INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




Desenvolvimento de um Sistema de Reconhecimento Facial para o Instituto Federal do Maranhão (IFMA)

RAFAEL NASCIMENTO DE SOUSA
novembro de 2024

Desenvolvimento de um Sistema de Reconhecimento Facial para o Instituto Federal do Maranhão (IFMA)

Rafael Nascimento de Sousa
Aluno nº: 1222622

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

Orientador: Doutor Carlos Fernando Silva Ramos
Co-orientador: Doutor Aristóteles de Almeida Lacerda Neto

Júri:
Presidente: Doutor Luiz Felipe Rocha de Faria, Professor Coordenador do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Arguente: Doutor António Constantino Lopes Martins, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto
Arguente: Doutor Luís Manuel Silva Conceição, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto
Orientador: Doutor Carlos Fernando da Silva Ramos, Professor Coordenador Principal do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto
Co-orientador: Doutor Aristóteles de Almeida Lacerda Neto, Professor do Instituto Federal de Educação Ciência e Tecnologia do Maranhão

Porto, novembro de 2024

Dedicatória

Eu dedico esta tese ao meu pai, Joel Reis de Sousa, que faleceu no ano de 2015, mas não antes de deixar seus ensinamentos e seu legado aos filhos que sempre o amarão.

Resumo

Este documento apresenta o estudo e desenvolvimento de um sistema de reconhecimento facial para aplicação no Instituto Federal do Maranhão (IFMA). Este programa tem como objetivo otimizar processos do campus, especialmente a autenticação dos alunos selecionados para o auxílio alimentação. A pesquisa foi feita baseada em algoritmos de aprendizagem profunda (*deep learning*) e técnicas de otimização dos treinos. Além do desenvolvimento, foi feita também uma pesquisa aprofundada sobre o estado da arte da tecnologia de forma que orientasse os primeiros passos da implementação do *software*. O principal modelo utilizado na implementação é composto por uma rede siamesa regida por *triplet* loss. Os resultados indicam uma boa capacidade de reconhecimento e autenticação dos alunos, principalmente em uma base de dados menor, assim como é o objetivo. O projeto também discute os problemas encontrados, como otimização, aprendizado, ferramentas utilizadas e métodos. Também é abordado o tratamento de dados sensíveis como as fotografias de alunos do ensino médio brasileiro.

Palavras-chave: deep learning, inteligência artificial, redes siamesas, triplet loss

Abstract

This document presents the study and development of a facial recognition system for application at the Federal Institute of Maranhão (IFMA). This program aims to optimize campus processes, especially the authentication of students selected for food aid. The research was based on deep learning algorithms and training optimization techniques. In addition to the development, in-depth research was also carried out into the state of the art of the technology in order to guide the first steps of the software's implementation. The main model used in the implementation consists of a Siamese network governed by triplet loss. The results indicate a good ability to recognize and authenticate students, especially in a smaller database, as is the goal. The project also discusses the problems encountered, such as optimization, learning, tools used and methods. The treatment of sensitive data such as photographs of Brazilian high school students is also addressed.

Keywords: deep learning, artificial intelligence, siamese networks, triplet loss

Agradecimentos

Eu gostaria de agradecer primeiramente a Deus, por me dar sabedoria e forças para superar os desafios da vida. Ao meu pai (in memoriam) Joel e minha mãe Angela, que sempre me deram a melhor educação possível e lutaram com o que tinham para me dar tempo e possibilidade de estudar para alcançar meus objetivos. Aos meus avós paternos, Sebastião e Tereza, e maternos, José e Maria Aparecida, que me deram total suporte financeiro e emocional para suprir aquilo que sozinho não conseguiria. A minha fiel compannheira Nayara, que me deu apoio e me motivou nos momentos em que achei que não suportaria ou não seria capaz de avançar nas minhas metas. Ao IFMA, que, desde o começo, foi o principal possibilitador do inesperado intercâmbio que me trouxe a oportunidade de um mestrado. Ao ISEP e seus integrantes, por me apresentar novos caminhos e por me capacitar como profissional na área de inteligência artificial. E aos meus orientadores, Carlos Ramos e Aristóteles Lacerda Neto, que me trouxeram retorno e apoio durante toda a pesquisa, seja em forma de revisores ou de qualificadores.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

ACL Attention Center Loss

AE Attention Erasion

ANN Artificial Neural Networks

API Application Programming Interface

CAM Channel Attention Mechanism

CBAM Convolutional Block Attention Mechanism

CNN Convolutional Neural Network

CPU Central Processing Unit

CUDA Compute Unified Device Architecture

DLA Dynamic Link Architecture

DNN Deep Neural Network

EGM Elastic Graph Matching

FC Face/Facial Recognition

GAN Generative Adversarial Network

GAP Global Average Pooling

GDC Global Depthwise Convolution

GPU Graphics Processing Unit

HOG Histogram of Oriented Gradients

IA Inteligência Artificial

ICA Independent Component Analysis

IDQ Identification Quality Loss

IEEE Institute of Electrical and Electronic Engineers

ILSVRC ImageNet Large Scale Visual Recognition Challenge

IFMA Instituto Federal de Educação, Ciência e Tecnologia do Maranhão

LBP Local Binary Pattern

LDA Linear Discriminative Analysis

LFW Labeled Faces in the Wild

LLM Large Language Model

MCA Mutual Component Analysis

MFR Masked Face Recognition

NAN Neural Aggregation Network

OFC Occluded Face Recognition

PCA Principal Component Analysis

PNAES Programa Nacional de Assistência Estudantil

PRISMA Preferred Reporting Items for Systematic Reviews and Meta-Analyses

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

RQ Research Questions

SAM Spatial Attention Mechanism

SE Squeeze and Excitation

SVM Support Vector Machine

TFX Tensorflow Extended

ViT Visual Transformers

VRAM Video Random Acess Memory

1 Introdução

1.1 Contextualização

Desde o século passado, as tecnologias digitais vêm sendo cada vez mais estudadas e aprimoradas. Esse avanço é refletido na sociedade, principalmente, na forma de otimização de tempo e recursos com a criação de mecanismos que automatizam e/ou facilitam a execução de tarefas. É possível observar a utilização dessas ferramentas em vários ambientes na sociedade, como os sistemas de estoque no comércio, os equipamentos eletrônicos nos hospitais e a virtualização do ensino nas escolas.

Durante o período da pandemia do Covid-19, houve um aumento significativo na utilização dessas tecnologias, principalmente em áreas com foco na comunicação, como na administração das empresas [1] e educação [2]. Desde então, uma vez que a sociedade foi exposta de forma brusca ao intenso uso de tais sistemas, a necessidade de possuí-los cresceu ainda mais com o passar do tempo, transformando-os em recursos imprescindíveis em várias partes do mundo.

O Brasil, assim como a maioria dos países, teve uma evolução notável na aplicação de métodos digitais na educação, apesar das dificuldades de infraestrutura em muitas escolas do país [3]. Portanto, apesar da implementação de metodologias de ensino *online* nas instituições de ensino brasileiras, há uma certa lacuna tecnológica por falta de recursos. Por esse motivo, ainda hoje, há utilização de métodos obsoletos para a realização de processos que poderiam ser substituídos por ferramentas computacionais.

Neste contexto, a inteligência artificial (IA) como um conceito que se popularizou muito nos últimos anos, é uma área da computação capaz de resolver inúmeros problemas cotidianos que podem ser adaptados para as escolas. Os sistemas de reconhecimento, seja de voz, face ou biometria, podem ser utilizados como ilustração para essa utilização. *Softwares* como esse podem mudar a forma como é feito o controle de registros nas instituições, permitindo uma melhor automatização de processos.

Uma forma prática de preencher essa lacuna é o desenvolvimento de produtos de inovação e tecnologia pelos alunos dos cursos técnicos e superiores de computação. A maioria das

instituições de ensino públicas que oferecem tais cursos já utilizam essa estratégia por meio de projetos de pesquisa e de extensão. No entanto, como as pesquisas são publicadas em forma de artigo, há uma certa limitação no estudo aprofundado de cada produção.

Essa necessidade de produção científica e tecnológica se torna, portanto, a maior motivação para a produção desta dissertação, seguida do desejo de desenvolver um produto responsável pela verificação, controle e manutenção de acesso estudantil aos recursos disponibilizados aos discentes no Instituto Federal de Educação, Ciência e Tecnologia do Maranhão (IFMA) [4] – Campus Santa Inês, uma instituição de ensino brasileira.

1.2 Descrição do Problema

O Decreto Brasileiro nº 7.234, de 19 de julho de 2010, sancionou o Programa Nacional de Assistência Estudantil (PNAES) como programa governamental oficial [5]. O PNAES foi criado para auxiliar os estudantes brasileiros com renda familiar per capita menor que 1 (um) salário mínimo e meio, disponibilizando ajuda financeira nos gastos com locomoção, moradia, alimentação, material escolar, entre outros [6].

A fiscalização do uso desses recursos é regularmente realizada para assegurar que apenas indivíduos que realmente necessitam deles estejam os utilizando. Geralmente, comprovantes de gastos são solicitados para a comprovação da correta utilização da ajuda financeira, com algumas exceções que demandam mais trabalho para controlar dessa forma.

No exemplo específico do auxílio alimentação no IFMA – Campus Santa Inês, há um contrato com uma empresa de alimentação terceirizada que fornece um conjunto de alimentos a todo aluno que apresentar uma ficha de uso único indicando que ele é utilizador do programa. Para conseguir a ficha, o estudante deve ir até um funcionário público responsável por registrar a assinatura de todos os discentes que receberam o auxílio naquele dia.

É notável que tal abordagem é lenta e obsoleta, tendo que alocar funcionários para executar tarefas que podem ser automatizadas, e também podendo ocasionar possíveis problemas humanos, como erros na assinatura. Além disso, o processo de extração desses registros também é lento, uma vez que para analisá-los de maneira eficiente, seria necessário digitalizar todos os documentos, o que demandaria ainda mais tempo.

Portanto, é importante que haja uma forma de aprimorar esses processos, e esta dissertação apresenta uma possível resolução para o problema do auxílio alimentação no campus Santa Inês do IFMA. Como forma de automatizar o controle e registro dos beneficiários do programa, seria criado um sistema de reconhecimento facial capaz de identificar os rostos e compará-los com uma base de dados contendo a face dos auxiliados. A base de dados seria devidamente protegida e deveria ser aprovada pelos órgãos responsáveis pela segurança das informações.

1.3 Objetivos e Questões de Pesquisa

O principal objetivo dessa dissertação é o desenvolvimento de um sistema de reconhecimento facial capaz de identificar precisamente os usuários de auxílio alimentação do IFMA. A fim de alcançar essa meta maior, foram definidos outros objetivos específicos da pesquisa.

- **Objetivo 1:** Estudar o estado da arte do reconhecimento facial.
- **Objetivo 2:** Desenvolver um *software* funcional e preciso de reconhecimento facial.
- **Objetivo 3:** Trabalhar na implementação de um sistema de armazenamento seguro para os dados dos alunos.
- **Objetivo 4:** Desenvolver uma interface de utilização simples e intuitiva.

O próximo capítulo desta dissertação usará as seguintes questões de pesquisa (Research Questions - RQ) como guia de pesquisa. As RQs são uma parte da questão principal da investigação: “Qual é o estado da arte atual dos métodos de reconhecimento facial seguros e eficazes que conseguem abranger um grande número de pessoas?”

- **RQ1:** Quais são os principais algoritmos de deep learning utilizados para reconhecimento facial e como funcionam?
- **RQ2:** Quais são os *datasets* mais utilizados para reconhecimento facial de múltiplas pessoas?

1.4 Aspectos Éticos

A inteligência artificial está trazendo muitos avanços para a sociedade. Automatização de processos, melhoria de segurança, execução de tarefas perigosas e cansativas são algumas das contribuições feitas por essa tecnologia. No entanto, há sempre uma discussão sobre os impactos negativos de tais ferramentas, como invasão de privacidade, enviesamento dos dados, perda de empregos, entre outros. É sempre importante que esses aspectos possam ser considerados, de forma que nenhum indivíduo seja prejudicado com a falta deles. Portanto, as principais questões que devem ser consideradas para este trabalho são:

- **Privacidade:** É de suma importância que o armazenamento das informações dos alunos seja segura. As únicas pessoas que poderiam ter acesso ao sistema interno, seriam os funcionários autorizados da administração do instituto.
- **Substituição de Emprego:** Apesar de estar sendo citado, esse fator não é preocupante, uma vez que o projeto visa automatizar uma função designada a um funcionário que, normalmente, deveria estar ocupado com outras tarefas.
- **Enviesamento dos Dados:** É imprescindível que o conjunto de dados seja imparcial e não tenha viés em relação à raça, gênero e idade das pessoas.

1.5 Estrutura da Dissertação

A tese foi divida em cinco capítulos mais as referências. Os capítulos são: Introdução, Estado da Arte, Metodologia, Implementação e Conclusão. Este capítulo, a Introdução apresenta alguns tópicos de contextualização da produção da tese. O segundo capítulo, contempla todo o desenvolvimento do Estado da Arte, desde a metodologia até as conclusões da pesquisa. O capítulo 3 apresenta os métodos utilizados para a produção dos códigos e desenvolvimento do software. Esse desenvolvimento é exibido no capítulo seguinte, que aborda a implementação do modelo. Por fim, a conclusão é feita no último capítulo que precede as referências.

2 Estado da Arte

Neste capítulo, é apresentado um estudo sobre o estado da arte das tecnologias abordadas nesta dissertação. O principal objetivo da investigação é responder às questões de pesquisa abordadas anteriormente, a fim de obter um resultado para a pergunta principal. A estrutura do capítulo é composta pela metodologia da pesquisa seguida pelos resultados de cada busca.

2.1 Metodologia Geral de Pesquisa

Para um estudo detalhado do tema, uma revisão sistemática foi conduzida seguindo parte dos elementos descritos na *checklist* do *Preferred Reporting Items for Systematic Reviews and Meta-Analyses* (PRISMA) de 2015 [7]. O guia foi utilizado para que houvesse uma melhor consistência e robustez nas referências coletadas e que a investigação fosse mais precisa e coerente. Os itens considerados foram: critério de inclusão; critério de exclusão; fixação de fontes; definição de palavras fixas para a pesquisa em cada fonte.

As fontes selecionadas para toda a pesquisa foram os sites Google Scholar [8] e *Institute of Electrical and Electronic Engineers (IEEE)* Xplore [9], por disponibilizarem artigos que abrangem temas computacionais, incluindo aqueles que são importantes para esta pesquisa. No processo de investigação dos temas, foram selecionados 30 artigos filtrados pela relevância e proximidade com a questão. O número foi escolhido a fim de otimizar, mas manter a consistência na recolha de informações, uma vez que os textos foram filtrados utilizando os critérios de inclusão e exclusão, até que a quantidade de artigos obtidos atingisse o limite pré-estabelecido. No entanto, na RQ2, por ser uma dúvida sobre *datasets*, poderia ser considerado utilizar fontes especializadas em *datasets* como Kaggle [10] e Zenodo [11], mas como um tema consolidado na pesquisa, há conjuntos de dados que se repetem múltiplas vezes nos textos revisados, como ainda será explorado neste capítulo, tornando mais viável a utilização das fontes originais a fim de ter uma melhor consistência.

Como forma de obter os resultados mais apropriados na investigação, foram selecionadas palavras de pesquisa a serem utilizadas na busca avançada nas duas fontes. O comando

formado pelas palavras foi: (("deep learning" OR "neural networks") AND ("facial recognition" OR "face recognition") AND ("state-of-the-art" OR "performance evaluation"))

A Tabela 1 demonstra os critérios de inclusão e exclusão definidos para cada pergunta, onde cada elemento foi identificado como IC e EC, respectivamente. O idioma português no IC1 abrange, principalmente, a variação brasileira em razão do foco do estudo em instituições brasileiras. Como apresentado no IC3, os artigos foram filtrados pelo título e resumo como forma de remover rapidamente fontes irrelevantes para a solução do problema. Textos disponíveis para acesso aos estudantes do Instituto Politécnico do Porto [12] não são excluídos pelo EC1. O intervalo de 5 anos foi definido para uma pesquisa atualizada das tecnologias, como é mostrado no EC3, mas *softwares* fortemente consolidados no mercado também serão considerados na pesquisa, apesar de serem datados.

Tabela 1: Critérios de Inclusão de Exclusão

Critérios de Inclusão	**Critérios de Exclusão**
IC1: Artigos escritos em português ou em inglês; IC2: Artigos publicados nos sites IEEE Xplore e Google Scholar; IC3: Artigos que envolvam, no título ou no resumo, o mesmo assunto da pergunta em questão.	EC1: Texto completo indisponível; EC2: Duplicatas; EC3: Artigos publicados antes do ano de 2019.

2.2 Algoritmos de Reconhecimento Facial

Os algoritmos de reconhecimento facial (FR, do inglês *facial recognition*) são muito variados, podendo abranger sistemas de arquiteturas e técnicas completamente diferentes. Esse é mais um motivo para que a comunidade esteja sempre buscando avançar nessa área e, com a popularização da inteligência artificial, essa busca cresceu ainda mais.

O principal foco da pesquisa é o desenvolvimento de um sistema de FR. Por isso, esta fase do estudo será a mais densa do estado da arte. Este tópico abrangerá os resultados obtidos pela revisão que busca responder à RQ1: “Quais são os principais algoritmos de deep learning utilizados para reconhecimento facial e como funcionam?”.

Após a recolha de dados e filtragem dos artigos, foram selecionados 28 documentos para a obtenção da resposta para a RQ1. Durante a seleção, foi possível observar que os textos retirados da IEEE Xplore tinham aspectos mais técnicos e apresentavam soluções para problemas específicos e, em geral, avançados. Por outro lado, na página do Google Scholar, havia muito mais revisões e apresentações do estado da arte da tecnologia, expondo aspectos genéricos dela.

2.2.1 Algoritmos Clássicos de FR

Dentre as abordagens apresentadas nos trabalhos de revisão, também são incluídas técnicas além das redes neurais, representadas na Figura 1 [13]:

- *Principal Component Analysis* (PCA): É um modelo estatístico que representa um conjunto de variáveis correlacionadas (características faciais) em variáveis lineares não correlacionadas, chamadas de componentes principais, utilizando uma transformação ortogonal [14], [15], [16].
- *Linear Discriminative Analysis* (LDA): É uma técnica parecida com a anterior, mas que ao invés de criar um subespaço para representar as características individuais, ela cria um subespaço para representar faces de vários indivíduos e tenta maximizar a distância entre eles [14], [17], [18].

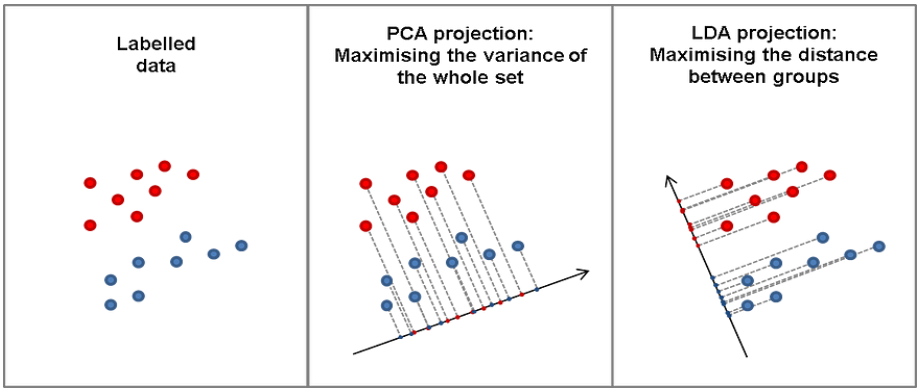

Figura 1: Exemplo da funcionalidade dos algoritmos PCA e LDA.

Esses dois algoritmos foram citados mais de uma vez dentre os documentos de estado da arte encontrados, mas também houve outras técnicas citadas que devem ser destacadas: *Support Vector Machine* (SVM) [18], *Independent Component Analysis* (ICA) [18], *Local Binary Pattern* (LBP) [14], *Histogram of Oriented Gradients* (HOG) [14], *Elastic Graph Matching* (EGM) [15], *Dynamic Link Architecture* (DLA) [17].

2.2.2 Arquiteturas

É perceptível, também, que as referências utilizadas acima apresentaram algumas semelhanças e destacaram várias abordagens que não fazem parte do escopo da aprendizagem profunda, mas houve também citações a tecnologias desse gênero, mas, como dito anteriormente, representando técnicas genéricas e citando trabalhos específicos que utilizam tais ferramentas.

Como técnicas de *deep learning* mencionadas nos textos, houve uma grande repetição das *Convolutional Neural Networks* (CNN), *Artificial Neural Networks* (ANN) e *Deep Neural Networks* (DNN). Mas de acordo com Saeed et al. [19], as CNNs são o futuro do reconhecimento de expressões, justificando com sua pesquisa que abordou a evolução da tecnologia na área.

Mas apesar da repetição dos métodos de deep learning, os autores deixam explícitas as arquiteturas mais utilizadas provindas de tais métodos. Dentre elas, as mais citadas nas revisões são quatro: *AlexNet*, *ResNet*, *VGGNet* e *GoogleNet*.

A AlexNet (Figura 2 [20]) arquitetura vencedora do *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) de 2012 [21], é composta por cinco camadas convolucionais e três *fully-connected layers* (FC), utilizando também a função de ativação *rectified linear unit* (ReLU) e técnicas como *dropout* e *data augmentation* [22].

Singhal et al. [14] afirmam em seu artigo que houve uma pesquisa [23] que produziu um *framework* para óculos inteligentes capaz de executar a tarefa de FR. Esta tecnologia era destinada a agentes de segurança pública para que a identificação de suspeitos fosse mais precisa e rápida. A arquitetura AlexNet foi utilizada e atingiu uma precisão de 98,5% após o treinamento de 2500 fotografias.

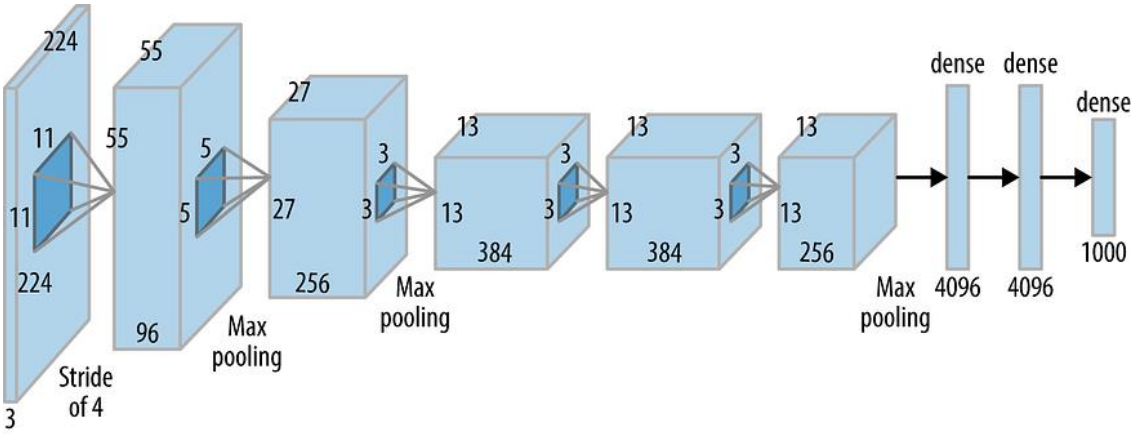

Figura 2: Estrutura da arquitetura AlexNet.

A ResNet, por sua vez, é um método que utiliza blocos residuais que fazem as informações pularem camadas da rede de treinamento. Isso resulta na diminuição do desaparecimento do gradiente, facilitando o processo de aprendizado de redes muito profundas [14]. Essa arquitetura tem inúmeras variações que, em sua grande maioria, apresentam um número seguido do nome ResNet, como o modelo ResNet-50 [24]. Os estudos de Du et al. [25] apresentam essa variedade de derivações da arquitetura. Neste artigo, são mencionados vários algoritmos da época, que na maioria foram feitos utilizando alguma variação da ResNet.

O desenvolvimento da arquitetura SqueezeNet foi um dos textos referenciados por Du et al. em sua pesquisa. O projeto desenvolvido por Iandola et al. [26] resultou em uma tecnologia que mantinha o nível de precisão da AlexNet apresentada na ImageNet, mesmo com 50 vezes menos parâmetros. Além disso, foi feita uma compactação que fez com que esse algoritmo exigisse, no fim, 510 vezes menos espaço de armazenamento do que a AlexNet: 0.5MB. Este trabalho utilizou a arquitetura ResNet como sua principal base.

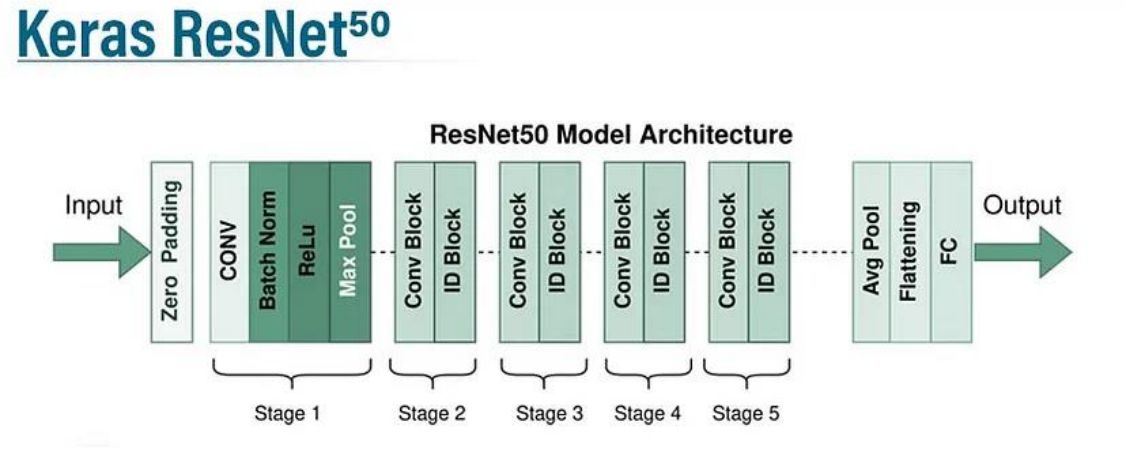

Figura 3: Estrutura da arquitetura ResNet50 apresentada pelo Keras [20]

Na ILSVRC de 2014, a arquitetura conhecida como VGGNet foi a vencedora. Dessa vez, Simonyan e Zisserman [27] aplicaram camadas com pequenos filtros convolucionais seguidos de camadas de pooling, compondo, assim, cerca de 17 *layers* totais [15]. Essa tecnologia também teve várias derivações, assim como é apresentado em um artigo [22].

Wang e Deng [22] mencionam o projeto desenvolvido por Sun et al. [28] chamado DeepID3, que mantinha um grande índice de precisão utilizando a base de dados LFW [29], atingindo 99.56% de precisão na tarefa de identificação facial e 96% na tarefa de detecção facial. A tecnologia em questão resultou de uma união das arquiteturas VGGNet e GoogleNet.

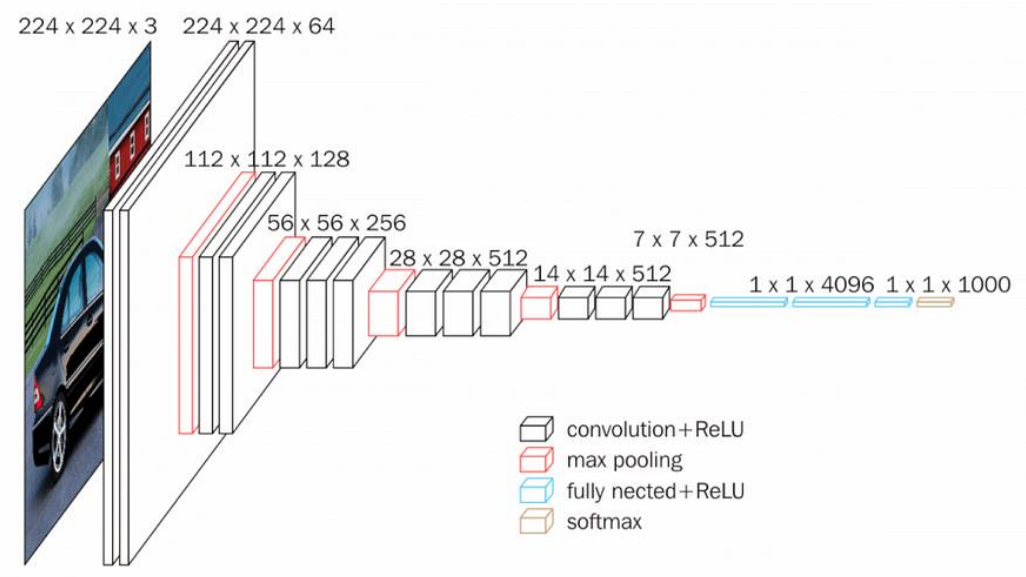

Figura 4: Estrutura da arquitetura VGGNet [20].

Por fim, a GoogleNet foi uma arquitetura desenvolvida pela Google também para reconhecimento de imagens. Dessa vez, utilizam-se módulos chamados de inception que são

compostos por camadas de convolução 1x1, 3x3, 5x5, seguidos de uma camada de pooling. Essa estratégia permite que a rede aprenda representações complexas e hierárquicas [15].

Fuad et al. [30] citam o desenvolvimento de um software de verificação e identificação facial que utiliza a arquitetura GoogleNet para extração de características em forma de vetores. O projeto criado por Yang et al. [31] conhecido como *Neural Aggregation Network* (NAN) utiliza dois blocos de atenção para agregar os vetores e associar um peso a eles. Após isso, para a verificação facial, foram utilizadas redes siamesas e minimização média de perda contrastiva. E para a identificação, foi colocado uma camada FC seguida de softmax e minimização média de perda de classificação.

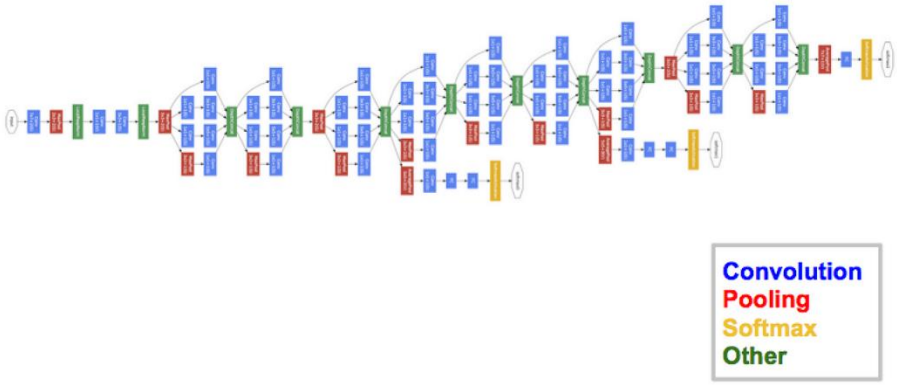

Figura 5: Estrutura da arquitetura GoogleNet [32].

2.2.3 Técnicas Específicas e Avançadas

Além dos estudos sobre as arquiteturas amplamente utilizadas, é importante destacar que houve artigos que exploram técnicas à parte.

Dada a especificidade dos problemas solucionados pelos artigos da IEEE, foi notável que as tecnologias utilizadas variaram bastante. Dentre elas, houve técnicas complexas que abordavam problemas avançados, mas também haviam resoluções simples para problemáticas igualmente simples. A Tabela 2 mostra a diversidade de abordagens encontradas na pesquisa, com o tema do seu respectivo texto associado.

Tabela 2: Abordagens utilizadas em artigos de pesquisa encontrados na IEEE.

Artigo	Problema Estudado	Abordagem Escolhida
[33]	Reconhecimento Tridimensional	DepthNet e Multi Modal RGB-D CNN
[34]	Reconhecimento Tridimensional	Nuvem de Pontos - PointFace
[35]	Amostra Única por Pessoa	Framework SSLRR com Modelo P+V (IDGL)
[36]	Melhoria de Aprendizado em Deep Learning	EnhanceFace CNN
[37]	Intensificação de Atenção	Attention Erasion (AE) com Attention Center Loss (ACL)[1]
[38]	Faces Obstruídas	FROM CNN
[39]	União de Reconhecimento Holísticos e de Faces Obstruídas	Face T-B com Transformer
[40]	Reconhecimento de Expressão em Ambiente Online	ResNet-50 com Convolutional Block Attention Mechanism (CBAM)
[41]	Associação de Imagens em Formatos Heterogêneos	Domain Specific Unit DCNN
[42]	Associação de Imagens em Formatos Heterogêneos	Mutual Component Convolutional Neural Network (MC-CNN)
[43]	Identificação de Gênero para Sistemas Mobile	MobileNet CNN
[44]	Reconhecimento para Sistemas Embarcados	GhostFaceNet MTCNN
[45]	Desconsideração de Imagens de Baixa Qualidade durante o treino	LightQNet
[46]	Imagens de Baixa Resolução	MIND-Net CNN
[47]	Reconhecimento de Frames de Vídeo	SiamSRC CNNs

Chiu et al. [33] propuseram dois modelos de FR tridimensional. O primeiro, chamado DepthNet utiliza uma arquitetura encoder-decoder, onde ele dispõe de uma *Generative Adversarial Network* (GAN) com um codificador e três decodificadores, um para a as cores, outro para os relevos do rosto e o último para as características dos órgãos faciais para treinar os discriminadores. O outro modelo se chama Multi Modal RGB-D que prediz, a partir de um classificador, o estilo e a identidade de dois componentes separados: as cores e o relevo do rosto. Porém, o classificador de estilo sempre considera também o mapa de órgãos faciais da pessoa. Já Jiang et al. [34] encaram o mesmo desafio de uma forma diferente. Eles utilizam uma base de dados de nuvem de pontos (*point cloud*), como mostrado na Figura 6, e treinam dois *encoders* que transformam essas informações em um *embedding* cada. Em seguida, é feita uma comparação entre os dois *embeddings* e aplicada uma função de perda nos *encoders* para que eles aprendam a reconhecer e padronizar faces iguais. E por fim, esses *embeddings* são

[1] Esta técnica é aplicada durante o treinamento, portanto, não corresponde a uma tecnologia individual.

transformados em um vetor de identidade contendo apenas as características principais da face, tendo uma avaliação *softmax* para que o encoder aprenda os padrões do vetor de identidade.

Pang et al. [35] destacaram um problema comum em muitas bases de dados: a contaminação da base por amostras únicas por pessoa. Eles afirmam que muitas vezes pode haver uma amostra com problemas externos como iluminação, posição e afins. Para resolver, ele aplica um sistema de reconhecimento semi-supervisionado muito densamente complexo para gerar um protótipo de resultado e aplicar em um sistema P+V, onde P é o protótipo e V é o dicionário de rótulos. No fim, se o rótulo gerado se aplica a face, ele mantém, caso contrário, ele altera a identificação na base de dados.

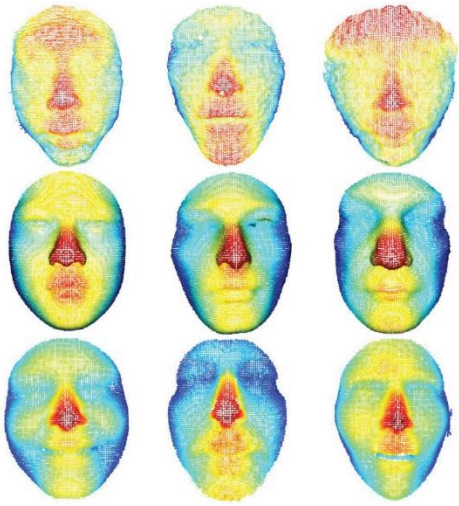

Figura 6: Representação de três bases de dados de nuvens de pontos utilizadas para reconhecimento tridimensional [34].

Um estudo que aborda a criação de uma nova *loss function* foi produzido por Wang et al. [36]. Neste artigo, é denotado que uma parte dos conjuntos de dados utilizados para treinamento de redes profundas utiliza uma estratégia de divisão de imagens por seu nível de dificuldade entre fácil e difícil. No entanto, haveria um problema no reconhecimento das difíceis, por abranger imagens com problemas externos. A solução para isso foi dividir o conjunto de imagens difíceis em dois grupos: difíceis e semi-difíceis. Além disso, foram criadas novas *loss functions* para tratar as faces de forma diferente dependendo de sua dificuldade.

Alguns estudos sobre FR sob oclusão (OFR do inglês *Occluded Face Recognition*) foram identificados, os quais serão abordados a seguir. Wang e Guo [37] propõem um método de aprimoramento do treinamento de redes neurais para FR. A união de um mecanismo de apagamento de atenção (AE do inglês *attention erasion*) com um sistema de perda de centro de atenção (ACL do inglês *attention center loss*) gera seu modelo chamado ANN-Net. O motivo da pesquisa foi a melhoria da detecção facial que utilize base de dados com poucas imagens fora do comum, como faces ocultas, posições diversas e iluminação variante. O AE é responsável por receber um mapa de atenção de uma CNN local e apagar uma parte pseudo-aleatória dessa imagem. O ACL recebe esse resultado e balanceia a atenção removendo de possíveis pontos irrelevantes. O treinamento é feito várias vezes considerando várias CNNs

locais. A estratégia adotada por Qiu et al. [38] para detectar faces oclusas é diferente. Dessa vez, um pequeno conjunto de imagens é inserido em uma *Feature Pyramid Network*, como mostra a Figura 7, baseado em uma versão refinada da arquitetura ResNet-50, para extrair um mapa de características das faces. O modelo é baseado no treinamento de um módulo chamado de *Mask Encoder* para que detecte precisamente a posição da oclusão na imagem na forma de máscara. A imagem mascarada é então exposta ao reconhecimento facial padrão.

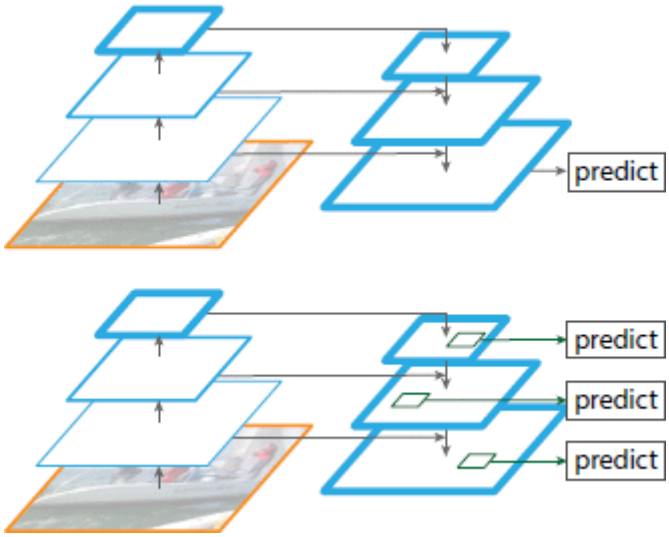

Figura 7: Estrutura da *Feature Pyramid Network* [48].

Pedro Neto et al. [49] dissertam em seu trabalho que, devido ao vírus COVID-19, pesquisas sobre FR com máscaras (MFR, do inglês *Maked Face/Facial Recognition*) foram amplamente estudadas, ramificando o tema OFR. Por tal motivo, eles comparam o estado da arte das duas tecnologias e do FR atual, visando a utilização do MFR em problemas de OFR. Para isso, eles reuniram 8 algoritmos de FR comuns, 10 de faces oclusas e 6 para rostos mascarados e executaram testes nos mesmos conjuntos de dados. O resultado foi a confirmação de que os softwares de MFR não só executam bem a função, como são mais leves que os outros.

Uma última pesquisa encontrada nesta área de estudo foi o modelo proposto por Zhu et al. [39] que é capaz de unir a tecnologia de MFR com reconhecimento holístico utilizando *vision transformers* (ViT). Ele utiliza dois métodos baseados em equações complexas para que a tecnologia funcione, os quais se chamam *prompt tuning* e *prompt pooling*. O modelo consegue manter uma precisão de até 99.90%, rivalizando com os algoritmos de estado da arte.

Aly et al. [40] criaram um sistema que, periodicamente, registra as faces de alunos durante aula online e adapta a aula do professor baseado nas expressões dos discentes. Eles utilizam um mecanismo chamado de *Convolutional Block Attention Mechanism* (CBAM) que representa a união entre o *Channel Attention Mechanism* (CAM) e o *Spatial Attention Mechanism* (SAM), módulos responsáveis pela definição de pesos na rede. O sistema é baseado na arquitetura ResNet-50.

Inúmeras pesquisas foram concluídas em um outro tópico recorrente no contexto deste trabalho conhecido como reconhecimento facial de mídias heterogêneas, ou seja, a identificação através da comparação de mídias diferentes, como fotografias, desenhos e imagens próximas ao infravermelho [50]. Neste âmbito, Freitas Pereira et al. [41] utilizam redes siamesas e triplet loss para implementar seu modelo denominado *Domain Specific Unit*, que entrega resultados comparáveis ao estado da arte. Um segundo desenvolvimento foi proposto por Deng et al. [42], onde é utilizado uma GAN juntamente de uma CNN contendo um módulo chamado *Mutual Component Analysis* (MCA), que verifica elementos mútuos entre duas entradas. A arquitetura da CNN utilizada nesse trabalho era baseada na ResNet-41.

As técnicas anteriores são de enorme importância para o avanço do reconhecimento facial e apresentam problemas e soluções muito úteis para a sociedade. No entanto, tais abordagens não tem tanto a agregar ao problema apresentado neste trabalho, seja por motivos de complexidade, como os algoritmos de reconhecimento tridimensional, seja por demasiada especificação, como os softwares para oclusões ou HFR. As tecnologias mais relacionadas com este trabalho e o motivo dessa relação serão melhor abordados a seguir.

Para criar um sistema de FR que reconhece o gênero de uma pessoa através da face, Greco et al. [43] desenvolveram um modelo baseado na arquitetura MobileNet [51]. A escolha se deu pela priorização dos sistemas móveis e embarcados e é este o motivo da importância deste artigo. O sistema a ser produzido neste trabalho tem altas chances de ser instalado em um aparelho embarcado e esse texto referencia uma tecnologia capaz de lidar com isso. O software criado por Greco et al. atingiu uma porcentagem de precisão de 98.73% sem a utilização de qualquer acelerador de hardware. Este estudo influenciou em outras pesquisas sobre identificação de gênero em aparelhos de baixo custo [52], [53].

Assim como na pesquisa anterior, Alansari et al. [44] produziram um modelo de FR para dispositivos de baixo custo. Segundo os autores, a complexidade computacional necessária para executar o software é de 60-275 MFLOPs, quando a maioria dos sistemas de FR que utilizam CNN necessitam de centenas de milhares de MFLOPs. A arquitetura base utilizada também foi um modelo de MobileNet, a GhostNet. Na verdade, o modelo GhostFaceNet proposto neste artigo é composto pela GhostNetV1 e V2, sendo que a diferença entre as duas foi a adição de um mecanismo de atenção na segunda versão. O motivo da importância deste texto é o mesmo do anterior: economia de recursos. A precisão deste modelo alcançou 98.72%.

Mais um problema discutido por Chen, Yi e Lv [45] é a utilização de imagens de baixa resolução nas bases de dados fornecidas para os treinamentos. Neste caso, os autores, além de fazerem uma busca sobre qualidade de dados, desenvolveram uma técnica de remoção de imagens de baixa resolução antes do treino. Foi determinado um *threshold* que indica a qualidade da face, e as fotografias que foram escolhidas passaram por uma CNN também baseada em um modelo da MobileNet. Este artigo se destacou pela melhoria da base de dados automatizada antes do treinamento.

Low, Teoh e Park [46] criaram um modelo de reconhecimento facial que lida com imagens de baixa resolução, como gravações de câmera de segurança. Esse método utiliza um conjunto de

dados de imagens holísticas com um grande número de registros, uma cópia desse conjunto com resolução artificialmente reduzida e um conjunto de imagens reais de baixa qualidade. Essas entradas são processadas por uma CNN de duas vias (*dual-streamed*), onde as duas primeiras fontes passam juntas por um canal e a última é processada sozinha. No processo de identificação, é feita uma extração de características mútuas para que locais genéricos das faces não sejam considerados. No fim, é aplicado um classificador *softmax* para avaliar a rede. A arquitetura da CNN utilizada foi baseada na ResNet-50 e atingiu uma precisão levemente inferior aos modelos apresentados. Esse artigo foi considerado importante pela sua técnica simples de processar imagens com baixa resolução, imaginando que possa haver problemas desse tipo neste trabalho.

O reconhecimento de rostos em vídeo é o foco da pesquisa de Mokhayeri e Granger [47]. Para resolver este problema, eles criaram uma arquitetura utilizando duas CNNs para extrair as características faciais de imagens originais e modelos processados a partir dessas imagens. Esse par de características é comparado por um módulo chamado SiamSRC, formado por CNNs siamesas. As redes são todas baseadas em ResNet-50. Este modelo foi destacado por sua capacidade aprimorada de lidar com imagens ao vivo.

2.2.4 Funções de Ativação e Perda

A fim de aprimorar o treinamento dos modelos, são utilizadas funções para tomadas de decisão ou para ajuste do algoritmo, conhecidas como *activation function* e *loss function*, respectivamente. A primeira recebe um valor de entrada e calcula sua saída baseada em uma função matemática, essa saída é a decisão. A segunda é aplicada em modelos com *reinforcement learning*, avaliando as respostas do software e ajustando para melhor ou pior.

As principais funções de ativação descritas nos trabalhos selecionados serão descritas a seguir.

- **Sigmóide:** É uma das funções mais utilizadas para ativação. Isso se deve à capacidade de normalização dela. Como pode ser observado na Figura 8, a saída dessa função sempre está entre os valores 0 e 1. Mas é perceptível que a maioria dos valores de entrada resulta em uma saída muito próxima aos extremos, por isso, essa função é usada principalmente para classificação binária.

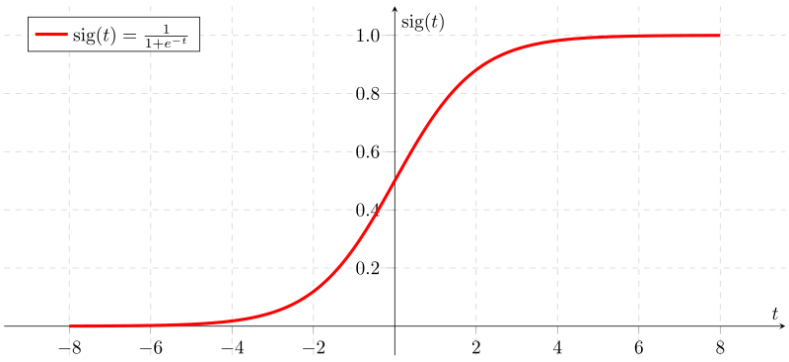

Figura 8: Representação da função sigmóide em um plano cartesiano.

- **Softmax:** Essa função também atua gerando valores entre 0 e 1, mas, resolvendo o problema da classificação, as saídas internas são muito mais acessíveis, podendo, assim, lidar com multi-classes. Na Figura 9, está claro que o formato das funções é muito parecido, no entanto, este método utiliza probabilidades para cada classe, fazendo com que a soma de todas as saídas seja igual a 1, utilizando de forma mais efetiva os valores internos.

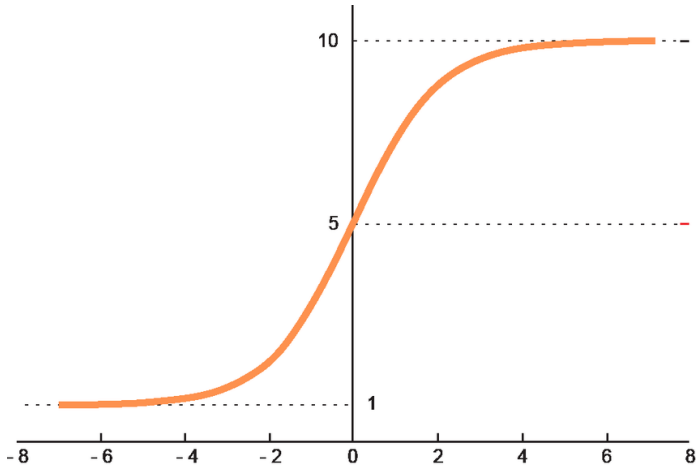

Figura 9: Representação da função softmax no plano cartesiano [54]

- **Rectified Linear Units (ReLU):** Também é uma função amplamente utilizada, mais encontrada em camadas internas das redes neurais. Seu funcionamento é baseado em uma função linear para valores de entradas positivos e é constante em 0 para todos os valores de entrada negativos, como pode ser visto na Figura 10. Ela é muito efetiva para reduzir o desaparecimento do gradiente descendente, mas quando um neurônio que a usa resulta em constantes zeros, ele dificilmente irá se recuperar, gerando um neurônio morto.

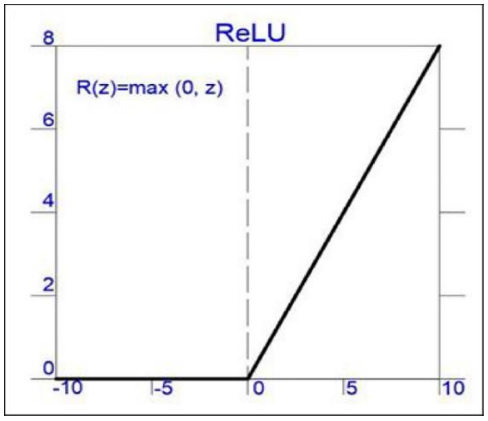

Figura 10: Representação da função ReLU no plano cartesiano [55].

Já para as funções de perda, a utilização é muito relativa. Mas dentre os algoritmos observados nesta revisão, pode-se destacar aquelas que mais foram mencionadas: *Triplet Loss, Contrastive Loss, Marginal Loss* e *Softmax Loss*. Normalmente, cada uma delas é responsável por resolver um problema muito específico de avaliação de resultados.

Em relação aos algoritmos a serem utilizados e sua efetividade nesta pesquisa, serão feitas experimentações com as técnicas destacadas para avaliar seus resultados para este problema.

2.3 Datasets

Os conjuntos de dados ou *datasets* são elementos imprescindíveis para o desenvolvimento de uma rede neural. O algoritmo os usa para aprender quais rótulos determinar para cada entrada e isso só é possível a partir de um *dataset* denso e extenso. A qualidade desses dados são fatores cruciais para determinar a qualidade do produto final, portanto é de suma importância que haja uma seleção de alguns dos melhores desta categoria. Portanto, considerando os projetos do estado da arte, serão determinados os conjuntos de treino a partir daqueles mais utilizados nas pesquisas.

Este tópico tem como objetivo explicar as principais diferenças entre cada um e a justificativa da seleção.

Os conjuntos de dados a seguir foram amplamente encontrados nas pesquisas revisadas. Vale lembrar que já foi aplicado um filtro e a grande maioria dos *datasets* mencionados nos artigos foi descartado, dadas as suas funcionalidades. Por exemplo, não faz sentido selecionar um *dataset* focado em faces oclusas, uma vez que esse não é o foco desta pesquisa. Dito isso, os seguites conjuntos se destacaram:

- **VGGFace2:** É uma atualização do *dataset* VGGFace [56], que foi planejado para lidar com variações de idade, iluminação e posição. A primeira utilidade não é interessante para este trabalho, mas as outras, por outro lado, afetam diretamente o reconhecimento dos alunos. Esse conjunto de dados possui mais de 3 milhões de

identidades registradas. Grande parte dos usuários reduz essa quantidade para ter uma melhor normalização e balanceamento.

- **Labeled Faces in the Wild (LFW):** Foi o *dataset* mais popular entre os artigos selecionados. A grande maioria das pesquisas utilizaram esse conjunto para comparar seus modelos com os algoritmos de estado da arte. De acordo com Oloyede, Hancke e Myburgh [17], ele funciona muito bem em condições negativas como variação de pose, iluminação, resolução e expressão. Uma vez que é possível que os registros de imagens dos discentes não sejam supervisionados, é importante que o algoritmo saiba verificar faces nessas condições. O *dataset* conta com 13233 imagens de 5749 indivíduos únicos com grande variação de condições externas [29].
- **CASIA WebFace Dataset:** O *dataset* proposto por Yi et al. [57] também é capaz de apresentar resultados na resolução de problemas de condições externas. Este conjunto é formado por 494414 imagens de 10575 pessoas, provando ser bastante denso. No entanto, de acordo com os estudos de Yi et al., o *dataset* consegue manter as precisões de outros conjuntos utilizando os mesmos métodos.

2.4 Observações do Capítulo

Este capítulo teve como objetivo apresentar as tecnologias utilizadas no desenvolvimento de algoritmos de reconhecimento facial. Após todos os estudos, foram determinados alguns sistemas mais compatíveis com a proposta desta dissertação, assim como conjuntos de dados mais relacionados com o problema. Foram apresentados métodos que não utilizam redes neurais, assim como aqueles que utilizam. Também foram abordadas as arquiteturas mais famosas como AlexNet, ResNet, GoogleNet e VGGNet. E por fim, há uma breve apresentação das funções designadas para tomada de decisão e avaliação do algoritmo.

Todas essas informações foram extraídas de uma pequena revisão literária, a qual utiliza elementos da metodologia PRISMA, a fim de responder às perguntas mencionadas no capítulo 1.

3 Metodologia

Nesse capítulo, são detalhadamente apresentados os métodos e as ferramentas utilizadas para a conclusão deste estudo, destacando-se a importância de cada um deles para o desenvolvimento do trabalho. A organização dos elementos metodológicos é fundamental, pois estabelece uma linha de orientação clara a ser seguida, que funcionará tanto como um guia para as etapas subsequentes deste projeto quanto como uma referência para leitores futuros. Esses leitores poderão, assim, compreender não só o processo de desenvolvimento, mas também os critérios adotados e as justificativas por trás de cada escolha metodológica. A estrutura do capítulo está organizada em seções específicas, cada uma voltada para discutir métodos e abordagens individuais, assim como ferramentas cruciais para o desenvolvimento, facilitando uma leitura segmentada e aprofundada de cada aspecto.

3.1 Ferramentas e Recursos

Algumas tarefas no âmbito tecnológico podem exigir muito tempo e recursos computacionais para serem completadas. No desenvolvimento de redes neurais mais robustas, essa necessidade cresce surpreendentemente, uma vez que, para melhorá-las, é preciso cada vez mais de maiores bases de dados que, por sua vez, irão requerir muito mais poder de processamento para serem analisadas.

Diante desse problema, os facilitadores são cruciais para que o desenvolvimento não tenha prazos comprometidos. A seguir, as ferramentas facilitadoras serão detalhadamente descritas, assim como sua importância para o estudo.

3.1.1 Recursos Online

A internet atualmente é muito ampla e pode mostrar enorme utilidade se manejada da forma correta. Dentre as inúmeras ferramentas úteis encontradas nos *websites*, pode-se citar algumas que fazem total diferença no desenvolvimento da Inteligência Artificial, como o Kaggle e o Google Colab.

3.1.1.1 Kaggle

O Kaggle [10] é uma das plataformas mais reconhecidas e utilizadas para a produção e experimentação de algoritmos de *machine learning*. Com o objetivo de potencializar o aprendizado nesse tema, o *site* oferece uma vasta gama de recursos valiosos para esse tipo de tarefa. Os usuários da plataforma desenvolvem softwares voltados para o mundo da aprendizagem automática e a usam para compartilhar seus projetos, assim como ideias e materiais. Essa conexão da comunidade auxilia novos estudantes a aumentarem sua produtividade e conhecimento nesse aspecto.

Um item imprescindível para o andamento de um projeto de IA é o *dataset*, que pode ser encontrado com várias configurações e objetivos no *site*. É comum que ao publicar seu projeto na plataforma, o desenvolvedor também exponha seu *dataset* e suas bibliotecas, ou informe qual foi utilizada. Dessa forma, o leitor pode fazer o *download* do arquivo e aplicá-lo em seu próprio desenvolvimento.

Outro elemento que pode fazer toda a diferença durante o processo de produção do software é o *notebook*. Neste material, é onde os desenvolvedores explicam a funcionalidade de seus códigos, assim como as limitações e requisitos. Para o desenvolvimento deste estudo, foram analisados muitos *notebooks*, a fim de obter conhecimento prático dos códigos de redes neurais convolucionais. Por ter sido relacionado a um estudo individual para auxiliar na implementação, não serão citados tais códigos, mas é necessário informar a importância da ferramenta para a pesquisa.

3.1.1.2 Google Colaboratory

Criado pelo Google em 2017, o Colaboratory [58] veio com o objetivo de disponibilizar um ambiente de desenvolvimento de *machine learning* com armazenamento em nuvem. Essa característica da plataforma é o que a faz tão especial, uma vez que projetos com baixo orçamento (o que resulta em computadores com baixo poder de processamento) poderiam utilizá-la sem medo de perder dias em um único treinamento. É importante denotar que as rotinas de treino de *machine* e *deep learning* demoram muito tempo para serem concluídas, devido ao grande volume de informações que precisam ser processadas pela rede. Ao usar um ambiente online para criar os softwares, o tempo era poupado, já que o processamento do servidor disponibilizado é suficientemente elevado, para a maior parte dos casos.

Essa não é uma ferramenta gratuita, o que pode ser um ponto negativo forte para grande parte dos pesquisadores. Mas, se um sacrifício for viável, ela pode poupar um tempo significativo a

depender dos recursos físicos disponíveis e do nível de exigência do treinamento da rede. Para esse projeto, essa plataforma foi utilizada justamente por esses motivos e fez grande diferença para que os prazos fossem respeitados, apesar de não ter sido utilizada durante toda a implementação.

3.1.1.3 Microsoft Azure

Similarmente ao Google Colaboratory, o Microsoft Azure [59] também oferece um ambiente computacional hospedado em nuvem. A diferença é que o Azure tem muito mais ferramentas disponíveis e muitas delas específicas para o desenvolvimento de *machine learning*. No entanto, isso traz desafios de complexidade, já que a utilização adequada desses recursos não é tão simples. Além disso, há uma amostra grátis generosa para os novos usuários, mas o valor integral é bem maior que o da ferramenta do Google. Dessa forma, caso haja problemas financeiros, a continuidade dos serviços Azure fica em risco. Outra diferença, que talvez seja a principal, é que no Azure, o pagamento é após o uso, ou seja, o usuário paga apenas o que foi utilizado, diferentemente do Colab que utiliza mensalidades. Dito isso, ambas as ferramentas foram bastantes úteis para o prosseguir da implementação, uma vez que o problema computacional foi recorrente.

3.1.2 Bibliotecas

Na comunidade da programação, sempre foi muito comum o compartilhamento de códigos bem-sucedidos, pois é um hábito importante para o avanço tecnológico. Muitos desses projetos são divulgados como forma de bibliotecas, que compactam um conjunto de arquivos de código para executar funções específicas. Hoje em dia, a programação de softwares extensos e robustos ficou cada vez mais dependente dessas bibliotecas, uma vez que o gasto de tempo é drasticamente minimizado ao pular etapas que já foram resolvidas por outra pessoa, direcionando, assim, o foco do desenvolvedor para o problema que realmente importa para ele.

Para a aprendizagem automática e profunda, já existem algumas bibliotecas bastante consolidadas, como o Tensorflow, Keras, Pytorch, etc., e sua importância deve ser destacada.

3.1.2.1 Tensorflow

O TensorFlow é uma biblioteca gratuita criada pelo Google para facilitar a criação e o treinamento de modelos de inteligência artificial e aprendizado de máquina. Desde que foi lançada em 2015, tornou-se uma ferramenta popular entre cientistas de dados e desenvolvedores por ser flexível e eficiente, permitindo criar modelos complexos, como redes neurais profundas, de maneira prática. Um dos pontos fortes do TensorFlow é a forma como ele organiza as operações em grafos de dados [60], o que facilita a distribuição do trabalho entre CPUs e GPUs, acelerando o processamento de grandes volumes de dados. Além disso, ele é compatível com várias linguagens de programação, como Python, JavaScript e C++, o que amplia ainda mais suas possibilidades de uso.

A grande vantagem do TensorFlow está em sua versatilidade. Ele oferece ferramentas como o TensorFlow Extended (TFX), que ajuda a criar fluxos de trabalho completos para modelos de aprendizado de máquina, e o TensorFlow Lite, ideal para levar esses modelos a dispositivos móveis e integrados. A biblioteca conta com uma comunidade ativa e muitos recursos de apoio, como tutoriais e modelos pré-treinados, que economizam tempo e ajudam na experimentação. Com essa combinação de recursos e o suporte da comunidade, o TensorFlow se destaca como uma das melhores opções para quem quer desenvolver soluções de IA com alta performance e em larga escala.

Para um melhor aproveitamento dos recursos do tensorflow, foi necessária a instalação de uma versão que contempla o uso da GPU, utilizando as ferramentas *Computer Unified Device Arquitecture* (CUDA) e o *CUDA Deep Neural Network* (cuDNN), para potencializar o poder computacional aplicado ao treinamento da rede. Após a aplicação dessa estratégia, o tempo deixou de se tornar um problema, pois a rotina finalizava muito mais rapidamente, no entanto, um desafio de memória insuficiente veio à tona por processar uma quantidade massiva de imagens do *dataset*. O grande problema foi que a GPU utilizava apenas a VRAM, o que limitou um pouco o uso da memória.

3.1.2.2 Keras

O Keras é uma biblioteca de código aberto criada para simplificar a vida de quem desenvolve redes neurais e modelos de aprendizado profundo. Ela é feita para ser intuitiva e fácil de usar, ideal tanto para quem está começando na área quanto para especialistas em ciência de dados que querem agilidade nos experimentos. O objetivo do Keras é permitir que desenvolvedores testem rapidamente diferentes modelos de aprendizado profundo, evitando configurações complexas e rotinas técnicas que, em outras bibliotecas, poderiam exigir mais tempo e conhecimentos avançados.

Desde que foi integrada ao TensorFlow como a API de alto nível oficial, o Keras tornou-se uma das formas mais práticas de construir e treinar redes neurais, sem precisar mexer com operações detalhadas. Com apenas algumas linhas de código, é possível configurar uma rede neural, definir as camadas e ajustar parâmetros essenciais, como a taxa de aprendizado e o número de épocas. Essas características são importantíssimas durante o desenvolvimento de uma inteligência artificial, devido à agilidade da produção.

3.1.2.3 Pytorch

Pytorch é uma solução alternativa ao Tensorflow. A biblioteca criada pelo Facebook tem muitas vantagens em relação a praticidade e conforto do desenvolvedor. A compatibilidade com Python (linguagem de programação muito utilizada em treinamentos de *machine learning*) e a execução dinâmica são diferenciais nessa ferramenta. É, também, uma solução muito viável para o desenvolvimento do software, mas não foi encontrada tanto nas fontes base para estudo da implementação. A diferença, no geral, seria mínima, então não acarretou em problemas no modelo final.

3.1.2.4 Outras bibliotecas básicas

Foram utilizadas algumas outras bibliotecas que merecem ser citadas, por serem tão importantes para a implementação: matplotlib e numpy.

Para quem já é familiarizado com desenvolvimentos como esse, a exposição de tais ferramentas pode ser considerada exagerada. Mas é importante denotar que todo o funcionamento das redes foram baseadas em *arrays* trabalhados pelo numpy, o que novamente permitiu a economia de tempo.

3.1.3 Large Language Models

Para concluir com sucesso um software de inteligência artificial, especialmente quando o objetivo é o reconhecimento facial, muitos desafios devem ser superados. E para aqueles que aprendem durante o desenvolvimento, esses desafios são muito mais difíceis. Um dos grandes obstáculos é a configuração da arquitetura do modelo de forma que haja precisão suficiente. Como já foi dito, muitas vezes, o tempo necessário para executar uma rotina de treino é enorme e ainda assim, frequentemente o desenvolvedor irá se deparar com um erro no final, consumindo seu precioso tempo. Para resolver isso, foi utilizada uma outra espécie de ferramenta: os Large Language Models (LLM's).

3.1.3.1 Conceito

Os LLM's são modelos de inteligência artificial focados em identificação, interpretação e produção de textos, podendo ser também adaptados para processar imagens e outras mídias. Normalmente, são utilizadas Redes Neurais Recorrentes (RNN – do inglês Recurrent Neural Networks) com um grande número de informações e textos da internet. Elas são capazes de manter uma conversa por muito tempo e responder perguntas com precisão, apesar de nem sempre estarem corretas. Geralmente, são re-alimentadas com novos conjuntos de dados, de forma que sempre fiquem atualizadas sobre as informações do mundo.

3.1.3.2 Utilização

O uso dessas ferramentas foi bastante polêmico nos últimos anos, após muitos estudantes utilizarem-na para evitar o esforço de produzir textos e responder atividades. Desde então, levantou-se o questionamento se isso não estaria prejudicando o pensamento crítico dos alunos [61]. Essa ideia acabou influenciando a comunidade de tecnologia da informação em relação ao desenvolvimento dos códigos. Houve então uma grande discussão sobre como o uso desse tipo de recurso seria apropriado nas produções, mas alguns autores chegam a conclusão de que ele pode ser usado efetivamente como assistente de programação [61], [62], [63].

Portanto, LLM's como ChatGPT e Llama 3 foram utilizados como assistentes para um aprendizado mais rápido sobre o funcionamento dos códigos de inteligência artificial, uma vez que, apesar das bibliotecas facilitadoras, continuam necessitando de um tratamento de dados

e de definições um pouco complexas. Dessa forma, pode-se afirmar que esse projeto não utilizou os modelos de linguagem como ferramenta base, mas sim como auxiliar de aprendizado de algumas particularidades do tema.

3.1.4 Estudo de Códigos e Estratégias

Com a união de todos os recursos citados acima, houve então um estudo intensivo sobre o funcionamento das implementações de inteligências artificiais voltadas para o tema. *Notebooks* do Kaggle e códigos escritos pelas LLM's foram importantíssimos para o desenvolvimento de aplicações práticas das técnicas estudadas nesse projeto. Como será observado em um tópico futuro, as abordagens analisadas no Estado da Arte foram analisadas e testadas para o sistema que está sendo visado.

3.2 Métodos Utilizados

Como dito anteriormente, as estratégias são extremamente importantes para manter objetivos e o caminho até eles claramente definidos. Esse caminho pode envolver formas de se executar uma determinada pesquisa, as fontes utilizadas, a definição do objeto de estudo, abordagens para a execução de um determinado desenvolvimento, etc. Nesse caso, a pesquisa teórica já foi concluída, portanto, os métodos apresentados a seguir estão diretamente relacionados com a produção do desenvolvimento e suas particularidades. Haverá um capítulo no futuro exclusivo para a dissertação das estratégias teóricas e práticas aplicadas ao software, onde será definido o modelo proposto. Neste item, será abordado o planejamento inicial do estudo de desenvolvimento e algumas de suas conclusões.

3.2.1 Abordagens do Estado da Arte

Uma revisão dos estudos recentes sobre as abordagens mais viáveis para a produção de um sistema de reconhecimento facial foi elaborada no capítulo 2. Dentre elas, algumas se destacaram e um procedimento de avaliação mais aprofundada dessas estratégias foi planejado e iniciado. A avaliação envolveu a produção de um código correspondente ao que foi descrito em cada um dos artigos selecionados e a aplicação (ou tentativa) direta do software no cenário prático estudado neste trabalho.

3.2.1.1 Reconhecimento de Gênero – MobileNET

Como citado anteriormente, Greco et al. [43] desenvolveram um modelo de reconhecimento de gênero que utiliza a rede MobileNet como base. Eles aplicaram algumas modificações nas dimensões das imagens, a quantidade de camadas e o multiplicador de largura. Esse modelo foi escolhido por ser capaz de ser executado em equipamentos de pequeno porte como celulares

e sistemas embarcados. Isso se deve ao fato de que a MobileNet utiliza operações *depthwise* e *pointwise*, gerando assim um efeito de *bottleneck.*

A operação de *depthwise* envolve a aplicação de um filtro convolucional 3x3 individualmente em cada canal da imagem. Por exemplo: em uma figura RGB, o filtro seria aplicado a cada canal de cor separadamente. Isso ocorre para facilitar as operações e cálculos realizados no momento da convolução. Dessa forma, os canais são separados e precisam novamente ser agrupados para que a imagem seja analisada pela rede corretamente após a convolução.

Para agrupar mais uma vez os canais, a operação *pointwise* é feita. Ela consiste em aplicar um filtro 1x1 que fará um cálculo justamente para unir as saídas novamente. E isso gera um efeito chamado *bottleneck*, tendo esse nome por haver uma expansão e uma redução do número de canais e das dimensões da rede. Essas estratégias permitem que o processamento seja muito menos exigente e os aparelhos de menor porte podem então executar o software.

O modelo MobileNet então, ficou a ser testado em alguma rotina de treino a ser ainda estudada, pois as abordagens de treino desse artigo envolvia tarefas de classificação, o que impede que possam ser utilizadas neste trabalho. Um modelo de classificação não funcionaria para o nosso objetivo, pois ele apenas definiria a entrada entre duas classes (no caso do texto em questão: masculino e feminino). As classes devem ser definidas ainda no treino, pois o modelo deve aprender a reconhecer cada tipo de resposta em cada imagem. O problema é que em um sistema dinâmico como o proposto aqui, a quantidade de respostas não pode ser definida previamente, muito menos as respostas em si, pois cada saída seria um aluno específico e as bases de dados podem ser atualizadas com uma frequência desconhecida. Por esse motivo, todo e qualquer modelo de classificação a partir de então foi desconsiderado, mas a rede proposta foi analisada posteriormente.

3.2.1.2 GhostFaceNets

A GhostFaceNet é uma rede neural voltada para reconhecimento facial, inspirada na GhostNet e adaptada com alguns ajustes bem específicos. A ideia central por trás dela é utilizar uma estrutura leve para processamento, mas ainda focada em extração de características faciais. Para isso, foram feitas três principais modificações em relação à GhostNet: eles ajustaram a camada de saída para uma convolução em profundidade global (GDC – *Global Depthwise Convolution*) modificada, substituíram a função de ativação ReLU pela PReLU e adaptaram os módulos *Squeeze and Excitation* (SE), responsáveis por aumentar a atenção aos pequenos detalhes, para otimizar a capacidade de discriminação da rede.

No caso da camada GDC modificada, a GhostFaceNet substitui a camada de *Global Average Pooling* (GAP) comum (usada geralmente em classificação) para tentar evitar que todos os canais da saída tenham o mesmo peso. Isso ajuda a rede a enfatizar canais que trazem informações mais relevantes para o reconhecimento facial. A função PReLU, por sua vez, permite valores negativos nas ativações, o que contribui para uma maior complexidade de aprendizado e melhora o desempenho da rede. Os módulos SE foram ajustados para

potencializar a percepção em canais específicos, uma técnica que normalmente intensifica a capacidade de captar nuances nas imagens.

Ainda que tenha essas adaptações interessantes, a GhostFaceNet não foi escolhida para este trabalho. Se fosse necessário recorrer a uma rede leve, a MobileNet já seria mais vantajosa pela simplicidade e eficiência. Com suas convoluções depthwise e pointwise, a MobileNet consegue reduzir o custo computacional sem muitas complicações adicionais. Além disso, os ajustes feitos na GhostFaceNet são voltados para classificações bem específicas, enquanto aqui o foco é no reconhecimento facial, que tem requisitos diferentes.

3.2.1.3 LightQNet

De forma semelhante à rede anterior, esse método tem como objetivo a priorização do desempenho no reconhecimento de imagens de baixa qualidade. Chen et al. [45] propuseram um modelo baseado no MobileNetv3 com altereações em algumas estratégias internas da rede. A primeira delas é a aplicação do *Identification Quality Loss* (IDQ), que seleciona imagens de baixa qualidade, as quais seriam mais difíceis do sistema reconhecer, e as adiciona na entrada do modelo. Dessa forma, a rede terá mais capacidade de detectar rostos em imagens com baixa resolução. A segunda abordagem dos autores foi a *Branch-Based Quality Distilation*¸ que utiliza redes maiores para treinar a rede alvo. O que mais foi interessante nesse artigo, foi a primeira abordagem, que foi testada no software final. No capítulo 4, haverá uma explicação mais aprofundada sobre a aplicação desse método.

3.2.1.4 MIND-Net

A MIND-Net foi projetada para melhorar o reconhecimento facial em imagens de baixa resolução. Esse modelo utiliza uma estrutura de duas redes em paralelo: uma rede principal (target stream) e uma rede auxiliar (cross-target stream). Ambas são configuradas com um classificador softmax e compartilham os mesmos parâmetros, o que permite aprender características comuns em diferentes conjuntos de dados de rostos.

Durante o treinamento, a MIND-Net otimiza suas previsões usando uma combinação de perdas, incluindo uma perda auxiliar (triplet loss) que ajuda a diferenciar melhor as identidades. Esse modelo também calcula a informação mútua entre as redes para capturar traços faciais redundantes que são úteis para melhorar a precisão do reconhecimento de rostos. A triplet loss foi um fator muito importante para o desenvolvimento do código, e por isso, deve ser destacada. No capítulo 4, haverá uma explicação detalhada sobre como ela funciona.

3.2.1.5 Reconhecimento facial em vídeo – SiamSRC

O modelo proposto por Mokhayeri e Granger [47] tem como objetivo o reconhecimento de rostos aplicado em um vídeo em comparação com uma galeria de imagens estáticas. Para isso, ele utilizou uma rede siamesa, uma vez que são modelos excelentes para comparação entre faces [64], [65], [66], [67]. Eles organizaram sua base de dados em grupos separados por posição do rosto e iluminação, e a rede associava a face detectada no vídeo com algum desses grupos para reduzir a quantidade de comparações. Essa estratégia se mostrou excelente e eficaz, por

se tratar de um modelo de comparação e não de classificação. O grande problema é que o poder computacional necessário é bem maior em relação às outras abordagens devido ao uso de duas CNNs (rede siamesa) e a aplicação delas em vídeo. No entanto, a capacidade comparativa da rede apresentada chamou muita atenção, se mostrando ser muito mais precisa que redes comuns.

3.2.2 Ajustes Metodológicos

Durante o desenvolvimento do software, é comum que surjam novas ideias, métodos e ferramentas. Como já foi citado, os recursos da internet foram, em sua maioria, introduzidos em momentos diferentes da implementação, conforme a necessidade aparecia. Isso se deve ao fato de um grande dinamismo de ideias de uma pesquisa com uma duração mais estendida, como uma tese de Mestrado. Essa aparição de novos pensamentos continuaram sendo frequentes em relação à aplicação prática do código, quando foram identificados novos problemas e a solução era necessária.

Primeiramente, um obstáculo foi a preparação dos dados para treinamento. Dado que o *dataset Labeled Faces in the Wild* (LFW) é o *dataset* mais utilizado em tarefas de reconhecimento facial, por sua grande quantidade de registros e sua versatilidade de uso, ele foi utilizado em boa parte do desenvolvimento da rede. A variação de conjuntos de dados tem sua importância, mas perde mais sentido nesse caso, pois o trabalho não envolve comparação entre *datasets*, e é justo recorrer àquele mais requisitado. No entanto, o problema foi que as redes observavam a imagem inteira e acabavam por captar elementos que fogem aos limites faciais das pessoas. Logo, foi aplicado um modelo ResNet pré-treinado para cortar as imagens do *dataset*. Mas, em seguida, um novo conjunto de dados [68] formado também a partir do LFW foi analisado e aplicado. Essa substituição ocorreu não só porque as faces estavam todas cortadas corretamente, mas também por haver um leve *data augmentation* e uma melhor organização das imagens para a tarefa específica de treinamento de comparação.

Além disso, também foram adicionados pequenos métodos de melhoria do modelo em código que serão mais aprofundados no próximo capítulo. O modelo por muito tempo teve problemas de *overfitting*, que ocorre quando a rede se adequa demais para os casos do treinamento, mas não consegue fazer previsões na prática. Para ajustar esse problema, foi aplicada uma técnica parecida com a que foi utilizada no modelo LightQNet, seleção de *hard samples*, que na prática utiliza a previsão do modelo para selecionar as entradas mais difíceis de acertar e treinar elas mais uma vez. Dessa forma, o modelo se adapta a observar melhor as características singulares de cada rosto, aumentando a eficiência do modelo. A outra técnica usada foi a aplicação de um maior *data augmentation* para aumentar a quantidade de amostras no *dataset*.

4 Solução Proposta

De acordo com o que foi apurado nos capítulos anteriores, para a resolução deste problema, será usado uma rede siamesa composta por duas redes neurais convolucionais (CNN) combinadas com um método de *triplet loss*, para que seja potencializada a capacidade de comparação do software. Portanto, neste capítulo, será abordado todo o processo de desenvolvimento do software, incluindo implementação, descrição dos conceitos e aplicação prática da *siamese network* e *triplet loss*, as estratégias de processamento de dados como *data augmentation* e seleção de *hard* e *semi-hard triplets*, assim como a exibição dos testes e resultados.

4.1 Siamese Network

As redes neurais siamesas foram introduzidas em 1993 pelos pesquisadores Bromley et al. [69], acompanhados de outros colaboradores no laboratório de pesquisas da AT&T Bell Labs [70]. O conceito surgiu para resolver problemas específicos de verificação de assinatura manuscrita, com a tarefa de identificar se duas assinaturas provinham da mesma pessoa. Com o objetivo de desenvolver um software que pudesse distinguir assinaturas genuínas de falsificações, essa nova arquitetura foi idealizada para comparar diretamente dois padrões e analisar sua similaridade. A inovação foi um passo significativo, pois permitiu que o sistema aprendesse um espaço de características que refletia a autenticidade da assinatura com base nas características do traçado, pressão e outros detalhes perceptíveis na escrita.

O funcionamento padrão dessa arquitetura é relativamente simples: ela opera com duas entradas que são passadas simultaneamente por duas redes neurais idênticas, compartilhando pesos e parâmetros. Essas duas redes, que compõem a arquitetura "siamesa", processam individualmente cada entrada, extraindo as principais características do padrão em questão e gerando uma representação vetorial única para cada um. Após a extração, essas representações

vetoriais são comparadas, e um limiar (ou threshold) pré-definido é utilizado para determinar se ambas as entradas possuem um nível de similaridade aceitável para serem consideradas iguais ou, no caso da assinatura, autênticas. Uma representação ilustrativa dessa arquitetura pode ser observada na Figura 11.

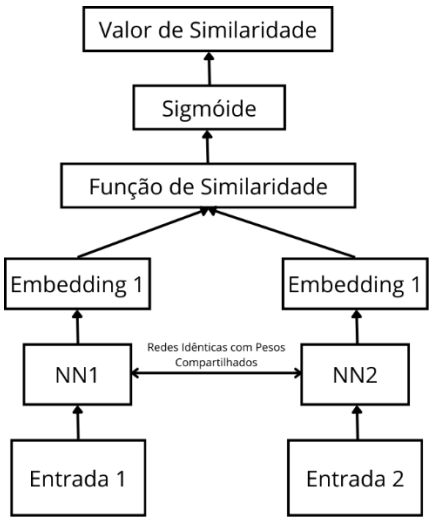

Figura 11: Arquitetura Clássica de uma rede neural siamesa.

Desde essa época, o uso das redes siamesas se expandiu muito além do escopo original da verificação de assinaturas. Hoje, elas têm aplicações diversificadas em software de detecção e reconhecimento de padrões, incluindo o reconhecimento facial [47], [71], [72], [73], de objetos [74], [75], [76], de voz [77], [78], [79] e até mesmo em sistemas de emparelhamento de documentos [80]. Essa capacidade de aprender um "espaço de similaridade" entre padrões distintos, em vez de apenas classificar as entradas em categorias fixas, é o que tornou as redes siamesas amplamente adotadas e tão bem avaliadas em estudos subsequentes. Essa abordagem possibilita, por exemplo, que a rede identifique com precisão se duas imagens de rostos são da mesma pessoa.

As redes neurais siamesas apresentam grandes vantagens que explicam o motivo de terem sido tão utilizadas em estudos recente de reconhecimento de padrões e problemas de similaridade. Uma dessas vantagens é a sua capacidade de comparar as entradas de maneira direta, sem a necessidade de utilizar uma base de dados muito grande, permitindo, assim, sua aplicação em treinamentos com pouca amostra (*few-shot learning*). Essa característica é muito interessante para casos em que os recursos são custosos ou difíceis de coletar. Além disso, o próprio fato de operarem com duas entradas em redes idênticas torna o modelo mais eficiente, uma vez que não seria necessário uma classificação e sim uma avaliação de similaridade entre as duas entradas, simplificando o problema para casos em que as classes fixas não são necessárias, como é o caso desse trabalho.

Ademais, essa capacidade de aprendizagem de um “espaço de similaridade” é um ponto forte dessa arquitetura. Sem precisar classificar cada entrada em uma categoria predefinida, a rede foca em criar um espaço vetorial onde os valores representam o grau de semelhança entre elas.

Esse recurso é essencial na tarefa de identificação e reconhecimento facial, uma vez que os fatores externos, como iluminação, expressão e ângulo, podem afetar diretamente as características faciais da pessoa. Essa generalização possibilitada pela rede a torna ideal para ambiente menos controlados e mais dinâmicos.

Entretanto, sempre há obstáculos e limitações. Entre os principais desafios das redes siamesas é a escalabilidade. Esses modelos trabalham comparando entradas, o que significa que quanto mais entradas forem analisadas, mais processamento será necessário para concluir o reconhecimento. Na tarefa de reconhecimento facial estudada neste trabalho, por exemplo, cada fotografia capturada pelo sistema seria comparada com a base de dados dos estudantes cadastrados. Conforme essa base aumenta, a capacidade de processamento necessária para a conclusão do reconhecimento aumenta drasticamente. No entanto, pode-se afirmar que a quantidade de estudantes a serem cadastrados no sistema, apesar de poder variar durante o período letivo, não deve crescer a ponto de prejudicar o desempenho do software.

Um outro problema a ser observado é a necessidade de definição de um limiar (*threshold*), o limite de similaridade. O ponto ideal de definição da similaridade suficiente é um fator que demanda muita atenção, pois pessoas muito parecidas não devem ser determinadas como a mesma pessoa, assim como uma alteração na face do indivíduo não deve ser o suficiente para que o sistema a rejeite. Esse valor a ser definido, conhecido também como margem (ou *margin*) deve ser bem estabelecido para fugir de falsos positivos. No entanto, esse é um desafio relativamente simples de se resolver em comparação com algum problema na ideia do modelo. Uma atenção é necessária, mas nada que custe processamento ou recursos.

4.1.1.1 Funções de Perda

Adicionalmente, o desenvolvimento de novas variantes das redes siamesas permitiu avanços significativos. Redes siamesas combinadas com funções de perda específicas, como a *contrastive loss* ou a *triplet loss*, ajudam a melhorar a precisão ao forçar a rede a aprender representações de maneira que imagens similares estejam próximas no espaço vetorial, enquanto imagens de classes diferentes estão afastadas.

A *contrastive loss* foi uma das primeiras perdas desenvolvidas para redes siamesas e possui um funcionamento relativamente mais simples. Ela incentiva que a rede aproxime pares semelhantes e afaste pares que não são semelhantes, e a sua fórmula pode ser representada como exibido na Equação 1:

$$L = (1 - Y)\frac{1}{2}D^2 + (Y)\frac{1}{2}max(0, m - D)^2$$

Equação 1: Contrastive Loss

onde D representa a distância entre os *embeddings* dos pares, e Y é um valor binário que indica se as entradas são parecidas (0) ou diferentes (1). A margem m funciona para que a rede mantenha uma distância mínima entre pares não semelhantes. Esse método binário ajuda o modelo a ter uma distinção simples entre classes. Conforme estudos iniciais, como o de Hadsell

et al. [81], a *constrastive loss* teve bons resultados em reconhecimento de padrões, mas apresenta certas limitações quando se trata de variabilidade interna, como poses e iluminação variadas, que são comuns em rostos.

Por outro lado, a *triplet loss* é uma função de perda mais avançada que foi criada justamente para lidar melhor com essa variabilidade. Ela utiliza três amostras em vez de duas: uma âncora, uma positiva (da mesma classe) e uma negativa (de outra classe). O objetivo é que a distância entre a âncora e a positiva seja menor do que a distância entre a âncora e a negativa, formando um espaço de similaridade mais organizado. A formulação dessa perda pode ser observada na Equação 2:

$$L = max(0, \|f(a) - f(p)\|^2 - \|f(a) - f(n)\|^2 + \propto)$$

Equação 2: Triplet Loss

onde $f(a)$, $f(p)$ e $f(n)$ representam os *embeddings* da âncora, positiva e negativa, e $\propto$ é uma margem que ajuda a rede a estabelecer uma separação clara. Essa margem serve para que a rede consiga estruturar melhor as amostras parecidas e diferentes.

No caso de reconhecimento facial, a *triplet loss* é mais eficaz em lidar com variações dentro das classes, como expressões e ângulos diferentes. Um bom exemplo de aplicação dessa perda é o sistema *FaceNet* [82], onde o uso da *triplet loss* ajudou a rede a criar uma organização coerente no espaço de *embeddings* para diferenciar melhor as classes, mesmo com muita variação.

Apesar disso, a *triplet loss* apresenta maior complexidade, pois é necessário escolher os tripletos de forma adequada (usando métodos como *hard triplet mining*, que será abordado em breve). Esse processo demanda mais atenção no treinamento, mas oferece uma precisão mais alta, sendo ideal para redes siamesas em aplicações que requerem maior discriminação entre amostras. Por outro lado, a *contrastive loss* é mais direta e funciona bem em problemas menos complexos, onde o objetivo é apenas verificar a similaridade entre pares sem grande variação interna.

Em resumo, a *contrastive loss* é indicada para tarefas de verificação com baixa complexidade, enquanto a *triplet loss* se destaca em cenários que exigem um controle mais detalhado das variações intraclasse, como o reconhecimento facial.

A *triplet loss* foi escolhida, no entanto, por ter havido muitos problemas em relação a essas variações de intraclasse. O modelo final, como ainda será abordado, ainda luta com problemas do tipo, mas foi drasticamente minimizado com a aplicação da *triplet*. Essa função de perda foi aplicada inicialmente, mas se mostrou mais eficaz no decorrer da implementação.

4.1.1.2 Modelos Utilizados

A tarefa de desenvolvimento de IA é complexa por si só. Com isso em mente, utilizaram-se de arquiteturas pré-treinadas na arquitetura da rede siamesa. O objetivo foi adicionar camadas no modelo para aplicá-lo em um exemplo específico como este. Para escolher a melhor rede base

para ser utilizada no *software* final, foram feitos testes com três modelos: FaceNet, ResNet-50 e MobileNet.

As três redes possuem objetivos distintos. Enquanto a primeira está mais relacionada com a otimização dos resultados, a ResNet-50 propõe um modelo mais robusto, porém mais pesado, e a MobileNet apresenta um modelo leve e menos preciso. Os testes feitos para esses modelos serão apresentados em um tópico futuro.

4.1.1.3 Arquitetura

Utilizando o modelo base da ResNet-50 e da MobileNet para implementar a rede, foram feitas algumas adições de camadas a fim de afunilar a função delas para o reconhecimento facial. Essa técnica é conhecida como *transfer learning*, quando um modelo pronto é aplicado como ponto de partida para a resolução de um outro problema.

```
inputs = keras.Input(imageSize + (3,))
x = resnet.preprocess_input(inputs)
baseCnn = resnet.ResNet50(weights="imagenet", include_top=False)
baseCnn.trainable = False
extractedFeatures = baseCnn(x)
x = layers.GlobalAveragePooling2D()(extractedFeatures)
x = layers.Dense(units=1024, activation="relu")(x)
x = layers.Dropout(0.5)(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(units=512, activation="relu")(x)
x = layers.Dropout(0.5)(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(units=256, activation="relu")(x)
x = layers.Dropout(0.5)(x)
outputs = tf.nn.l2_normalize(layers.Dense(units=128)(x), axis=-1)
```

Snippet de Código 1: Arquitetura da CNN

Como pode ser observado no Snippet de Código 1, foram adicionadas camadas extras como *Global Average Pooling 2D, Dense, Dropout e Batch Normalization*. As camadas de *Pooling 2D* servem para a redução de dimensionalidade dos features e retorna um vetor de características. As camadas de *Dense* são totalmente conectadas (*fully-connected layers*) aplicando uma função de ativação ReLU para evitar a linearidade do modelo. Percebe-se que a quantidade de unidades diminui com o decorrer do código, o que significa que os nós da arquitetura convergem para um saída. A camada de *Dropout* é responsável pela desativação de 50% dos nós, de forma que o *overfitting* possa ser evitado. E o *Batch Normalization* normaliza as ativações, acelerando o treinamento e melhorando a estabilidade. No fim da definição da CNN, pode-se notar uma camada de normalização L2, que define todos os vetores como unitários, o que elimina a preocupação com a magnitude, focando apenas na distância angular entre os *embeddings*. E por ser um espaço vetorial de 128 dimensões, a quantidade de direções e ângulos possíveis é impressionante.

O modelo FaceNet, por sua vez, poderia ter sido implementado da mesma forma, mas por problemas de compatibilidade, teve que ser descrito e treinado do zero. Dessa forma, há textos sobre a estrutura do FaceNet que podem ser consultados [82], [83], [84].

A Figura 12 representa a arquitetura completa da rede proposta, demonstrando o fluxo de processamento das entradas até terminar na saída. Temos que as entradas são representadas como as imagens em cada coluna, que serão inseridas diretamente na CNN.

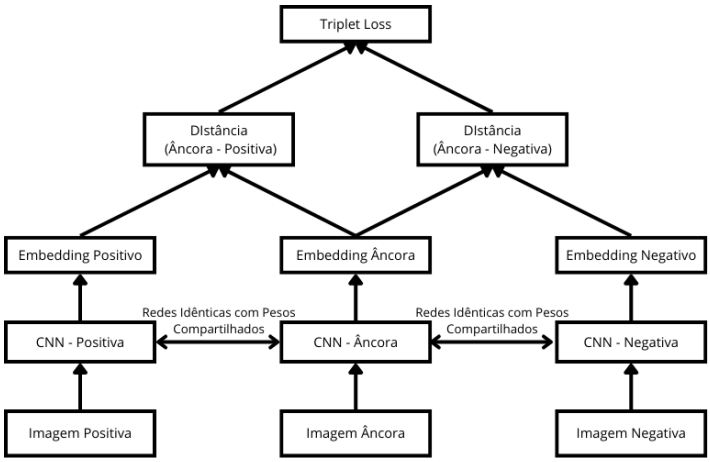

Figura 12: Arquitetura Completa da Rede Siamesa com Triplet Loss

Além disso, está sendo reiterado, na imagem, a similaridade entre as CNNs, que extraem ao mesmo tempo os *embeddings* de cada imagem. Em seguida, é feito um cálculo da distância euclidiana entre os *embeddings*, de forma que sua similaridade seja detectada. E por fim, as distâncias passam por uma *triplet loss* que verificará se o modelo foi bem sucedido. Para isso, a distância âncora-positiva deve ser maior que a distância âncora-negativa. Pode-se também, em alguns casos, considerar uma margem mínima que limitará a diferença necessária entre as duas distâncias que figurará em um acerto da rede.

4.2 Processamento de Dados

Como foi discutido nos capítulos anteriores, uma versão do conjunto de dados *Labeled Faces in the Wild* foi utilizada no processo de treinamento do modelo deste projeto. No entanto, para que ele se ajustasse melhor aos objetivos específicos do estudo, foram feitas algumas adaptações e modificações no *dataset*.

Essas alterações foram implementadas com o propósito de melhorar a performance do modelo e adequar o conjunto de dados às necessidades particulares do projeto. Dessa forma, em vez de usar o *dataset* em sua forma original, ele passou por ajustes que ajudaram a tornar o treinamento mais alinhado com as exigências do modelo, considerando aspectos específicos que o projeto requeria.

Essas mudanças foram fundamentais para garantir que o modelo conseguisse alcançar um bom desempenho, lidando melhor com os dados disponíveis e trazendo resultados mais consistentes e relevantes.

4.2.1 Geração de Triplets

No início do desenvolvimento, foi preciso gerar tripletos de imagens, ou seja, conjuntos com três fotos: uma âncora, uma positiva e uma negativa. A âncora e a imagem positiva pertenciam à mesma pessoa, enquanto a negativa era de uma pessoa diferente. Para isso, foi necessário excluir pessoas que tinham apenas uma imagem no *dataset*, já que eram precisas ao menos duas fotos de cada pessoa para que o processo funcionasse corretamente.

Durante os testes, porém, o modelo começou a apresentar resultados estranhos. Ao comparar uma foto capturada pela câmera com as imagens de uma pequena base de dados, o modelo retornava valores muito próximos entre si, dificultando a identificação correta. Assim, foram feitas algumas mudanças na forma de gerenciamento das imagens para tentar melhorar a precisão dos resultados.

A primeira modificação foi a aplicação de uma técnica chamada *hard triplet mining*. Essa abordagem envolve uma seleção mais criteriosa dos tripletos, priorizando pessoas com rostos mais semelhantes. Para implementar essa técnica, o modelo foi inicialmente treinado com o conjunto de dados e, em seguida, testado com os tripletos gerados anteriormente para medir as distâncias entre eles. Depois disso, foram selecionados os tripletos que não atenderam a uma margem de distância pré-definida, ou seja, que não passaram no teste. Trabalhos como os de [45], [85], [86], [87] aplicam essa mesma estratégia, já que ela ajuda a evitar o overfitting do modelo, tornando-o mais generalizável para novos dados.

No entanto, apesar da melhora nos resultados, o modelo continuou a manter os *embeddings* próximos demais, o que exigiu uma implementação do semi-hard triplet mining, que tem uma função parecida com o seu antecessor. A diferença nesse caso é que os tripletos que beiram a margem também são selecionados e priorizados. Isso evita ainda mais o overfitting, pois no método anterior, o modelo poderia acabar se adaptando a detalhes muito específicos de cada amostra, podendo acabar causando um efeito inverso. Nessa nova abordagem, há uma curva de aprendizagem para o modelo, em vez de apenas aumentar drasticamente e subitamente a dificuldade.

A terceira modificação, e a mais simples, foi o *data augmentation*, ou aumento de dados. Esta é uma técnica muito utilizada em *machine learning* para aumentar a variedade e quantidade de dados disponíveis sem a necessidade de novas coletas. No reconhecimento facial, por exemplo, os treinamentos utilizam uma quantidade exorbitante de imagens em busca de uma eficiência maior, e nesses casos, o *data augmentation* se torna especialmente útil. O objetivo dessa abordagem é gerar novas amostras a partir das que já existem, aplicando várias transformações como rotações, espelhamento, ajustes de brilho, variações de contraste, entre outras possíveis

alterações nas imagens originais. Essas modificações permitem criar um conjunto maior de variações para que o modelo se torne mais robusto e adaptável a diferentes situações.

Essa estratégia ganha mais força quando o treinamento tem acesso a poucos dados, o que é comum em aplicações muito específicas. Em muitos casos, os *datasets* disponíveis são pequenos, e aumentar as quantidade de amostras ajuda a evitar o *overfitting*. Com o aumento de dados, como o modelo teria acesso a uma maior variação de amostras, ele não fica preso em características muito individuais, podendo se adaptar errôneamente ao caso.

Para se ter uma ideia, um exemplo prático no aumento de dados é a rotação das imagens em diferentes graus. No caso de reconhecimento facial, uma pequena rotação simula variações naturais na postura ou no enquadramento das fotos, o que prepara o modelo para identificar rostos que aparecem em ângulos variados. Além disso, o espelhamento das imagens permite que o modelo aprenda a reconhecer o rosto de uma pessoa tanto do lado esquerdo quanto do direito, o que aumenta sua capacidade de adaptação.

Outras técnicas de *data augmentation* incluem o ajuste de brilho e contraste, que é especialmente útil para criar variações na iluminação. Dessa maneira, o modelo se torna mais robusto contra diferenças de luz entre as imagens, uma vez que passa a aprender a diferenciar o que é relevante para o reconhecimento facial mesmo sob condições de iluminação variáveis. A adição de pequenos ruídos também pode ser utilizada, o que prepara o modelo para lidar com imagens de qualidade inferior, algo que pode ocorrer em fotos capturadas em ambientes de baixa resolução ou com distorções.

Dessa forma, o modelo se mostrou muito mais eficiente, conseguindo, portanto, demonstrar diferenças maiores entre os *embeddings* negativos e positivos, aumentando a eficiência e a precisão.

4.3 Implementação

Este item aborda especificamente da estruturação do código e algumas decisões de programação. Todo o conteúdo compilado na pesquisa convergiu para a produção desses arquivos que serão descritos a seguir.

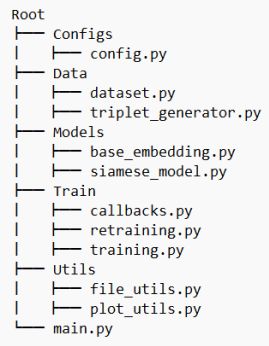

Figura 13: Estrutura do Código

A implementação do código foi dividida como pode ser observado na Figura 13. Um módulo para as configurações, para o *dataset*, os modelos, para o treino e ferramentas. Uma breve explicação de cada módulo é descrita a seguir:

```
IMG_DIMS = (224, 224)
BATCH_SZ = 64
AUTO_TUNE = tf.data.AUTOTUNE
INIT_LEARNING_RATE = 0.0001
TRAIN_EPOCH_STEPS = 50
VALID_EPOCH_STEPS = 10
NUM_EPOCHS = 50
```

Snippet de Código 2: Hiperparâmetros do Treinamento

O módulo de configuração contém apenas um arquivo, *config.py*. Ele é responsável pela definição de variáveis globais como hiperparâmetros e caminhos dos arquivos de entrada e saída. Dentre os hiperparâmetros mostrados no Snippet de Código 2, é importante destacar a variável global "INIT_LEARNING_RATE", que define a taxa de aprendizado inicial. Isso se deve ao fato de que foi implementada uma função de alteração dessa taxa caso o modelo comece a divergir. Também pode ser observado o valor do *batch size*, que corresponde a quantidade de amostras por etapa a serem processadas. Por ter poucos recursos disponíveis, esse foi o limite que o modelo pode atingir em relação ao tamanho do *batch*. Por fim, a dimensão das imagens "IMG_DIMS" é diferente para o modelo FaceNet que trabalha com dimensões (160, 160).

```
class TripletDataGenerator:
    def __init__(self, data_path, max_triplets=1000):
        self.image_folders = [os.path.join(data_path, folder) for folder in
os.listdir(data_path) if len(os.listdir(os.path.join(data_path, folder))) >
1]
        self.all_images_dict = self._create_images_dict()
        self.triplet_limit = max_triplets

    def _create_images_dict(self):
        images_dict = {}
        for person_folder in self.image_folders:
            images = os.listdir(person_folder)
            images_dict[person_folder] = [os.path.join(person_folder, img)
for img in images]
        return images_dict

    def generate_triplets(self):
        triplet_count = 0
        while triplet_count < self.triplet_limit:
            anchor_person = random.choice(self.image_folders)
            other_people = self.image_folders.copy()
            other_people.remove(anchor_person)
            negative_person = random.choice(other_people)
```

```
            anchor_img, positive_img =
np.random.choice(self.all_images_dict[anchor_person], size=2,
replace=False)
            negative_img =
random.choice(self.all_images_dict[negative_person])

            triplet_count += 1
            yield anchor_img, positive_img, negative_img
```

Snippet de Código 3: Classe TripletGenerator

A pasta "*data*" é composta por dois arquivos, sendo que o *dataset.py* contém as funções que fazem o processamento das imagens carregadas do *dataset* e o *triplet_generator.py* responsável pela classe que gera as triplets para o treinamento. Observando o Snippet de Código 3, pode-se notar a forma como a seleção de tripletos foi feita, selecionando aleatoriamente uma pessoa na base de dados para as imagens de âncora e positiva e uma outra pessoa aleatória para a imagem negativa.

```
def build_siamese_network(input_size, embedding_network):
    anchor_input = keras.Input(name="anchor_img", shape=input_size + (3,))
    positive_input = keras.Input(name="positive_img", shape=input_size + (3,))
    negative_input = keras.Input(name="negative_img", shape=input_size + (3,))
    anchor_embedding = embedding_network(anchor_input)
    positive_embedding = embedding_network(positive_input)
    negative_embedding = embedding_network(negative_input)

    return keras.Model(
        inputs=[anchor_input, positive_input, negative_input],
        outputs=[anchor_embedding, positive_embedding, negative_embedding]
    )
class SiameseTripletModel(keras.Model):
    def __init__(self, siamese_network, margin, loss_tracker):
        super().__init__()
        self.siamese_network = siamese_network
        self.margin = margin
        self.loss_tracker = loss_tracker

    def _calculate_distances(self, images):
        anchor, positive, negative = images
        embeddings = self.siamese_network((anchor, positive, negative))
        anchor_embed, positive_embed, negative_embed = embeddings
              anchor_pos_dist   =   tf.reduce_sum(tf.square(anchor_embed   -
positive_embed), axis=-1)
              anchor_neg_dist   =   tf.reduce_sum(tf.square(anchor_embed   -
negative_embed), axis=-1)
        return anchor_pos_dist, anchor_neg_dist
    def _compute_triplet_loss(self, anchor_pos_dist, anchor_neg_dist):
        loss_value = anchor_pos_dist - anchor_neg_dist
        return tf.maximum(loss_value + self.margin, 0.0)
```

Snippet de Código 4: Definição da Rede Siamesa

O módulo "*models*" é composto pelos arquivos *base_embeddings.py* e *siamese_model.py*. O primeiro teve sua parte mais importante explicitada no Snippet de Código 1, representando as configurações da arquitetura da rede convolucional que compõe o sistema siamês. Já o outro arquivo apresenta todas as funções que definem a rede como siamesa, como a construção do modelo com a definição das triplets. No Snippet de Código 4, é apresentada a função que define as entradas, assim como o cálculo da *triplet loss*, que condiz com a fórmula apresentada anteriormente neste capítulo. Esse código representa parte do arquivo *siamese_model.py*.

Os arquivos responsáveis pelo treinamento do código são *training.py*, *retraining.py* e *callbacks.py* e eles compõem a pasta "*train*". O primeiro, como o próprio nome já diz, executa a rotina de treinamento da rede. O segundo, por sua vez, é responsável pela seleção das *semi-hard triplets* e as utiliza para um novo treinamento com o objetivo de reduzir o *overfitting*. Os *callbacks*, ou chamadas de retorno, são definidos no último arquivo e servem para dar um *feedback* para o desenvolvedor sobre o que está ocorrendo na execução, devido à quantidade de tempo elevada para executar a tarefa.

Por fim, o módulo "*utils*" contém as ferramentas para a execução do código, como carregamento de arquivos – *file_utils.py* – e plotagem das métricas e gráficos – *plot_utils.py*. O arquivo *main.py*, que está fora dos módulos representa a rotina completa do código, desde o carregamento dos arquivos até a plotagem dos resultados. É importante explicitar, também, que a rotina de testes do software não está incluída nesta organização, uma vez que ela faz parte de uma outra tarefa.

4.4 Testes e Resultados

A partir das definições apresentadas anteriormente, foram feitos alguns testes para conclusão das melhores alternativas de abordagens para o modelo final. Esses testes são importantes para a verificação de possíveis erros conceituais e práticos, e sua função principal é analisar as capacidades do objeto de teste – no caso, o modelo de reconhecimento facial.

4.4.1 Métodos e Métricas

Tão importante como a execução dos testes, os métodos e métricas devem ser sempre planejados e fundamentados, pois eles devem apresentar certa credibilidade para serem válidos. Para este projeto, foram feitos dois testes de desempenho voltados para as capacidades do algoritmo.

O primeiro deles foi um teste empírico, que consistiu em comparar uma imagem capturada pela câmera com uma base de dados com poucas pessoas. O objetivo deste teste era verificar se o modelo era capaz de identificar a pessoa a ter sua foto capturada dentre àquelas que estavam cadastradas na base de dados. Para isso, foi feita uma pequena interface que simula o que

poderia estar presente no sistema final. Ela captura a imagem e expõe os dados da pessoa na tela. A comparação é feita da seguinte forma: a fotografia recente se torna a âncora e passa por um bloco de repetição que itera sobre todas as imagens da base de dados, que correspondem às imagens a serem comparadas com a âncora. Para completar, foi adicionado uma imagem qualquer de algo não-humano apenas para preencher. Esse esquema pode ser alterado depois para uma forma mais otimizada de comparação. Em seguida, o modelo é aplicado nesse tripleto e retorna a distância entre os *embeddings* das faces. Após todas a iterações, a pessoa que tivesse o menor valor de distância teria seu nome exibido na interface. Essa abordagem é a ideia inicial de aplicação final do modelo.

No entanto, não existe embasamento que credibilize este método, portanto, o segundo teste foi feito. Ele consiste em arranjar os tripletos a partir de uma base de dados maior que não foi utilizada no treinamento da rede. Dessa forma, o modelo é aplicado em cada tripleto, contabilizando cada acerto e cada erro da rede. Em cada iteração, o modelo vai ter duas verificações, uma para a validação da imagem positiva e a outra para a negativa. O modelo acerta toda vez que a distância âncora-positiva é menor que a margem (caracterizando a indentificação da mesma pessoa) e que a distância âncora-negativa é maior que a margem. O contrário de ambas as situações caracterizam um erro do modelo. Esta forma de avaliar é baseada na matriz de confusão (*confusion matrix*), que consiste em separar os palpites da rede em positivos e negativos – que representa a identificação ou não da pessoa na comparação – e da mesma forma com as saídas esperadas. Isso gera quatro grupos de possíveis palpites: *true positives* (TP, ou positivos verdadeiros) e *true negatives* (TN, ou negativos verdadeiros), que ocorrem quando o modelo classifica a amostra como positivo ou negativo, respectivamente, e corresponde com o que era esperado; e os *false positives* (FP, ou positivos falsos) e *false negatives* (FN, ou negativos falsos), que ocorrem quando o modelo erra o previsto, sendo positivo ou negativo, respectivamente. Essa associação pode ser melhor observada na Figura 14, onde a saída prevista representa o que o modelo definiu e a saída esperada é a resposta real.

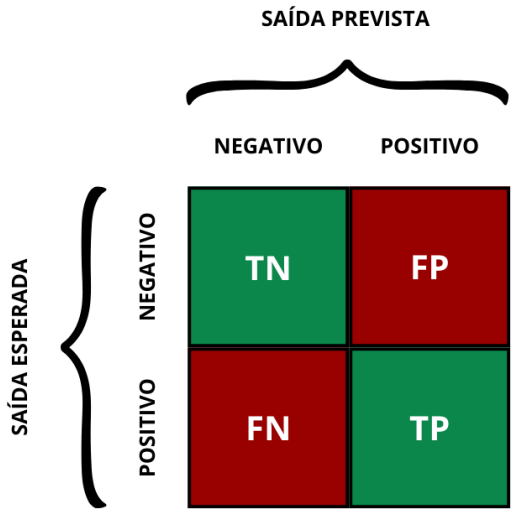

Figura 14: Matriz de confusão

Após a definição dessas variáveis, são calculadas as métricas utilizadas para avaliação do modelo: precisão, acurácia, revocação e *F1-score*. A primeira representa a porcentagem de vezes em que o modelo acertou ao definir uma amostra como correta, ou seja, é a razão entre a quantidade de positivos reais (TP) e a quantidade de amostras classificadas como positivas (TP + FP). A acurácia, por sua vez, representa a taxa de acerto total do modelo, ou seja, é a razão entre os palpites verdadeiros (TN + TP) e todas as amostras (TN + TP + FN + FP). A revocação, ou *recall*, representa o oposto da precisão, definindo a taxa de acerto de amostras previstas como negativas, ou seja, a razão entre a quantidade de negativos reais (TN) e a quantidade de amostras classificadas como negativas (TN + FN). Por fim, o *F1-score* significa a harmonia entre a precisão e a revocação, tendo seu limite atingido, quando ambas métricas têm resultado perfeito: 100%. Essa métrica é utilizada, pois em um bom classificador, as duas métricas devem ser ótimas, e ele resume a qualidade dessa harmonia em um valor só, seu cálculo pode ser observado na Equação 3:

$$\text{F1 Score} = 2 * \frac{\text{precisão} * \text{revocação}}{\text{precisão} + \text{revocação}}$$

Equação 3: F1-score

Esse método foi aplicado para todas as redes base e com todos os *dataset*s disponíveis.

4.4.2 Resultados dos testes

Iniciando com o segundo teste, temos que os objetos de teste foram os três modelos base destacados: ResNet50, MobileNet e Facenet; e os três *dataset*s utilizados: LFW, Extended Yale B e CelebA.

É importeante destacar que o CelebA e o Yale foram usado apenas para testes, uma vez que as bases de imagens são muito pequenas para um treinamento de redes neurais. Além disso, o *dataset* LFW usado para os testes foi separado do conjunto de treino antes do aprendizado começar. Essas alternativas de *dataset* para os testes foram escolhidas para testar o comportamento diante das diversas situações que o modelo pode se encontrar. O Extended Yale B fornece um conjunto de imagens frontais com leves alterações de expressões e muita alteração de iluminação. No entanto, muitas imagens extremamente escuras foram removidas para que elas não comprometam completamente os resultados com esse *dataset*. O conjunto CelebA, por sua vez, é composto por várias imagens de cada indivíduo com posições diferentes do rosto.

Por fim, foram feitos testes com duas margens diferentes: 0.5 e 0.75. Todas essas informações podem ser observadas na Tabela 3:

Tabela 3: Valores das métricas dos testes com margens 0.5 e 0.75, respectivamente.

Modelo / Dataset	Precisão	Revocação	Acurácia	F1-score
ResNet / LFW	66.43%/62.13%	20.92%/37.52%	55.17%/57.32%	31.81%/46.79%
ResNet / CelebA	63.65%/54.69%	73.58%/85.20%	65.78%/57.31%	68.26%/66.62%
ResNet / Yale	72.02%/63.73%	65.71%/81.46%	70.09%/67.55%	68.72%/71.51%
MobileNet / LFW	68.15%/63.80%	64.75%/80.03%	67.24%/67.31%	66.40%/71.00%
MobileNet / CelebA	67.99%/63.63%	51.98%/64.73%	63.75%/63.87%	58.92%/64.17%
MobileNet / Yale	56.88%/53.45%	84.42%/91.81%	60.21%/55.93%	67.97%/67.57%
FaceNet / LFW	99.36%/97.59%	8.31%/31.70%	54.13%/65.46%	15.33%/47.85%
FaceNet / CelebA	70.84%/53.99%	89.03%/91.48%	76.19%/56.76%	78.90%/67.91%
FaceNet / Yale	79.00%/59.86%	66.67%/87.87%	74.47%/64.48%	72.31%/71.21%

Antes de partir para uma análise mais profunda da tabela acima, deve-se ressaltar que os valores podem ser melhorados ao aplicar margens diferentes, específicas para cada modelo. Os exemplos escolhidos refletem a variação de desempenho de cada modelo conforme a permissividade também varia. Por exemplo, é notório que ao aumentar o valor da margem, a precisão cai em todos os testes. Isso acontece porque o modelo começa a categorizar como positivo mais facilmente, aumentando o número de falsos positivos. Um comportamento contrário ocorre com a revocação, que cresce conforme a margem aumenta. De forma análoga, ao aumentar a permissividade do modelo, a frequência de classificações negativas diminui, reduzindo os erros nessa categoria. A acurácia, por sua vez, mostra a capacidade geral do *software* de classificar corretamente, variando bastante entre os modelos. Por ter um contraste muito grande entre a precisão e a revocação, essa métrica não tem um valor elevado, atingindo apenas os 76% no melhor dos casos. O mesmo ocorre para o *F1-score,* que corresponde a harmonia entre as duas métricas, denotando de forma mais clara esse contraste.

Outra observação importante a ser feita é em relação aos *datasets* adotados, que possuem características únicas determinantes para a geração dos valores. O conjunto CelebA é composto por imagens com variações horizontais da posição do rosto da pessoa, o que não ocorre na aplicação prática desse projeto, uma vez que as imagens a ser utilizadas deverão ter posição, expressão e iluminação normalizadas. Por esse motivo, a baixa precisão não preocupa, mesmo que os valores estejam sendo utilizados para comparação. O mesmo ocorre para o conjunto Yale, que seria muito interessante para o treinamento, pois é composto por fotografias frontais de cada indivíduo com variação de iluminação e expressão, e assim como o anterior não deve preocupar por não ser necessária uma grande precisão relacionada a esses fatores na aplicação prática do *software*. O *dataset* LFW, por sua vez, apresenta imagens com bastantes variações, mas com a vantagem de ter esse mesmo padrão utilizado no treinamento das redes. Valores altos eram esperados nesses testes, mas não aconteceu com todos os modelos.

Finalmente, em relação aos modelos, pode-se observar uma grande variedade de porcentagens entre eles. O modelo mais robusto, ResNet, apresentou valores bem constantes entre os *datasets*. Na precisão, a variação não ultrapassou 10% em ambas as margens, evidenciando a estabilidade do modelo ao fazer classificações positivas. Em casos de variação específica de

iluminação, posição e expressão, o modelo consegue determinar de forma acertiva e constante os exemplos negativos, como pode ser visto nos resultados de revocação dos *datasets* Yale e CelebA, principalmente para a margem menos tolerante. No geral, o modelo se comporta melhor para essas variações de ambiente. A precisão baixa pode ser ajustada com a diminuição da margem, mas isso faria a revocação cair também. No entanto, para uma autenticação, é mais viável baixar a tolerância, porque negar autorização para um estudante cadastrado é menos problemático que autorizar um aluno sem cadastro.

Já para o MobileNet, a constância foi observada na variação de margens. Isso acontece porque a distância entre os positivos e negativos determinados pelo modelo, possivelmente, ficam distantes da margem, em sua maioria. Logo, ao aumentar a tolerância, os negativos continuam acima do valor especificado, mantendo os resultados das métricas próximos. O modelo, assim como o ResNet, não tem uma precisão tão alta, mas a falta de sensibilidade à variação das margens dificulta a possibilidade de um estado ótimo para a rede. A revocação não passa por problemas para os testes com os valores selecionados, mas deve diminuir ao encontrar uma margem ótima menor que 0.5. Também é possível observar que esse modelo tem uma pequena melhora ao identificar variações de posição, por ter um aumento na precisão no *dataset* CelebA. Vale destacar que esta é uma arquitetura mais leve, portanto, o desempenho esperado para ela é menor que para as outras em um ambiente geral, mas deve ter menos penalidades de desempenho na aplicação em dispositivos embarcados.

O FaceNet, por sua vez, apresentou valores muito interessantes. Os resultados de precisão foram altíssimos, principalmente para o *dataset* LFW, que atingiu 99.36%, apesar da baixíssima revocação. A margem, nesse caso, foi certeira na filtragem de amostras negativas, mas negou a autenticação de muitos exemplos positivos. Isso foi melhor ajustado na margem 0.75, que balaceou muito mais as métricas, mantendo a precisão em um nível alto, como pode ser visto em uma comparação do *F1-score* para as duas margens. É possível que haja uma grande evolução na revocação se a margem aumentar suavemente. No entanto, a acurácia se mantém Em uma análise geral, esse modelo se mostra ser superior para a aplicação, já que a precisão para imagens diversas se manteve em níveis altos. Além disso, apesar da precisão baixar nos outros conjuntos de imagens, o valor de 79% atingido ao utilizar o Yale foi inédito, evidenciando que a variação de iluminação não afeta tão drasticamente esse modelo.

Para fins de conclusão, o modelo FaceNet se mostrou mais estável e preciso, mantendo um equilíbrio semelhante aos concorrentes, mas com a precisão mais alta. Com isso, essa arquitetura pode ser considerada mais adequada para ser a base para a rede siamesa que será utilizada nesse sistema de reconhecimento facial.

5 Conclusão

Este capítulo apresenta as conclusões do estudo trabalhado nesta tese, com foco na avaliação final do estudo e do modelo, os objetivos concluídos, as limitações e trabalhos futuros, assim como as considerações finais.

5.1 Síntese e Objetivos Concluídos

Desde o advento da computação há algumas décadas, os processos se tornaram cada vez menos repetitivos e burocráticos, até onde a segurança permite. Programas de computadores, *hardwares* complexos, bancos de dados, aplicações em nuvem, etc. foram invenções que de certa forma revolucionaram as indústrias e instituições. De forma análoga, a IA surgiu como forma de adaptar a capacidade humana de resolver problemas de tomada de decisão complexos sem que alguém esteja necessariamente envolvido. Essa ferramenta foi se tornando mais popular no decorrer do século 21, sendo cada vez mais essencial na otimização de processos. Dessa forma, muitas aplicações de IA foram criadas para resolver muitos tipos de problema e entre eles está a visão computacional. O desafio de tornar o computador sensível a imagens evoluiu ao ponto de ser possível autenticar o acesso das pessoas a certos locais utilizando o reconhecimento facial. O principal objetivo dessa tese é criar um sistema de autenticação utilizando essa tecnologia, explorando os melhores meios de desenvolvê-lo. Após uma análise de estudos relacionados publicados nos últimos anos, várias abordagens foram avaliadas. Mas no fim, foi proposta uma arquitetura de redes siamesas que utilizam redes neurais convolucionais e uma função de perda baseada em tripletos de imagem. Foram testadas algumas arquiteturas que funcionariam como rede base, assim como conjuntos de dados e hiperparâmetros, para concluir que a aplicação desse *software* é viável e atinge bons resultados e tem bastante capacidade de melhoria.

Como dito anteriormente, o foco principal dessa tese era o desenvolvimento de um sistema de reconhecimento facial capaz de identificar de forma precisa os usuários de auxílio alimentação do IFMA. A partir deste objetivo maior, foi definido inicialmente quatro objetivos menores, dentre os quais, três foram concluídos.

- **O1.** Estudar o estado da arte do reconhecimento facial: essa tarefa foi concluída no capítulo 2, onde houve uma pesquisa sobre os melhores algoritmos já produzidos para um sistema de reconhecimento facial.
- **O2.** Desenvolver um *software* funcional e preciso de reconhecimento facial: esse objetivo também foi concluído, sendo o principal de toda a pesquisa. O software completo não foi apresentado neste texto, mas diante de todos os desafios, é um desafio trivial aplicar o modelo a uma interface. O modelo é preciso e seguro, tendo atingido uma precisão de 99%.
- **O3:** Trabalhar na implementação de um sistema de armazenamento seguro para os dados dos alunos: este objetivo não foi concluído, devido ao hiperfoco no trabalho dos outros. Pode ser algo a ser implementado futuramente como continuação da pesquisa. No entanto, vale ressaltar que a segurança dos dados dos alunos é de extrema importância e deve ser respeitada.
- **O4:** Desenvolver uma interface de utilização simples e intuitiva: como atestado no capítulo anterior, uma interface para a utilização do modelo foi feita, apesar de ser muito básica, ela é simples e intuitiva, e ainda pode ser bastante melhorada.

Portanto, pode-se dizer que foi uma tese bem-sucedida, pois, apesar de haver um objetivo sem conclusão, é uma tarefa que pode ser feita em trabalhos futuros.

5.2 Trabalhos Futuros e Limitações

Tão importante quanto a produção de um trabalho é a consciência de que sempre há o que melhorar. Portanto, apesar dos possíveis problemas, o estudo e a pesquisa nunca acaba e a comunidade científica sempre estará se auxiliando. Dito isso, é saudável a discussão das falhas e fatores a serem aprimorados.

Como apresentado na etapa de testes, o modelo não é capaz de lidar de forma adequada com imagens afetadas por diferentes iluminações, posições do rosto e variações de expressão. Isso ocorreu, provavelmente, por causa de um possível equívoco na seleção das imagens de treino, o que pode ter feito o modelo se apegar demais nos pontos de iluminação e usar eles demasiadamente nas previsões. Em um trabalho futuro, será possível que haja um estudo sobre a aplicação de diferentes conjuntos de dados ou de uma filtragem de imagens para treinamento que possa evitar esse pequeno overfitting. No geral, esse foi o grande defeito do software, pois a acurácia e revocação baixas foram causadas por ele.

Para a aplicação final em um sistema de autenticação, seria necessária uma comunicação com o *front-end* da instituição para a obtenção das imagens base de cada estudante com o objetivo de compor a base de dados local do *software*. Neste ponto, é importante que as requisições

sejam diárias e que as imagens obtidas sejam aplicadas como entrada na rede para que haja sempre uma base de *embeddings* atualizada. O ato de armazenar os *embeddings* é essencial para garantir a proteção contra fraudes, evitando que terceiros adicionem registros não autorizados na base. Além disso, deve-se remover as imagens do armazenamento local visando proteger a imagem dos alunos que poderiam ficar vulneráveis no dispositivo em ambiente público. Outro fator interessante para melhoria da aplicação é o desenvolvimento de uma interface inteligente e mais estética, de modo que a utilização seja mais agradável e intuitiva. E por fim, a instalação do software nos equipamentos e os testes do programa no ambiente para o qual ele foi projetado.

5.3 Considerações Finais

Neste tópico, encerra-se o presente documento, após um grande tempo de trabalho investido na pesquisa. Apesar de todas as dificuldades, representou um período de foco e estudo que foi retribuído com muitos aprendizados e experiências. No entanto, foram justamente essas dificuldades que geraram o esforço necessário para a obtenção do conhecimento que foi adquirido durante o estudo. Mesmo tendo metas não concluídas, este trabalho foi de grande importância para o autor, que, com certeza, auxiliará, não só a comunidade acadêmica do IFMA, como o seu próprio futuro.

Referências

[1] F. Almeida, J. Duarte Santos, e J. Augusto Monteiro, "The Challenges and Opportunities in the Digitalization of Companies in a Post-COVID-19 World", *IEEE Engineering Management Review*, vol. 48, nº 3, p. 97–103, set. 2020, doi: 10.1109/EMR.2020.3013206.

[2] E. Winter, A. Costello, M. O'Brien, e G. Hickey, "Teachers' use of technology and the impact of Covid-19", *Irish Educational Studies*, vol. 40, nº 2, p. 235–246, abr. 2021, doi: 10.1080/03323315.2021.1916559.

[3] W. F. Avelino e J. G. Mendes, "A realidade da educação brasileira a partir da Covid-19", *Boletim de Conjuntura (BOCA)*, vol. 2, nº 5, abr. 2020, doi: 10.5281/zenodo.3759679.

[4] "IFMA - Instituto Federal de Educação, Ciência e Tecnologia do Maranhão". Acessado: 5 de janeiro de 2024. [Online]. Disponível em: https://portal.ifma.edu.br/inicio/

[5] J. de O. Leite, "As múltiplas determinações do Programa Nacional de Assistência Estudantil – PNAES nos governos Luiz Inácio Lula da Silva", *Universidade Federal de Pernambuco*, 2015.

[6] BRASIL, *Decreto nº 7234*. Dispõe sobre o Programa Nacional de Assistência Estudantil, 2010. Acessado: 3 de janeiro de 2024. [Online]. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2007-2010/2010/decreto/d7234.htm

[7] D. Moher *et al.*, "Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement", *Syst Rev*, vol. 4, nº 1, p. 1, dez. 2015, doi: 10.1186/2046-4053-4-1.

[8] "Google Acadêmico". Acessado: 5 de janeiro de 2024. [Online]. Disponível em: https://scholar.google.com/

[9] “Institute of Eletrical and Eletronic Engineers Xplore Search Source”, Acessado: 5 de janeiro de 2024. [Online]. Disponível em: https://ieeexplore.ieee.org/search/advanced

[10] “Kaggle”. Acessado: 6 de janeiro de 2024. [Online]. Disponível em: https://www.kaggle.com

[11] “Zenodo”. Acessado: 6 de janeiro de 2024. [Online]. Disponível em: https://zenodo.org

[12] “Instituto Politécnico do Porto”. Acessado: 7 de janeiro de 2024. [Online]. Disponível em: https://www.ipp.pt/

[13] Vivek Muraleedharan, “Medium”, What is Linear Discriminant Analysis (LDA). Acessado: 8 de maio de 2024. [Online]. Disponível em: https://vivekmuraleedharan73.medium.com/what-is-linear-discriminant-analysis-lda-7e33ff59020a

[14] N. Singhal, V. Ganganwar, M. Yadav, A. Chauhan, M. Jakhar, e K. Sharma, “COMPARATIVE STUDY OF MACHINE LEARNING AND DEEP LEARNING ALGORITHM FOR FACE RECOGNITION”, *Jordanian Journal of Computers and Information Technology*, nº 0, p. 1, 2021, doi: 10.5455/jjcit.71-1624859356.

[15] I. Adjabi, A. Ouahabi, A. Benzaoui, e A. Taleb-Ahmed, “Past, Present, and Future of Face Recognition: A Review”, *Electronics (Basel)*, vol. 9, nº 8, p. 1188, jul. 2020, doi: 10.3390/electronics9081188.

[16] S. M. Saleem Abdullah e A. M. Abdulazeez, “Facial Expression Recognition Based on Deep Learning Convolution Neural Network: A Review”, *Journal of Soft Computing and Data Mining*, vol. 02, nº 01, abr. 2021, doi: 10.30880/jscdm.2021.02.01.006.

[17] M. O. Oloyede, G. P. Hancke, e H. C. Myburgh, “A review on face recognition systems: recent approaches and challenges”, *Multimed Tools Appl*, vol. 79, nº 37–38, p. 27891–27922, out. 2020, doi: 10.1007/s11042-020-09261-2.

[18] W. Ali, W. Tian, S. U. Din, D. Iradukunda, e A. A. Khan, “Classical and modern face recognition approaches: a complete review”, *Multimed Tools Appl*, vol. 80, nº 3, p. 4825–4880, jan. 2021, doi: 10.1007/s11042-020-09850-1.

[19] S. Saeed, A. A. Shah, M. K. Ehsan, M. R. Amirzada, A. Mahmood, e T. Mezgebo, “Automated Facial Expression Recognition Framework Using Deep Learning”, *J Healthc Eng*, vol. 2022, p. 1–11, mar. 2022, doi: 10.1155/2022/5707930.

[20] A. Anwar, “Medium”, Difference between AlexNet, VGGNet, ResNet, and Inception. Acessado: 8 de maio de 2024. [Online]. Disponível em: https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96

[21] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge”, set. 2014.

[22] M. Wang e W. Deng, “Deep face recognition: A survey”, *Neurocomputing*, vol. 429, p. 215–244, mar. 2021, doi: 10.1016/j.neucom.2020.10.081.

[23] S. Khan, E. Ahmed, M. H. Javed, S. A. A Shah, e S. U. Ali, “Transfer Learning of a Neural Network Using Deep Learning to Perform Face Recognition”, em *2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, IEEE, jul. 2019, p. 1–5. doi: 10.1109/ICECCE47252.2019.8940754.

[24] J. Hernandez-Ortega, J. Galbally, J. Fierrez, e L. Beslay, “Biometric Quality: Review and Application to Face Recognition with FaceQnet”, jun. 2020.

[25] H. Du, H. Shi, D. Zeng, X.-P. Zhang, e T. Mei, “The Elements of End-to-end Deep Face Recognition: A Survey of Recent Advances”, *ACM Comput Surv*, vol. 54, nº 10s, p. 1–42, jan. 2022, doi: 10.1145/3507902.

[26] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, e K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, fev. 2016.

[27] K. Simonyan e A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, *In Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, abr. 2014.

[28] Y. Sun, D. Liang, X. Wang, e X. Tang, “DeepID3: Face Recognition with Very Deep Neural Networks”, fev. 2015.

[29] G. B. Huang, M. Mattar, T. Berg, e E. Learned-Miller, “Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments”, em *Workshop on Faces in “Real-Life” Images: Detection, Alignment, and Recognition*, Marseille, France, out. 2008. [Online]. Disponível em: https://inria.hal.science/inria-00321923

[30] Md. T. H. Fuad *et al.*, “Recent Advances in Deep Learning Techniques for Face Recognition”, *IEEE Access*, vol. 9, p. 99112–99142, 2021, doi: 10.1109/ACCESS.2021.3096136.

[31] J. Yang *et al.*, “Neural Aggregation Network for Video Face Recognition”, em *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jul. 2017, p. 5216–5225. doi: 10.1109/CVPR.2017.554.

[32] S. Das, “Medium”, CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more…. Acessado: 8 de maio de 2024. [Online]. Disponível em: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

[33] M.-T. Chiu, H.-Y. Cheng, C.-Y. Wang, e S.-H. Lai, “RGB-D Face Recognition With Identity-Style Disentanglement and Depth Augmentation”, *IEEE Trans Biom Behav Identity Sci*, vol. 5, nº 3, p. 334–347, jul. 2023, doi: 10.1109/TBIOM.2022.3233769.

[34] C. Jiang, S. Lin, W. Chen, F. Liu, e L. Shen, “PointFace: Point Cloud Encoder-Based Feature Embedding for 3-D Face Recognition”, *IEEE Trans Biom Behav Identity Sci*, vol. 4, nº 4, p. 486–497, out. 2022, doi: 10.1109/TBIOM.2022.3197437.

[35] M. Pang, Y.-M. Cheung, Q. Shi, e M. Li, “Iterative Dynamic Generic Learning for Face Recognition From a Contaminated Single-Sample Per Person”, *IEEE Trans Neural Netw Learn Syst*, vol. 32, nº 4, p. 1560–1574, abr. 2021, doi: 10.1109/TNNLS.2020.2985099.

[36] J. Wang, C. Zheng, X. Yang, e L. Yang, “EnhanceFace: Adaptive Weighted SoftMax Loss for Deep Face Recognition”, *IEEE Signal Process Lett*, vol. 29, p. 65–69, 2022, doi: 10.1109/LSP.2021.3125267.

[37] Q. Wang e G. Guo, “AAN-Face: Attention Augmented Networks for Face Recognition”, *IEEE Transactions on Image Processing*, vol. 30, p. 7636–7648, 2021, doi: 10.1109/TIP.2021.3107238.

[38] H. Qiu, D. Gong, Z. Li, W. Liu, e D. Tao, “End2End Occluded Face Recognition by Masking Corrupted Features”, *IEEE Trans Pattern Anal Mach Intell*, vol. 44, nº 10, p. 6939–6952, out. 2022, doi: 10.1109/TPAMI.2021.3098962.

[39] Y. Zhu, M. Ren, H. Jing, L. Dai, Z. Sun, e P. Li, “Joint Holistic and Masked Face Recognition”, *IEEE Transactions on Information Forensics and Security*, vol. 18, p. 3388–3400, 2023, doi: 10.1109/TIFS.2023.3280717.

[40] M. Aly, A. Ghallab, e I. S. Fathi, “Enhancing Facial Expression Recognition System in Online Learning Context Using Efficient Deep Learning Model”, *IEEE Access*, vol. 11, p. 121419–121433, 2023, doi: 10.1109/ACCESS.2023.3325407.

[41] T. de Freitas Pereira, A. Anjos, e S. Marcel, “Heterogeneous Face Recognition Using Domain Specific Units”, *IEEE Transactions on Information Forensics and Security*, vol. 14, nº 7, p. 1803–1816, jul. 2019, doi: 10.1109/TIFS.2018.2885284.

[42] Z. Deng, X. Peng, Z. Li, e Y. Qiao, “Mutual Component Convolutional Neural Networks for Heterogeneous Face Recognition”, *IEEE Transactions on Image Processing*, vol. 28, nº 6, p. 3102–3114, jun. 2019, doi: 10.1109/TIP.2019.2894272.

[43] A. Greco, A. Saggese, M. Vento, e V. Vigilante, “A Convolutional Neural Network for Gender Recognition Optimizing the Accuracy/Speed Tradeoff”, *IEEE Access*, vol. 8, p. 130771–130781, 2020, doi: 10.1109/ACCESS.2020.3008793.

[44] M. Alansari, O. A. Hay, S. Javed, A. Shoufan, Y. Zweiri, e N. Werghi, “GhostFaceNets: Lightweight Face Recognition Model From Cheap Operations”, *IEEE Access*, vol. 11, p. 35429–35446, 2023, doi: 10.1109/ACCESS.2023.3266068.

[45] K. Chen, T. Yi, e Q. Lv, “LightQNet: Lightweight Deep Face Quality Assessment for Risk-Controlled Face Recognition”, *IEEE Signal Process Lett*, vol. 28, p. 1878–1882, 2021, doi: 10.1109/LSP.2021.3109781.

[46] C.-Y. Low, A. B.-J. Teoh, e J. Park, “MIND-Net: A Deep Mutual Information Distillation Network for Realistic Low-Resolution Face Recognition”, *IEEE Signal Process Lett*, vol. 28, p. 354–358, 2021, doi: 10.1109/LSP.2021.3053480.

[47] F. Mokhayeri e E. Granger, “Video Face Recognition Using Siamese Networks With Block-Sparsity Matching”, *IEEE Trans Biom Behav Identity Sci*, vol. 2, nº 2, p. 133–144, abr. 2020, doi: 10.1109/TBIOM.2019.2949364.

[48] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, e S. Belongie, “Feature Pyramid Networks for Object Detection”, em *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jul. 2017, p. 936–944. doi: 10.1109/CVPR.2017.106.

[49] P. C. Pedro Neto, J. R. Pinto, F. Boutros, N. Damer, A. F. Sequeira, e J. S. Cardoso, “Beyond Masks: On the Generalization of Masked Face Recognition Models to Occluded Face Recognition”, *IEEE Access*, vol. 10, p. 86222–86233, 2022, doi: 10.1109/ACCESS.2022.3199014.

[50] S. Chokkadi, “A Study on various state of the art of the Art Face Recognition System using Deep Learning Techniques”, *International Journal of Advanced Trends in Computer Science and Engineering*, p. 1590–1600, ago. 2019, doi: 10.30534/ijatcse/2019/84842019.

[51] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, abr. 2017.

[52] M. D. Putro, A. Priadana, D.-L. Nguyen, e K.-H. Jo, “A Faster Real-time Face Detector Support Smart Digital Advertising on Low-cost Computing Device”, em *2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, jul. 2022, p. 171–178. doi: 10.1109/AIM52237.2022.9863289.

[53] Md. N. Islam Opu, T. K. Koly, A. Das, e A. Dey, “A Lightweight Deep Convolutional Neural Network Model for Real-Time Age and Gender Prediction”, em *2020 Third International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, IEEE, dez. 2020, p. 1–6. doi: 10.1109/ICAECC50550.2020.9339503.

[54] F. Es-Sabery, A. Hair, J. Qadir, B. Sainz-De-Abajo, B. Garcia-Zapirain, e I. Torre-Diez, “Sentence-Level Classification Using Parallel Fuzzy Deep Learning Classifier”, *IEEE Access*, vol. 9, p. 17943–17985, 2021, doi: 10.1109/ACCESS.2021.3053917.

[55] P. Arafin, A. M. Billah, e A. Issa, “Deep learning-based concrete defects classification and detection using semantic segmentation”, *Struct Health Monit*, vol. 23, nº 1, p. 383–409, jan. 2024, doi: 10.1177/14759217231168212.

[56] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, e A. Zisserman, “VGGFace2: A Dataset for Recognising Faces across Pose and Age”, em *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, IEEE, maio 2018, p. 67–74. doi: 10.1109/FG.2018.00020.

[57] D. Yi, Z. Lei, S. Liao, e S. Z. Li, “Learning Face Representation from Scratch”, nov. 2014.

[58] “Google Colab”. Acessado: 4 de novembro de 2024. [Online]. Disponível em: https://colab.google/

[59] “Microsoft Azure”. Acessado: 4 de novembro de 2024. [Online]. Disponível em: https://azure.microsoft.com/

[60] K. Wongsuphasawat *et al.*, “Visualizing Dataflow Graphs of Deep Learning Models in TensorFlow”, *IEEE Trans Vis Comput Graph*, vol. 24, nº 1, p. 1–12, jan. 2018, doi: 10.1109/TVCG.2017.2744878.

[61] B. Dong, J. Bai, T. Xu, e Y. Zhou, “Large Language Models in Education: A Systematic Review”, em *2024 6th International Conference on Computer Science and Technologies in Education (CSTE)*, IEEE, abr. 2024, p. 131–134. doi: 10.1109/CSTE62025.2024.00031.

[62] J. Liu, X. Tang, L. Li, P. Chen, e Y. Liu, “ChatGPT vs. Stack Overflow: An Exploratory Comparison of Programming Assistance Tools”, em *2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*, IEEE, out. 2023, p. 364–373. doi: 10.1109/QRS-C60940.2023.00105.

[63] T. Elvira, T. T. Procko, J. O. Couder, e O. Ochoa, “Digital Rubber Duck: Leveraging Large Language Models for Extreme Programming”, em *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, IEEE, jul. 2023, p. 295–304. doi: 10.1109/CSCE60160.2023.00051.

[64] G. Koch, R. Zemel, e R. Salakhutdinov, “Siamese neural networks for one-shot image recognition”, *ICML deep learning workshop*, vol. 2, p. 1–30, 2015.

[65] R. R. Varior, M. Haloi, e G. Wang, “Gated Siamese Convolutional Neural Network Architecture for Human Re-identification”, 2016, p. 791–808. doi: 10.1007/978-3-319-46484-8_48.

[66] H. Liu, J. Feng, M. Qi, J. Jiang, e S. Yan, “End-to-End Comparative Attention Networks for Person Re-Identification”, *IEEE Transactions on Image Processing*, vol. 26, nº 7, p. 3492–3506, jul. 2017, doi: 10.1109/TIP.2017.2700762.

[67] E. Ahmed, M. Jones, e T. K. Marks, “An improved deep learning architecture for person re-identification”, em *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jun. 2015, p. 3908–3916. doi: 10.1109/CVPR.2015.7299016.

[68] “Face Recognition Dataset - Oneshot Learning”, Kaggle. Acessado: 6 de novembro de 2024. [Online]. Disponível em: https://www.kaggle.com/datasets/stoicstatic/face-recognition-dataset

[69] J. Bromley *et al.*, “Signature verification using a ‘siamese’ time delay neural network”, *Intern J Pattern Recognit Artif Intell*, vol. 07, nº 04, p. 669–688, ago. 1993, doi: 10.1142/S0218001493000339.

[70] “AT&T Bell Labs”. Acessado: 7 de novembro de 2024. [Online]. Disponível em: https://about.att.com/sites/labs

[71] H. Wu, Z. Xu, J. Zhang, W. Yan, e X. Ma, “Face recognition based on convolution siamese networks”, em *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, IEEE, out. 2017, p. 1–5. doi: 10.1109/CISP-BMEI.2017.8302003.

[72] F. Qiu, S. Kamata, e L. Ma, “Deep Face Recognition under Eyeglass and Scale Variation Using Extended Siamese Network”, em *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, IEEE, nov. 2017, p. 471–476. doi: 10.1109/ACPR.2017.48.

[73] Z. He, W. Su, Z. Bi, M. Wei, Y. Dong, e G. Xu, “The Improved Siamese Network in Face Recognition”, em *2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, IEEE, dez. 2019, p. 443–446. doi: 10.1109/ICICAS48597.2019.00099.

[74] I. I. Osman e M. S. Shehata, “MODSiam: Moving Object Detection using Siamese Networks”, em *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, IEEE, ago. 2020, p. 1–6. doi: 10.1109/CCECE47787.2020.9255776.

[75] B. Cuan, K. Idrissi, e C. Garcia, “Deep Siamese Network for Multiple Object Tracking”, em *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, ago. 2018, p. 1–6. doi: 10.1109/MMSP.2018.8547137.

[76] Y. Wu, Y. Wang, Y. Li, e Q. Xu, “Optical Satellite Image Change Detection Via Transformer-Based Siamese Network”, em *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, jul. 2022, p. 1436–1439. doi: 10.1109/IGARSS46834.2022.9884408.

[77] A. Siddhant, P. Jyothi, e S. Ganapathy, “Leveraging native language speech for accent identification using deep Siamese networks”, em *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, IEEE, dez. 2017, p. 621–628. doi: 10.1109/ASRU.2017.8268994.

[78] L. J. Jie, M. M. A. Zabidi, S. Sadiah, e A. A.-H. A. Rahman, “Siamese Networks for Speaker Identification on Resource-Constrained Platforms”, *J Phys Conf Ser*, vol. 2622, nº 1, p. 012014, out. 2023, doi: 10.1088/1742-6596/2622/1/012014.

[79] A. Hajavi e A. Etemad, “Siamese Capsule Network for End-to-End Speaker Recognition in the Wild”, em *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, jun. 2021, p. 7203–7207. doi: 10.1109/ICASSP39728.2021.9414722.

[80] J. Zhang, S. Xue, J. L. Li, e J. She, “Automated Plagiarism Detection Model Based On Deep Siamese Network”, em *2022 IEEE 8th International Conference on Cloud Computing and Intelligent Systems (CCIS)*, IEEE, nov. 2022, p. 298–302. doi: 10.1109/CCIS57298.2022.10016354.

[81] R. Hadsell, S. Chopra, e Y. LeCun, “Dimensionality Reduction by Learning an Invariant Mapping”, em *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR’06)*, IEEE, p. 1735–1742. doi: 10.1109/CVPR.2006.100.

[82] F. Schroff, D. Kalenichenko, e J. Philbin, “FaceNet: A unified embedding for face recognition and clustering”, em *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jun. 2015, p. 815–823. doi: 10.1109/CVPR.2015.7298682.

[83] I. William, D. R. Ignatius Moses Setiadi, E. H. Rachmawanto, H. A. Santoso, e C. A. Sari, “Face Recognition using FaceNet (Survey, Performance Test, and Comparison)”, em *2019 Fourth International Conference on Informatics and Computing (ICIC)*, IEEE, out. 2019, p. 1–6. doi: 10.1109/ICIC47613.2019.8985786.

[84] F. Cahyono, W. Wirawan, e R. Fuad Rachmadi, “Face Recognition System using Facenet Algorithm for Employee Presence”, em *2020 4th International Conference on Vocational Education and Training (ICOVET)*, IEEE, set. 2020, p. 57–62. doi: 10.1109/ICOVET50258.2020.9229888.

[85] C. Wang, R. Xu, Y. Zhang, S. Xu, e X. Zhang, “Retinal Vessel Segmentation Via Context Guide Attention Net With Joint Hard Sample Mining Strategy”, em *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, IEEE, abr. 2021, p. 1319–1323. doi: 10.1109/ISBI48211.2021.9433813.

[86] C. Li, C. Yan, X. Xiang, J. Lai, H. Zhou, e D. Tang, “HADGEO: Image Based 3-DoF Cross-View Geo-Localization with Hard Sample Mining”, em *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, abr. 2024, p. 3520–3524. doi: 10.1109/ICASSP48485.2024.10445839.

[87] K. Chen, Y. Chen, C. Han, N. Sang, C. Gao, e R. Wang, “Improving Person Re-Identification by Adaptive Hard Sample Mining”, em *2018 25th IEEE International Conference on Image Processing (ICIP)*, IEEE, out. 2018, p. 1638–1642. doi: 10.1109/ICIP.2018.8451129.