade, proporcionado uma base sólida para treinar modelos robustos de reconhecimento facial. Foi também construído um *dataset* privado dinâmico, que se adaptou continuamente ao número de pessoas autenticadas no sistema. Esta abordagem permitiu uma grande escalabilidade, ajustando-se às necessidades do sistema em tempo real.

5.1.3 Objetivo 4 e 5 – Modelos de detecção e reconhecimento

Foi o objetivo mais importante do projeto, no qual inicialmente seria apenas desenvolver um modelo para cada tarefa de detecção e reconhecimento facial. No entanto, ao longo do desenvolvimento, foi possível construir e implementar três métodos distintos para cada uma dessas tarefas, totalizando seis modelos/técnicas. Isso permitiu uma análise aprofundada da performance e dos casos de uso de cada uma dessas técnicas.

Para a detecção facial, foram implementados três modelos/técnicas diferentes, nomeadamente Haar Cascade, Faster RCN e YOLOv8. Relativamente ao reconhecimento facial, também três modelos distintos foram desenvolvidos sendo eles Eigenfaces, um modelo *Triplet Loss* e um modelo *Softmax*, tendo a rede ResNet50 como *backbone*.

A implementação de múltiplos modelos para cada tarefa permitiu não apenas um estudo comparativo detalhado entre métodos tradicionais e técnicas modernas de *deep learning*, mas também a identificação de pontos fortes e fracos de cada abordagem. Isso forneceu uma visão abrangente de como diferentes técnicas se comportam em diferentes contextos, oferecendo uma base sólida para futuras melhorias e otimizações.

5.1.4 Objetivo 6 – Sistema de Vídeo

Foi implementado com sucesso um sistema de vídeo com recurso a uma Raspberry Pi e ao protocolo RTSP que utiliza os diferentes modelos de detecção e reconhecimento facial desenvolvidos.

5.1.5 Objetivos 7, 8, 9 e 10 – Desenvolvimento do sistema de vigilância

Foi concluído com sucesso o desenvolvimento da aplicação denominada posteriormente de iDetect, que serve como o *frontend* do projeto, permitindo ao utilizador criar um perfil e configurar o seu ambiente de vigilância. A aplicação oferece funcionalidades importantes, como a adição de perfis de pessoas ao sistema através do carregamento de fotos e vídeos, sendo possível também fazer configuração dos diferentes modelos já abordados, a visualização em tempo real do estado do ambiente com identificação das pessoas presentes, e a ativação ou desativação do sistema. Além do *frontend*, também foi desenvolvida uma arquitetura complexa que possui uma série de serviços de *backend*, que gerenciam os dados dos utilizadores e que aplicam os modelos de IA.

Foi possível também implementar um sistema de envio automático de notificações e alertas ao utilizador quando este deteta um comportamento suspeito, como a presença de uma pessoa não familiar ou a falha na identificação de uma pessoa durante um determinado intervalo de tempo.

Portanto, foi construído um sistema completo, que não se limita apenas aos modelos de inteligência artificial, mas sim à integração desses modelos numa aplicação prática e funcional que demonstra a viabilidade e a aplicabilidade dos conceitos teóricos discutidos no projeto, o que resultou numa solução de vigilância inteligente dinâmica e escalável. Este desenvolvimento abrangente agrega um valor significativo à dissertação, evidenciando a criação de um produto completo que une a teoria à prática de forma eficaz.

5.2 Melhorias e Trabalhos futuros

Aquando da implementação deste projeto diversas ideias foram surgindo para enriquecer esta solução.

A primeira, seria a adição de um modelo de detecção corporal extra que iria trabalhar em paralelo com o modelo de detecção facial de modo a ser acionado quando a detecção facial não fosse possível num determinado momento, aumentando ainda mais a robustez do sistema, que ficaria apto a captar movimentos mesmo aquando da inexistência de um rosto, podendo alertar o utilizador de um possível perigo.

Outra melhoria seria a inclusão de algumas melhorias de UI, relativamente ao processo de geração de *dataset* e treino dos diversos modelos, nomeadamente com um mecanismo que forneceria o *feedback* ao utilizador através um estado de carregamento, pois ambas as operações exigem um tempo considerável. A atual implementação não possui esta característica pois não existe nenhum tipo de *feedback* ao utilizador para além das notificações, não sendo demonstrado o tempo real que aquela determinada ação irá demorar a executar, por exemplo, o treino dos modelos, que é consideravelmente longo.

A adição de mais configurações ao sistema também seria uma opção, nomeadamente o nível de sensibilidade do sistema, níveis de alerta, entre outros.

Uma melhoria significativa ao sistema seria a capacidade de se adaptar em tempo real, dependendo do contexto, sem que o utilizador tivesse de alterar os modelos selecionados manualmente, por exemplo: quando o utilizador está em casa, talvez não fará sentido recorrer aos mais poderosos modelos de ML para realizar a vigilância, sendo assim o sistema poderia adaptar para escolher as técnicas mais tradicionais de forma a poupar recursos. Outro exemplo seria o contrário, onde dependendo do espaço a ser vigiado por uma determinada câmara, o sistema iria optar pelos modelos mais pesados (embora mais lentos) para realizar uma detecção mais precisa.

Pelo facto da presente solução recorrer a uma câmara externa, nomeadamente uma Raspberry Pi, poderia ser interessante a implementação de um sistema de reconhecimento facial no qual o utilizador poderia configurar o sistema apenas com recurso à sua voz, semelhante ao comportamento de sistema inteligente do tipo Alexa²³. Esta funcionalidade seria enriquecedora pois tocaria numa temática que não foi abordada nesta presente investigação, o processamento de linguagem natural.

Um dos pontos que foi deixado de lado durante a implementação, pois estava fora do domínio de inteligência artificial, foram algumas questões relacionadas à segurança, nomeadamente os

²³ Know About Alexa -

<https://www.amazon.com/b?ie=UTF8&node=21576558011>

acessos às câmaras do sistema, pois embora exista um mecanismo que recorre a um JWT token de autenticação para aceder aos diversos serviços, este *token* poderia ser facilmente interceptado com programas externos, expondo assim uma grave falha de segurança. Sendo assim, seria necessário melhorar a segurança do sistema, possivelmente com algumas técnicas de encriptação.

Questões relacionadas com a escalabilidade da solução, tanto em termos de *hardware* quanto de *software*, também devem ser consideradas. Em particular, é importante refletir sobre como a solução seria implementada se eventualmente este sistema inteligente fosse comercializado. Existem duas opções principais: a alocação local, onde cada utilizador teria uma central de *hardware* instalada na sua própria residência para processar os dados de vigilância, ou a alocação remota, na qual uma infraestrutura centralizada, composta por servidores com elevado poder de processamento seria responsável por processar e analisar os *frames* enviados pelas câmaras de múltiplos utilizadores. A escolha entre estas abordagens depende de fatores como custo, segurança, manutenção e eficiência, e deverá ser ponderada e analisada com maior detalhe, de acordo com as necessidades específicas do mercado e dos utilizadores.

Por fim, a clara melhoria dos modelos existentes, nomeadamente os modelos de reconhecimento facial, pois embora possuísem bons resultados nos testes efetuados, é desconhecida a sua performance num sistema de larga escala, nomeadamente com milhares/milhões de indivíduos para autenticar. Para além das melhoras, seria interessante também abordar outro tipo de modelos que não foram considerados, embora possuam bons resultados, nomeadamente a implementação de um modelo com recurso a *arc loss*, semelhante ao modelo ArcFace [69] ou CosFace [70].

5.3 Considerações finais

Com a realização desta dissertação, foi possível realizar com sucesso um estudo aprofundado de três técnicas distintas de deteção facial e três técnicas de reconhecimento facial. Cada técnica foi analisada em detalhe, evidenciando as suas particularidades, pontos fortes e limitações. O estudo comparativo permitiu não só compreender melhor o funcionamento individual de cada técnica, mas também como elas se posicionam umas em relação às outras em termos de precisão, eficiência e robustez.

Além do estudo teórico, foi feita a implementação prática dos modelos estudados, o que resultou na incorporação dos modelos num sistema de vigilância inteligente, melhorando significativamente os atuais sistemas de videovigilância no mercado. Esse sistema foi projetado para ser totalmente configurável e dinâmico, de forma a simular um produto real que pode ser adaptado às necessidades específicas dos utilizadores. A implementação prática envolveu várias etapas críticas, incluindo a integração dos algoritmos de deteção e reconhecimento facial num ambiente de vigilância, o desenvolvimento de interfaces de configuração para os utilizadores e a realização de testes para garantir o funcionamento correto do sistema.

Um aspecto relevante deste projeto foi a consideração do potencial de comercialização do sistema desenvolvido. A natureza configurável e dinâmica do sistema de vigilância não apenas o torna um produto viável para uso em diversas situações de segurança, mas também abre a possibilidade de adaptação para outros mercados e aplicações. A flexibilidade do sistema é um dos seus principais diferenciais competitivos, que pode ser ajustada para atender desde pequenos ambientes de vigilância, sendo exemplo um ambiente doméstico no qual esta investigação se focou, sendo facilmente expansível para um ambiente mais complexo onde fosse necessário autenticar um maior número de pessoas.

Os resultados obtidos durante a experimentação e validação sugerem que as empresas de videovigilância devem continuar a investir fortemente em tecnologias de inteligência artificial e na personalização do utilizador, com o objetivo de aprimorar os sistemas atuais no mercado. A personalização permitirá que esses sistemas correspondam melhor às necessidades individuais dos clientes, garantindo uma maior segurança e eficiência. Para mais, os avanços em IA prometem trazer mais precisão e agilidade para as soluções de vigilância, com potencial para transformar profundamente a indústria.

Em suma, esta dissertação conseguiu atingir os seus objetivos de maneira satisfatória, na medida em que ofereceu um estudo detalhado das técnicas de deteção e reconhecimento facial e, ainda, demonstrou a viabilidade prática da sua aplicação em sistemas de vigilância. As descobertas e desenvolvimentos realizados têm o potencial de impactar significativamente tanto a pesquisa quanto a prática na área de segurança e vigilância. Assim, o investimento contínuo em IA será crucial para moldar o futuro da videovigilância, de forma a permitir que os sistemas cada vez mais inteligentes e personalizados liderem o mercado em constante evolução.

6 Referências

- [1] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "A review of video surveillance systems," *J Vis Commun Image Represent*, vol. 77, p. 103116, 2021, doi: <https://doi.org/10.1016/j.jvcir.2021.103116>.
- [2] M. Jayamohan, S. Yuvaraj, and P. Vijayakumar, "Review of Video Analytics Method for Video Surveillance," in *2021 4th International Conference on Recent Trends in Computer Science and Technology (ICRTCST)*, 2022, pp. 43–47. doi: 10.1109/ICRTCST54752.2022.9782005.
- [3] T. Monahan, *Surveillance and Security: Technological Politics and Power in Everyday Life*. New York: Routledge, 2006.
- [4] H. Chaouchi, *The Internet of Things: From RFID to the Next-Generation Pervasive Networked Systems*. London: Wiley, 2010.
- [5] H. Alloui and Y. Mourdi, "Exploring the Full Potentials of IoT for Better Financial Growth and Stability: A Comprehensive Survey," Oct. 01, 2023, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/s23198015.
- [6] M. Alarm, "What is the Best Resolution for a Security Camera?," 2023. [Online]. Available: <https://mesaalarm.com/blog/what-is-the-best-resolution-for-a-security-camera/>
- [7] Statista, "Market value of global video surveillance from 2019 to 2027," 2023. [Online]. Available: <https://www.statista.com/statistics/1234567/global-video-surveillance-market-size/>
- [8] S. P. Thirimanne, L. Jayawardana, L. Yasakethu, P. Liyanaarachchi, and C. Hewage, "Deep Neural Network Based Real-Time Intrusion Detection System," *SN Comput Sci*, vol. 3, no. 2, p. 145, 2022, doi: 10.1007/s42979-022-01031-1.
- [9] M. Alkasassbeh and S. Al-Haj Baddar, "Intrusion Detection Systems: A State-of-the-Art Taxonomy and Survey," *Arab J Sci Eng*, vol. 48, no. 8, pp. 10021–10064, 2023, doi: 10.1007/s13369-022-07412-1.
- [10] Prosegur, "Inteligência Artificial e Novos Produtos: A Estratégia da Prosegur," 2023. [Online]. Available: <https://www.prosegur.pt/artigos/sala-de-imprensa/inteligencia-artificial-e-novos-productos-estrategia-prosegur>
- [11] A. A. Dp, A. Annisa, N. Mamonto, B. Amiq, K. M. Rambe, and A. R. Syahputra, "Facial Recognition Technology: A Multinational Analysis of Regulatory Framework, Ethics, and Legal Implications in Security and Privacy," 2023. [Online]. Available: <http://ijsoc.goacademica.com>
- [12] D. Bhattacharyya, R. Ranjan, F. Alisherov, C. Minkyu, A. A. Farkhod, and M. Choi, "Biometric Authentication: A Review," 2009. [Online]. Available: <https://www.researchgate.net/publication/46189709>
- [13] L. V. Chernenkaya, E. N. Desyatirikova, and A. V. Rechinskii, "Realization of Computer Vision System for Biometric Identification of Personality," in *Proceedings - 2021 International Russian Automation Conference, RusAutoCon 2021*, Institute of Electrical and Electronics Engineers Inc., Sep. 2021, pp. 409–414. doi: 10.1109/RusAutoCon52004.2021.9537374.
- [14] J. Galbally, S. Marcel, and J. Fierrez, "Image quality assessment for fake biometric detection: Application to Iris, fingerprint, and face recognition," *IEEE Transactions on*

- Image Processing*, vol. 23, no. 2, pp. 710–724, Feb. 2014, doi: 10.1109/TIP.2013.2292332.
- [15] E. Horvitz and D. Mulligan, “Machine learning: Trends, perspectives, and prospects,” *Science (1979)*, vol. 349, no. 6245, pp. 253–255, Jul. 2015, doi: 10.1126/science.aac4520.
 - [16] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” May 27, 2015, *Nature Publishing Group*. doi: 10.1038/nature14539.
 - [17] C. Bishop, “Pattern Recognition and Machine Learning,” in *Journal of Electronic Imaging*, vol. 16, 2006, pp. 140–155. doi: 10.1117/1.2819119.
 - [18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
 - [19] E. Alpaydin, *Introduction to Machine Learning*. MIT Press, 2020.
 - [20] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2009.
 - [21] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.
 - [22] N. Sharma, R. Sharma, and N. Jindal, “Machine Learning and Deep Learning Applications-A Vision,” *Global Transitions Proceedings*, vol. 2, no. 1, pp. 24–28, Jun. 2021, doi: 10.1016/j.gltp.2021.01.004.
 - [23] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.
 - [24] A. Abraham, “Artificial Neural Networks,” vol. 17, 2005. doi: 10.1002/0471497398.mm421.
 - [25] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, “Medical image classification with convolutional neural network,” in *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 2014, pp. 844–848. doi: 10.1109/ICARCV.2014.7064414.
 - [26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, doi: 10.1109/5.726791.
 - [27] S. J. Pan and Q. Yang, “A survey on transfer learning,” 2010. doi: 10.1109/TKDE.2009.191.
 - [28] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A Survey on Deep Transfer Learning,” Aug. 2018, [Online]. Available: <http://arxiv.org/abs/1808.01974>
 - [29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” Nov. 2014, [Online]. Available: <http://arxiv.org/abs/1411.1792>
 - [30] H. C. Shin *et al.*, “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning,” *IEEE Trans Med Imaging*, vol. 35, no. 5, pp. 1285–1298, May 2016, doi: 10.1109/TMI.2016.2528162.
 - [31] M. G. Galterio, S. A. Shavit, and T. Hayajneh, “A review of facial biometrics security for smart devices,” Sep. 01, 2018, *MDPI AG*. doi: 10.3390/computers7030037.
 - [32] Y. Kortli, M. Jridi, A. Al Falou, and M. Atri, “Face recognition systems: A survey,” Jan. 02, 2020, *MDPI AG*. doi: 10.3390/s20020342.
 - [33] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features.”
 - [34] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” 2005. [Online]. Available: <http://lear.inrialpes.fr>
 - [35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

- [36] W. Liu *et al.*, “SSD: Single shot multibox detector,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.
- [37] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
- [38] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Sep. 2014, pp. 580–587. doi: 10.1109/CVPR.2014.81.
- [39] *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2017.
- [40] G. Silva, M. Alves, B. C. Bispo, and P. M. Rodrigues, “Desenvolvimento Preliminar de um Algoritmo de Detecção em Tempo Real de Estados Emocionais através de Redes Neurais Convolucionais,” *Sociedade Brasileira de Telecomunicacoes*, 2021. doi: 10.14209/sbrt.2021.1570724230.
- [41] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” 2005. [Online]. Available: <http://lear.inrialpes.fr>
- [42] H. S. Dadi and G. K. Mohan Pillutla, “Improved Face Recognition Rate Using HOG Features and SVM Classifier,” *IOSR Journal of Electronics and Communication Engineering*, vol. 11, no. 04, pp. 34–44, Apr. 2016, doi: 10.9790/2834-1104013444.
- [43] W. Liu *et al.*, “SSD: Single Shot MultiBox Detector,” Dec. 2015, doi: 10.1007/978-3-319-46448-0_2.
- [44] X. Hu and B. Huang, “Face detection based on SSD and CamShift,” 2020, pp. 2324–2328. doi: 10.1109/ITAIC49862.2020.9339094.
- [45] D. X. Ting, G. X. Juan, Y. Na, L. Chao, L. H. Tian, and L. Lu, “Face Recognition Application Based on Improved SSD Network Model,” in *Proceedings - 2020 International Conference on Virtual Reality and Visualization, ICVRV 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 30–32. doi: 10.1109/ICVRV51359.2020.00016.
- [46] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” *IEEE Signal Process Lett*, vol. 23, no. 10, pp. 1499–1503, Oct. 2016, doi: 10.1109/LSP.2016.2603342.
- [47] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Institute of Electrical and Electronics Engineers Inc., Nov. 2017, pp. 6517–6525. doi: 10.1109/CVPR.2017.690.
- [48] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Apr. 2018, [Online]. Available: <http://arxiv.org/abs/1804.02767>
- [49] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” Apr. 2020, [Online]. Available: <http://arxiv.org/abs/2004.10934>
- [50] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision*, Institute of Electrical and Electronics Engineers Inc., Dec. 2017, pp. 2980–2988. doi: 10.1109/ICCV.2017.322.
- [51] *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*. IEEE, 2018.
- [52] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. in Springer Series in Statistics. Springer, 2002.
- [53] A. M. Martõ Ānez and A. C. Kak, “PCA versus LDA.”

- [54] P. N. Belhumeur, J. ~ P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," 1997.
- [55] Ting. Zhang, *2011 3rd International Conference on Computer Research and Development : ICCRD 2011 : March 11-15, 2011, Shanghai, China*. IEEE Press, 2011.
- [56] S. Q. Nur Septi, I. N. Yulita, and H. Napitupulu, "Face Recognition Using Fisherface and Support Vector Machine Method," *Institute of Electrical and Electronics Engineers (IEEE)*, Feb. 2022, pp. 50–55. doi: 10.1109/icaibda53487.2021.9689738.
- [57] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Institute of Electrical and Electronics Engineers Inc., Nov. 2017, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.
- [58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.
- [59] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," May 2016, [Online]. Available: <http://arxiv.org/abs/1605.07146>
- [60] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017, [Online]. Available: <http://arxiv.org/abs/1704.04861>
- [61] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Jan. 2018, [Online]. Available: <http://arxiv.org/abs/1801.04381>
- [62] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Sep. 2014, [Online]. Available: <http://arxiv.org/abs/1409.1556>
- [63] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Institute of Electrical and Electronics Engineers Inc., Nov. 2017, pp. 1800–1807. doi: 10.1109/CVPR.2017.195.
- [64] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning." [Online]. Available: www.aaii.org
- [65] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2016, pp. 2818–2826. doi: 10.1109/CVPR.2016.308.
- [66] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823. doi: 10.1109/CVPR.2015.7298682.
- [67] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708. doi: 10.1109/CVPR.2014.220.
- [68] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition."
- [69] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4685–4694. doi: 10.1109/CVPR.2019.00482.
- [70] H. Wang *et al.*, "CosFace: Large Margin Cosine Loss for Deep Face Recognition," Jan. 2018, [Online]. Available: <http://arxiv.org/abs/1801.09414>
- [71] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments." [Online]. Available: <http://vis-www.cs.umass.edu/lfw/>.

- [72] S. Paul and S. K. Acharya, "A comparative study on facial recognition algorithms," *International Journal of Data Science and Big Data Analytics*, vol. 1, no. 2, p. 39, May 2021, doi: 10.51483/ijdsbda.1.2.2021.39-50.
- [73] X. Guo, "A KNN Classifier for Face Recognition," in *2021 IEEE 3rd International Conference on Communications, Information System and Computer Engineering, CISCE 2021*, Institute of Electrical and Electronics Engineers Inc., May 2021, pp. 292–297. doi: 10.1109/CISCE52179.2021.9445908.
- [74] G. Guo, S. Z. Li, and K. Chan, "Face Recognition by Support Vector Machines."
- [75] W. H. Lin, P. Wang, and C. F. Tsai, "Face recognition using support vector model classifier for user authentication," *Electron Commer Res Appl*, vol. 18, pp. 71–82, Jul. 2016, doi: 10.1016/j.elerap.2016.01.005.
- [76] C. Cortes and V. Vapnik, "Support-vector networks," *Mach Learn*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
- [77] L. Shamseer *et al.*, "Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015: elaboration and explanation," *BMJ*, vol. 349, 2015, doi: 10.1136/bmj.g7647.
- [78] K. Sharma, V. Gupta, S. Verma, and S. Avikal, "Study and Implementation of Face Detection Algorithm Using Matlab," in *2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE)*, 2018, pp. 2745–2748. doi: 10.1109/ICRIEECE44171.2018.9008990.
- [79] S. Naido and R. R. Porle, "Face Detection Using Colour and Haar Features for Indoor Surveillance," in *2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAJET)*, 2020, pp. 1–5. doi: 10.1109/IICAJET49801.2020.9257813.
- [80] R. Jayaswal and M. Dixit, "Comparative Analysis of Human Face Recognition by Traditional Methods and Deep Learning in Real-Time Environment," in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, 2020, pp. 66–71. doi: 10.1109/CSNT48778.2020.9115779.
- [81] V. Chawda, V. Arya, S. Pandey, Shristi, and M. Valleti, "Unique Face Identification System using Machine Learning," in *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2020, pp. 701–706. doi: 10.1109/ICIRCA48905.2020.9182981.
- [82] T. Edirisooriya and E. Jayatunga, "Comparative Study of Face Detection Methods for Robust Face Recognition Systems," in *2021 5th SLAAI International Conference on Artificial Intelligence (SLAAI-ICAI)*, 2021, pp. 1–6. doi: 10.1109/SLAAI-ICAI54477.2021.9664689.
- [83] M. Chandrakala and P. Durga Devi, "Two-stage classifier for face recognition using HOG features," *Mater Today Proc*, vol. 47, pp. 5771–5775, 2021, doi: <https://doi.org/10.1016/j.matpr.2021.04.114>.
- [84] K. K. Kumar, Y. Kasiviswanadham, D. V. S. N. V. Indira, P. Priyanka palesetti, and Ch. V. Bhargavi, "Criminal face identification system using deep learning algorithm multi-task cascade neural network (MTCNN)," *Mater Today Proc*, vol. 80, pp. 2406–2410, 2023, doi: <https://doi.org/10.1016/j.matpr.2021.06.373>.
- [85] M. Yuan, S. Y. Nikouei, A. Fitwi, Y. Chen, and Y. Dong, "Minor Privacy Protection Through Real-time Video Processing at the Edge," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–6. doi: 10.1109/ICCCN49398.2020.9209632.
- [86] B. Dai, J. Jiang, G. Shen, X. Wang, and Q. Wang, "Deep Face Recognition for Intelligent Video Surveillance at Electrical Substations," in *2021 IEEE 7th International Conference*

- on *Cloud Computing and Intelligent Systems (CCIS)*, 2021, pp. 514–518. doi: 10.1109/CCIS53392.2021.9754622.
- [87] R. C. Damale and Bazeshree. V Pathak, “Face Recognition Based Attendance System Using Machine Learning Algorithms,” in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2018, pp. 414–419. doi: 10.1109/ICCONS.2018.8662938.
 - [88] F. Majeed, F. Z. Khan, M. J. Iqbal, and M. Nazir, “Real-Time Surveillance System based on Facial Recognition using YOLOv5,” in *2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC)*, 2021, pp. 1–6. doi: 10.1109/MAJICC53071.2021.9526254.
 - [89] Y. Luo, “Occlusion face detection is based on Yolov5s,” in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, 2021, pp. 978–981. doi: 10.1109/ICIBA52610.2021.9688319.
 - [90] C. Peng and Q. Sang, “GMC-YOLO: A Face Detection Algorithm Based on Enhanced YOLOv8,” in *Proceedings of the 2024 3rd International Conference on Cyber Security, Artificial Intelligence and Digital Economy*, in CSAIDE '24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 123–126. doi: 10.1145/3672919.3672943.
 - [91] M. Wang, F. Zheng, and J. Lu, “Face Detection with Improved Faster-R-CNN,” in *Proceedings - 2022 4th International Conference on Artificial Intelligence and Advanced Manufacturing, AIAM 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 84–91. doi: 10.1109/AIAM57466.2022.00024.
 - [92] A. Mermet, A. Kitamoto, C. Suzuki, and A. Takagishi, “Face Detection on Pre-modern Japanese Artworks using R-CNN and Image Patching for Semi-Automatic Annotation,” in *Proceedings of the 2nd Workshop on Structuring and Understanding of Multimedia HeritAge Contents*, in SUMAC'20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 23–31. doi: 10.1145/3423323.3423412.
 - [93] W. Zhang, S. Guan, C. Wang, Y. Zhang, and X. Zhou, “Research on Face Detection and Face Attribute Recognition based on Deep Learning,” in *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2021, pp. 18–22. doi: 10.1109/IDAACS53288.2021.9660982.
 - [94] J. Nandre, S. Rai, and B. R. Kanawade, “Comparative Analysis of Transfer Learning CNN for Face Recognition,” in *2022 2nd International Conference on Intelligent Technologies (CONIT)*, 2022, pp. 1–6. doi: 10.1109/CONIT55038.2022.9847946.
 - [95] M. Masud *et al.*, “Deep learning-based intelligent face recognition in IoT-cloud environment,” *Comput Commun*, vol. 152, pp. 215–222, 2020, doi: <https://doi.org/10.1016/j.comcom.2020.01.050>.
 - [96] S. Almadby and L. Elrefaei, “Deep Convolutional Neural Network-Based Approaches for Face Recognition,” *Applied Sciences*, vol. 9, p. 4397, Oct. 2019, doi: 10.3390/app9204397.
 - [97] C. Medjahed, F. Mezzoudj, A. Rahmoun, and C. Charrier, “On an Empirical Study: Face Recognition using Machine Learning and Deep Learning Techniques,” in *Proceedings of the 10th International Conference on Information Systems and Technologies*, in ICIST '20. New York, NY, USA: Association for Computing Machinery, 2021. doi: 10.1145/3447568.3448521.
 - [98] S. Alsubai, M. Hamdi, S. Abdel-Khalek, A. Alqahtani, A. Binbusayyis, and R. F. Mansour, “Bald eagle search optimization with deep transfer learning enabled age-invariant face recognition model,” *Image Vis Comput*, vol. 126, p. 104545, 2022, doi: <https://doi.org/10.1016/j.imavis.2022.104545>.

- [99] M. Zulfiqar, F. Syed, M. J. Khan, and K. Khurshid, "Deep Face Recognition for Biometric Authentication," in *2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, 2019, pp. 1–6. doi: 10.1109/ICECCE47252.2019.8940725.
- [100] N. Jahan, P. K. Bhuiyan, P. A. Moon, and Md. A. Akbar, "Real Time Face Recognition System with Deep Residual Network and KNN," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2020, pp. 1122–1126. doi: 10.1109/ICESC48915.2020.9155579.
- [101] R. Munir and R. A. Khan, "An extensive review on spectral imaging in biometric systems: Challenges & advancements," *J Vis Commun Image Represent*, vol. 65, p. 102660, 2019, doi: <https://doi.org/10.1016/j.jvcir.2019.102660>.
- [102] L. Ouannes, A. Ben Khalifa, and N. E. Ben Amara, "Deep Learning vs Hand-Crafted Features for Face Recognition under Uncontrolled Conditions," in *2019 International Conference on Signal, Control and Communication (SCC)*, 2019, pp. 185–190. doi: 10.1109/SCC47175.2019.9116159.
- [103] Z. Cheng, X. Zhu, and S. Gong, "Face re-identification challenge: Are face recognition models good enough?," *Pattern Recognit*, vol. 107, p. 107422, 2020, doi: <https://doi.org/10.1016/j.patcog.2020.107422>.
- [104] S. Li, Z. Liu, D. Wu, H. Huo, H. Wang, and K. Zhang, "Low-resolution face recognition based on feature-mapping face hallucination," *Computers and Electrical Engineering*, vol. 101, p. 108136, 2022, doi: <https://doi.org/10.1016/j.compeleceng.2022.108136>.
- [105] N. Wang, Z. Wang, Z. He, B. Huang, L. Zhou, and Z. Han, "A Tilt-Angle Face Dataset And Its Validation," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 894–898. doi: 10.1109/ICIP42928.2021.9506052.
- [106] U. Subbiah and S. Padmavathi, "Analysis of Deep Learning Architecture for Non-Uniformly Illuminated Images," in *Proceedings of the 5th International Conference on Inventive Computation Technologies, ICICT 2020*, Institute of Electrical and Electronics Engineers Inc., Feb. 2020, pp. 38–43. doi: 10.1109/ICICT48043.2020.9112434.
- [107] O. R. Perdana, A. Tjahyanto, and F. Samopa, "Accuracy Comparison of Home Security Face Recognition Model in The Several Lighting Condition Using Some Kinect Produced Image," in *2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT)*, 2021, pp. 105–110. doi: 10.1109/EIConCIT50028.2021.9431860.
- [108] G. Jeevan, G. C. Zacharias, M. S. Nair, and J. Rajan, "An empirical study of the impact of masks on face recognition," *Pattern Recognit*, vol. 122, Feb. 2022, doi: 10.1016/j.patcog.2021.108308.
- [109] M. S. Mazli Shahar and L. Mazalan, "Face Identity for Face Mask Recognition System," in *2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, 2021, pp. 42–47. doi: 10.1109/ISCAIE51753.2021.9431791.
- [110] O. Ibitoye, "A Brief Review of Convolutional Neural Network Techniques for Masked Face Recognition," in *Proceedings - 2021 Concurrent Processes Architectures and Embedded Systems Conference, COPA 2021*, Institute of Electrical and Electronics Engineers Inc., Apr. 2021. doi: 10.1109/COPA51043.2021.9541448.
- [111] S. Malakar, W. Chiracharit, K. Chamnongthai, and T. Charoenpong, "Masked Face Recognition Using Principal component analysis and Deep learning," in *2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2021, pp. 785–788. doi: 10.1109/ECTI-CON51831.2021.9454857.
- [112] A. K. M. J. Majumder and J. A. Izaguirre, "A Smart IoT Security System for Smart-Home Using Motion Detection and Facial Recognition," in *2020 IEEE 44th Annual Computers*,

- Software, and Applications Conference (COMPSAC)*, 2020, pp. 1065–1071. doi: 10.1109/COMPSAC48688.2020.0-132.
- [113] T.-N. Do, C.-L. Le, and M.-S. Nguyen, “IoT-Based Security With Facial Recognition Smart Lock System,” in *2021 15th International Conference on Advanced Computing and Applications (ACOMP)*, 2021, pp. 181–185. doi: 10.1109/ACOMP53746.2021.00032.
 - [114] W. Xue, W. Hu, P. Gauranvaram, A. Seneviratne, and S. Jha, “An Efficient Privacy-preserving IoT System for Face Recognition,” in *2020 Workshop on Emerging Technologies for Security in IoT (ETSecIoT)*, 2020, pp. 7–11. doi: 10.1109/ETSecIoT50046.2020.00006.
 - [115] D. Alhajim, G. Akbarizadeh, and K. Ansari-Asl, “FFDR: Design and implementation framework for face detection based on raspberry pi,” in *Iranian Conference on Machine Vision and Image Processing, MVIP*, IEEE Computer Society, 2022. doi: 10.1109/MVIP53647.2022.9738788.
 - [116] F. Faisal and S. A. Hossain, “Smart Security System Using Face Recognition on Raspberry Pi,” in *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2019, pp. 1–8. doi: 10.1109/SKIMA47702.2019.8982466.
 - [117] G. Rajeshkumar *et al.*, “Smart office automation via faster R-CNN based face recognition and internet of things,” *Measurement: Sensors*, vol. 27, p. 100719, 2023, doi: <https://doi.org/10.1016/j.measen.2023.100719>.
 - [118] J. Harikrishnan, A. Sudarsan, A. Sadashiv, and R. A. S. Ajai, “Vision-Face Recognition Attendance Monitoring System for Surveillance using Deep Learning Technology and Computer Vision,” in *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*, 2019, pp. 1–5. doi: 10.1109/ViTECoN.2019.8899418.
 - [119] T. Kliangsuwan, A. Heednacram, and K. Silanon, “Face Recognition Algorithms for Online and On-Site Classes,” in *2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2022, pp. 1–6. doi: 10.1109/JCSSE54890.2022.9836309.
 - [120] V. A, R. R. Krishna, and R. V. U, “Facial Recognition System for Automatic Attendance Tracking Using an Ensemble of Deep-Learning Techniques,” in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–6. doi: 10.1109/ICCCNT51525.2021.9580098.
 - [121] E. Parliament and C. of the European Union, “Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation),” 2016. [Online]. Available: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>
 - [122] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” 2015. [Online]. Available: <http://download.tensorflow.org/paper/whitepaper2015.pdf>
 - [123] A. Paszke *et al.*, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” *CoRR*, vol. abs/1912.01703, 2019, [Online]. Available: <http://arxiv.org/abs/1912.01703>
 - [124] S. Yang, P. Luo, C. C. Loy, and X. Tang, “WIDER FACE: A Face Detection Benchmark,” *CoRR*, vol. abs/1511.06523, 2015, [Online]. Available: <http://arxiv.org/abs/1511.06523>
 - [125] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, “VGGFace2: A dataset for recognising faces across pose and age,” *CoRR*, vol. abs/1710.08092, 2017, [Online]. Available: <http://arxiv.org/abs/1710.08092>

Anexos

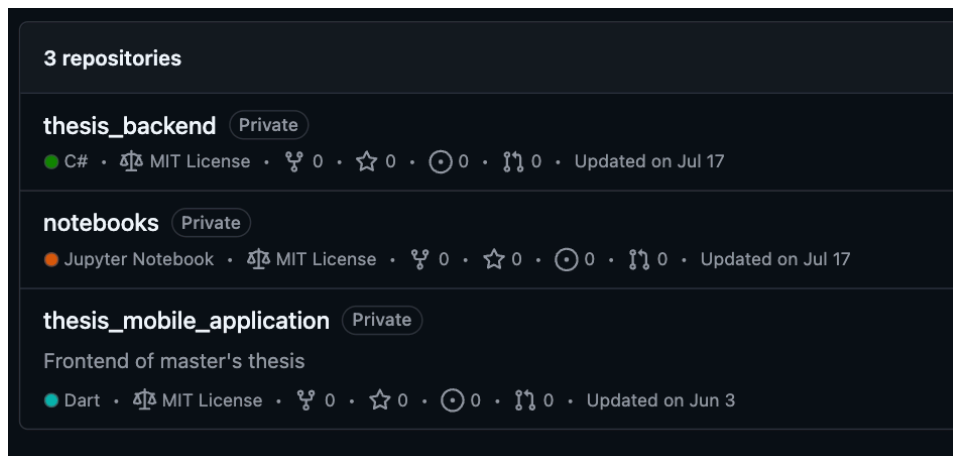


Figura 61 - Repositórios do GitHub.

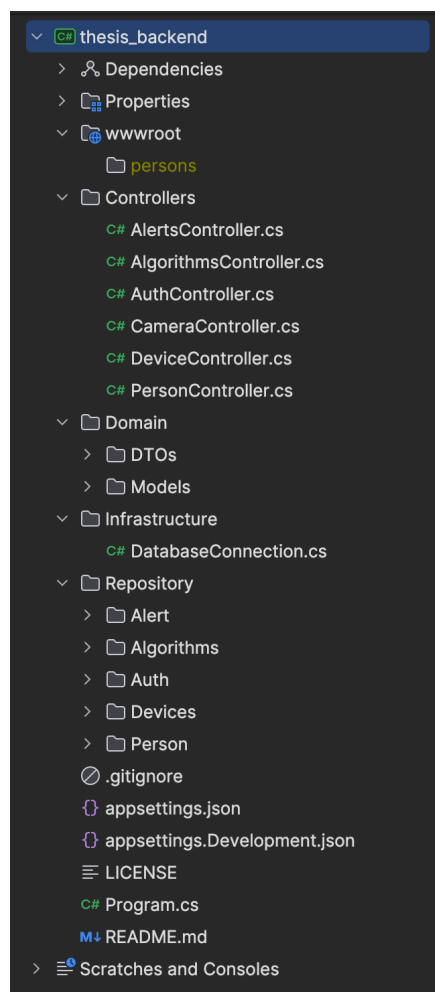


Figura 62 - Pastas do serviço de dados do utilizador.

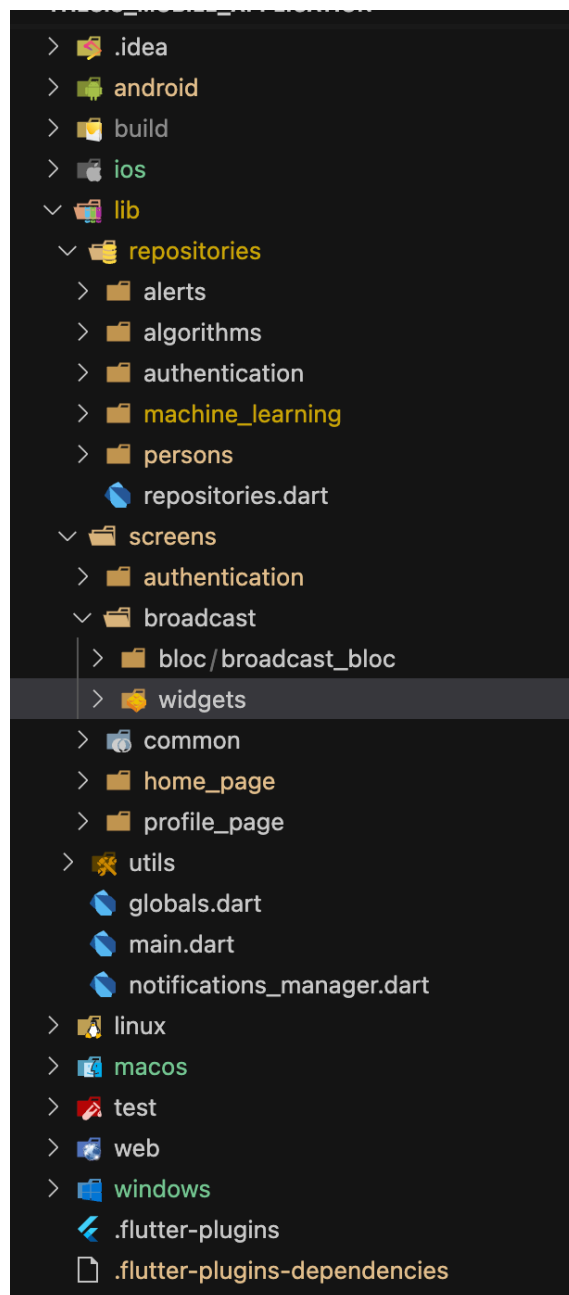


Figura 63 - Pastas da aplicação movel.

```

1. @app.route('/generate_dataset')
2. def generate_users_dataset():
3.     user_id = request.headers.get('userId')
4.     auth = request.headers.get('Authorization')
5.     get_user_videos(user_id,auth)
6.     devices = get_user_devices(user_id, auth)
7.     send_push_notification(devices, "Dataset generated ✅", "The dataset
   for your face recognition models has been generated")
8.     return Response()
9.
10. @app.route('/train_models')
11. def train_models():
12.     user_id = request.headers.get('userId')
13.     auth = request.headers.get('Authorization')
14.     notebook_path = str(path) +
   "/face_recognition/CNNs/train_recognition_models.ipynb"
15.     output_path = str(path) +
   "/face_recognition/CNNs/train_recognition_models_output.ipynb"
16.
17.     pm.execute_notebook(
18.         notebook_path,
19.         output_path,
20.         parameters={'user_id': user_id}
21.     )
22.     devices = get_user_devices(user_id, auth)
23.     send_push_notification(devices, "Finished models training ✅", "Your
   models have been trained with last generated dataset")
24.
25.     return Response()
26.
27. @app.route('/delete_dataset')
28. def delete_dataset():
29.     user_id = request.headers.get('userId')
30.     auth = request.headers.get('Authorization')
31.     delete_path = str(path) + "/data/cnn_recognition_train/" + str(user_id)
32.     delete_and_recreate_directory(delete_path)
33.     devices = get_user_devices(user_id, auth)
34.     send_push_notification(devices, "Dataset deleted ❌", "The dataset has
   been deleted from the server")
35. return Response()

```

Trecho de código 1 - Endpoints de configuração dos modelos.

```

1. def send_push_notification(devices, title, body):
2.     url = 'https://fcm.googleapis.com/fcm/send'
3.     headers = {
4.         'Authorization': 'API_KEY_HERE',
5.         'Content-Type': 'application/json'
6.     }
7.
8.     payload = {
9.         "registration_ids": devices,
10.        "notification": {
11.            "title": title,
12.            "body": body,
13.        },
14.
15.        "android": {

```

```

16.     "priority": "high",
17.     "notification": {
18.         "sound": "default",
19.         "color": "#0021FF",
20.         "click_action": "FLUTTER_NOTIFICATION_CLICK",
21.     }
22. }
23. }
24.     response = requests.post(url, headers=headers,
25.                               data=json.dumps(payload))
25. Return response.json()

```

Trecho de código 2 - *Request* ao Serviço de Notificações.

```

1. def create_user_alert(image_frame, user_id, auth_token):
2.     _, buffer = cv2.imencode('.jpg', image_frame)
3.     image_bytes = BytesIO(buffer)
4.     unique_filename = f'image_{uuid.uuid4()}.jpg'
5.     payload = {'userId': user_id}
6.     files = {'image': (unique_filename, image_bytes, 'image/jpeg')}
7.     headers = { 'Authorization' : auth_token }
8.     response = requests.post(idetect_api_url + '/api/alerts/create',
9.                               data=payload, files=files, headers=headers)
9. return

```

Trecho de código 3 - *Request* para guardar alerta.

```

1. def get_user_algorithms(user_id, auth_token):
2.     headers = { 'Authorization' : auth_token }
3.     algorithms_response = requests.get(idetect_api_url
4.                                         + '/api/algorithms/get_algorithms/' + user_id, headers = headers)
5.     algorithms_response_json = algorithms_response.json()
6.     selected_algorithms = [algorithm['algorithm'] for algorithm in
7.                             algorithms_response_json['userAlgorithms'] if algorithm['isSelected']]
8.     return selected_algorithms
9.
10. def get_user_devices(user_id, auth_token):
11.     headers = { 'Authorization' : auth_token }
12.     devices_response = requests.get(idetect_api_url
13.                                       + '/api/device/get_devices/' + user_id, headers = headers)
14.     devices_response_json = devices_response.json()
15.     devices = [device['deviceId'] for device in devices_response_json]
16. return devices

```

Trecho de código 4 - Funções para retornar dados do utilizador

```

1. class VideoStreamer:
2.     def __init__(self, face_detection_model, face_recognition_model,
   user_id, auth_token):
3.         self.face_detection_model = face_detection_model
4.         self.face_recognition_model = face_recognition_model
5.         self.label_encoder = label_encoder
6.         self.unknown_count = 0
7.         self.user_id = user_id
8.         self.auth_token = auth_token
9.         self.lock = threading.Lock()
10.        self.last_frame = None
11.
12.        def send_notification(self, frame):
13.            devices = get_user_devices(self.user_id, self.auth_token)
14.            send_push_notification(devices, "Suspicious movement 🚨", "The
system detected a suspect. Click here to watch video footage 📹")
15.            create_user_alert(frame, self.user_id, self.auth_token)
16.
17.
18.        def check_and_notify(self):
19.            with self.lock:
20.                if self.unknown_count >= 100:
21.                    if self.last_frame is not None:
22.                        threading.Thread(target=self.send_notification,
args=(self.last_frame,)).start()
23.                        self.unknown_count = 0
24.
25.        def detect_faces(self, frame):
26.            faces = []
27.            if self.face_detection_model == 'HaarCascades':
28.                gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
29.                faces = face_detection_model_haar.detectMultiScale(gray, 1.1,
30.3)
31.            elif self.face_detection_model == 'VGG16':
32.                original_height, original_width = frame.shape[:2]
33.                face_img = cv2.resize(frame, (224, 224))
34.                face_img = face_img / 255.0
35.                face_img = np.expand_dims(face_img, axis=0)
36.                faces = face_detection_model_vgg.predict(face_img, verbose = 0)
37.                faces = handle_faces_vgg(faces, original_width,
original_height)
38.            elif self.face_detection_model == 'YOLOv5':
39.                faces = face_detection_model_yolo(frame)
40.                faces = faces[0].boxes.xyxy.cpu().numpy()
41.                print(faces)
42.                faces = [[int(face[0]), int(face[1]), int(face[2]) -
int(face[0]), int(face[3]) - int(face[1]) ] for face in faces]
43.                return faces
44.
45.        def recognize_face(self, face):
46.            target_names = get_user_persons(user_id)
47.            if self.face_recognition_model == 'EigenFaces & SVM':
48.                face_pca =
face_recognition_eigenfaces.transform(face.reshape(1, -1))
49.                predicted_face = face_recognition_svm.predict(face_pca)
50.                label = target_names[predicted_face[0]]
51.            elif self.face_recognition_model == 'ResNet50':
52.                result = face_recognition_model_resnet.predict(face, verbose =
0)
53.                predicted_face = np.argmax(result, axis=1)
54.                label = self.label_encoder.inverse_transform(predicted_face)[0]

```

```

55.         elif self.face_recognition_model == 'InceptionV3':
56.             result = face_recognition_model_inceptionv3.predict(face,
57. verbose = 0)
58.             predicted_face = np.argmax(result, axis=1)
59.             label = self.label_encoder.inverse_transform(predicted_face)[0]
60.             return label
61.
62.     def stream_video(self):
63.         cap = cv2.VideoCapture(0)
64.         while cap.isOpened():
65.             ret, frame = cap.read()
66.             gray_image = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
67.             gray = cv2.cvtColor(gray_image, cv2.COLOR_GRAY2BGR)
68.             if ret:
69.                 faces = self.detect_faces(gray)
70.                 for face in faces:
71.                     x,y,w,h = face[:4]
72.                     face_img = frame[y:y+h, x:x+w]
73.                     face_img = cv2.resize(face_img, (224, 224))
74.                     face_img = face_img / 255.0
75.                     face_img = np.expand_dims(face_img, axis=0)
76.                     predicted_label = self.recognize_face(face_img)
77.                     color = (0,0,0)
78.                     if predicted_label == 'Unknown':
79.                         color = (0, 0, 255)
80.                         with self.lock:
81.                             self.unknown_count += 1
82.                     else:
83.                         color = (0, 255, 0)
84.                         with self.lock:
85.                             self.unknown_count = 0
86.
87.                 draw_ped(frame, predicted_label, x, y, x + w, y + h,
88. color, text_color=(50, 50, 50))
89.                 self.check_and_notify()
90.                 self.last_frame = frame.copy()
91.                 ret, jpeg = cv2.imencode('.jpg', frame)
92.                 yield (b'--frame\r\n'
93.                     b'Content-Type: image/jpeg\r\n\r\n' + jpeg.tobytes()
94. + b'\r\n\r\n')
95.             else:
96.                 break
97.         cap.release()

```

Trecho de código 5 - Classe *VideoStreamer* responsável por aplicar os modelos em tempo real.

```

1. ZoomableView(
2.     child: SizedBox(
3.         height: getLandscapeHeight(context),
4.         width: getLandscapeWidth(context),
5.         child: Stack(
6.             children: [
7.                 Center(
8.                     child: VlcPlayer(
9.                         controller: state.videoPlayerController!,
10.                        aspectRatio: 16 / 9,
11.                        placeholder: const Center(
12.                            child: CircularProgressIndicator(
13.                                backgroundColor: Colors.white,
14.                            ),
15.                        ),
16.                    ),
17.                ),
18.                const Padding(
19.                    padding: EdgeInsets.only(top: 16, left: 16),
20.                    child: Image(
21.                        image: AssetImage('rec.gif'),
22.                        width: 70,
23.                    ),
24.                ),
25.                Align(
26.                    alignment: Alignment.bottomRight,
27.                    child: IconButton(
28.                        onPressed: () {
29.                            context
30.                                .read<broadcastbloc>()
31.                                .add(BroadcastFullScreen(isFullscreen: !state.isFullscreen!));
32.                        },
33.                        icon: Icon(state.isFullscreen! ?
34.                            Icons.fullscreen_exit : Icons.fullscreen),
35.                        color: Colors.white,
36.                    ),
37.                ],
38.            ),
39.        ),
40.    ),
41.    if (!state.isFullscreen!)
42.    const Column(
43.        children: [
44.            Divider(
45.                height: 1,
46.                thickness: 1.5,
47.                color: themeDividerColor,
48.            ),
49.            BroadcastPageCamerasCard(),
50.        ],
51.    ),

```

Trecho de código 6 - Widget responsável por criar o ecrã do vídeo em direto.

```

1. class WiderFaceDataset(Dataset):
2.     def __init__(self, image_paths, bounding_boxes, transform=None):
3.         self.image_paths = image_paths
4.         self.bounding_boxes = bounding_boxes
5.         self.transform = transform
6.
7.     def __len__(self):
8.         return len(self.image_paths)
9.
10.    def __getitem__(self, idx):
11.        img_path = self.image_paths[idx]
12.        image = cv2.imread(img_path)
13.        if image is None:
14.            raise ValueError(f"Image not found: {img_path}")
15.        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
16.        boxes = np.array(self.bounding_boxes[idx])
17.
18.        target = {}
19.        target['boxes'] = torch.as_tensor(boxes,
dtype=torch.float32)
20.        target['labels'] = torch.ones((boxes.shape[0],),
dtype=torch.int64)
21.
22.        if self.transform:
23.            image = self.transform(image)
24.
25.        return image, target
26.
27.    def collate_fn(batch):
28.        return tuple(zip(*batch))
29.
30.    class ToTensor:
31.        def __call__(self, image):
32.            return F.to_tensor(image)
33.    transform = ToTensor()

```

Trecho de código 7 - *Dataloader* para treinar o modelo Faster RCNN.

```

1. class TripletDataGenerator(Sequence):
2.     def __init__(self, image_paths, labels, batch_size):
3.         self.image_paths = np.array(image_paths)
4.         self.labels = np.array(labels)
5.         self.batch_size = batch_size
6.         self.label_indices = self._create_label_indices()
7.         self.on_epoch_end()
8.
9.     def _create_label_indices(self):
10.        label_indices = {}
11.        for idx, label in enumerate(self.labels):
12.            if label not in label_indices:
13.                label_indices[label] = []
14.            label_indices[label].append(idx)
15.        return label_indices
16.
17.    def _get_positive_negative_indices(self, label):
18.        positive_indices = self.label_indices[label]

```

```

19.         negative_indices = [idx for l, idx_list in
self.label_indices.items() if l != label for idx in idx_list]
20.         return positive_indices, negative_indices
21.
22.     def __len__(self):
23.         return int(np.floor(len(self.image_paths) / self.batch_size))
24.
25.     def __getitem__(self, index):
26.         X, y = self.__data_generation()
27.         return X, y
28.
29.     def on_epoch_end(self):
30.         self.indices = np.arange(len(self.image_paths))
31.         np.random.shuffle(self.indices)
32.
33.     def __data_generation(self):
34.         anchors = np.zeros((self.batch_size, 224, 224, 3),
dtype=np.float32)
35.         positives = np.zeros((self.batch_size, 224, 224, 3),
dtype=np.float32)
36.         negatives = np.zeros((self.batch_size, 224, 224, 3),
dtype=np.float32)
37.
38.         for i in range(self.batch_size):
39.             idx_anchor = np.random.choice(self.indices)
40.             anchor_path = self.image_paths[idx_anchor]
41.             anchor = img_to_array(load_img(anchor_path, target_size=(224,
224))) / 255.0
42.             label = self.labels[idx_anchor]
43.             pos_indices, neg_indices =
self._get_positive_negative_indices(label)
44.
45.             idx_positive = np.random.choice(pos_indices)
46.             idx_negative = np.random.choice(neg_indices)
47.
48.             positive_path = self.image_paths[idx_positive]
49.             negative_path = self.image_paths[idx_negative]
50.
51.             positive = img_to_array(load_img(positive_path,
target_size=(224, 224))) / 255.0
52.             negative = img_to_array(load_img(negative_path,
target_size=(224, 224))) / 255.0
53.
54.             anchors[i] = anchor
55.             positives[i] = positive
56.             negatives[i] = negative
57.
58.         return [anchors, positives, negatives], np.zeros((self.batch_size,
1))
59.
60.     def get_triplet_batch(self):
61.         return self.__data_generation()
62.
63. validation_split = 0.10
64. train_image_paths, val_image_paths, train_labels, val_labels =
train_test_split(train_image_paths, train_labels,
test_size=validation_split, stratify=train_labels, random_state=42)
65.
66. batch_size = 512
67. train_triplet_data_gen = TripletDataGenerator(train_image_paths,
train_labels, batch_size)

```

```
68. val triplet_data_gen = TripletDataGenerator(val_image_paths, val_labels,
    batch_size)
```

Trecho de código 8 - Classe para geração de tripletos de treino e validação.

```
1. public IActionResult AddPerson([FromForm] PersonRequest person)
2. {
3.     _logger.LogInformation("AddPerson {DT}",
4.         DateTime.UtcNow.ToLongTimeString());
5.
6.     var hostAddress
7.     = $"{Request.Scheme}://{Request.Host}{Request.PathBase}";
8.     var personImageDirectoryPath = _webHostEnvironment.WebRootPath +
9.     "/persons/" + person.UserId + "/image";
10.    var personVideoDirectoryPath = _webHostEnvironment.WebRootPath +
11.    "/persons/" + person.UserId + "/video";
12.
13.    var hostAddressImagesUrls = "/persons/" + person.UserId +
14.    "/image";
15.    var hostAddressVideosUrls = "/persons/" + person.UserId +
16.    "/video";
17.
18.    if (!Directory.Exists(personImageDirectoryPath))
19.    {
20.        Directory.CreateDirectory(personImageDirectoryPath);
21.    }
22.    if (!Directory.Exists(personVideoDirectoryPath))
23.    {
24.        Directory.CreateDirectory(personVideoDirectoryPath);
25.    }
26.
27.    var imageFilePath = Path.Combine(personImageDirectoryPath,
28.    person.Image!.FileName);
29.    var videoFilePath = Path.Combine(personVideoDirectoryPath,
30.    person.Video!.FileName);
31.
32.    using Stream imageFileStream = new FileStream(imageFilePath,
33.    FileMode.Create);
34.    using Stream videoFileStream = new FileStream(videoFilePath,
35.    FileMode.Create);
36.
37.    person.Image.CopyTo(imageFileStream);
38.    person.Video.CopyTo(videoFileStream);
39.
40.    var imageUrl = Path.Combine(hostAddressImagesUrls,
41.    person.Image!.FileName);
42.    var videoUrl = Path.Combine(hostAddressVideosUrls,
43.    person.Video!.FileName);
44.
45.    var uuid = GeneratePersonUuid();
46.    var newPerson = new Person(person.UserId, uuid, person.Name,
47.    person.Surname, imageUrl, videoUrl);
48.
49.    var statusMessage=_personRepository.AddPerson(newPerson);
50.    return Ok(statusMessage);
51. }
```

Trecho de código 9 - Lógica para criar uma pessoa autenticada no sistema de um utilizador.