UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

JESSÉ DA COSTA ROCHA

RECONHECIMENTO FACIAL DE ALUNOS DE ESCOLA PÚBLICA NO USO DE ÔNIBUS ESCOLAR EM CIDADE INTELIGENTE

BELÉM
2023

JESSÉ DA COSTA ROCHA

RECONHECIMENTO FACIAL DE ALUNOS DE ESCOLA PÚBLICA NO USO DE ÔNIBUS ESCOLAR EM CIDADE INTELIGENTE

Dissertação de Mestrado submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica como requisito para a obtenção do Grau de Mestre em Engenharia Elétrica, na área de Computação Aplicada.

Orientador: Prof. Dr. Carlos Renato Lisboa Francês
Coorientadora: Profa. Dra. Jasmine Priscyla Leite de Araújo

BELÉM
2023

R672r Rocha, Jessé da Costa, 1980 -
Reconhecimento facial de alunos de escola pública no uso do ônibus escolar em cidade inteligente / Jessé da Costa Rocha.-2023
Orientador: Carlos Renato Lisboa Francês.
Coorientadora: Jasmine Priscyla Leite de Araújo
Dissertação (Mestrado) – Universidade Federal do Pará, Instituto deTecnologia. Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2023.
1.Inteligência artificial. 2. Biometria. 3. Estudantes – Processamento de dados. I.Título.

CDD 22.ed. 006.3

Elaborada por Lucicléa S. de Oliveira = CRB/2-648

JESSÉ DA COSTA ROCHA

RECONHECIMENTO FACIAL DE ALUNOS DE ESCOLA PÚBLICA NO USO DE ÔNIBUS ESCOLAR EM CIDADE INTELIGENTE

Dissertação de Mestrado submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica como requisito para a obtenção do Grau de Mestre em Engenharia Elétrica, na área de Computação Aplicada.

Data da aprovação: 04/08/2023

Conceito: Aprovado

BANCA EXAMINADORA

Professor. Dr. Carlos Renato Lisboa Francês
Orientador - PPGEE/UFPA

Professora Dra. Jasmine Priscyla Leite de Araújo
Coorientadora - PPGEE/UFPA

Marcelino Silva da Silva
Avaliador Interno – PPGEE/UFPA

Evelin Helena Silva Cardoso
Avaliadora Externa - UFRA

HUGO PEREIRA KURIBAYASHI
Avaliador Externo - UNIFESSPA

Professor Dr. Diego Lisboa Cardoso
Coordenador do PPGEE/ITEC/UFPA

AGRADECIMENTOS

Agradeço aos meus pais Celene Rocha e José Rocha que sempre se esforçaram para me dar uma boa educação em todos os sentidos.

Aos meus irmãos Joselene Rocha e Josué Rocha pela amizade e incentivo.

A minha esposa Ádria Pacheco pelo amor e companheirismo e pela paciência que teve comigo durante o desenvolvimento deste trabalho.

Aos professores e funcionários do PPGEE da UFPA pelo conhecimento compartilhado. Especialmente para o meu orientador, professor Renato Francês pelas oportunidades de aprendizado que me proporcionou e para minha coorientadora professora Jasmine Araújo pelos sábios conselhos que me deu.

Aos colegas do LPRAD e do projeto de Cidades Inteligentes de Canaã dos Carajás. Especialmente para as professoras Marcela Souza e Evelin Cardoso e para o professor Nandamudi Vijaykumar pela grande ajuda na elaboração do artigo.

Finalmente, a todas as pessoas que direta ou indiretamente contribuíram para que eu atingisse o objetivo de me formar no curso de mestrado em engenharia elétrica com ênfase em computação aplicada.

RESUMO

Nos dias atuais, o desaparecimento de crianças e adolescentes e a evasão escolar são grandes problemas enfrentados por países do mundo todo, em especial os países em desenvolvimento. Este trabalho propõe uma plataforma inteligente de monitoramento dos passos dos estudantes como ferramenta para mitigar esses problemas. Esta plataforma permite identificar os estudantes por meio do reconhecimento facial, e notificar os responsáveis e entidades competentes em diversas situações da vida escolar, tais como: entrada e saída do ônibus escolar, entrada e saída da escola, entrada na cantina escolar etc. O reconhecimento facial, por sua vez, emprega diferentes técnicas de inteligência artificial para reconhecer os estudantes, como: HOG (*Histograms of Oriented Gradients*), SVM (*Support Vector Machine*), CNN (*Convolutional Neural Network*) e KNN (*K-Nearest Neighbors*). Nos testes realizados, o sistema de reconhecimento obteve excelentes resultados em todas as métricas: acurácia de 98,10%, média ponderada da precisão de 99%, média ponderada da revocação de 98% e média ponderada da pontuação f1 de 98%.

Palavras-chave: Inteligência Artificial, Desaparecimento, Reconhecimento Facial, Plataforma Inteligente e Evasão Escolar.

ABSTRACT

In the present days, the disappearance of children and adolescents and school dropout are major problems faced by countries worldwide, in particular, developing countries. This work proposes an intelligent platform for monitoring students' steps as a tool to mitigate these problems. This platform can identify students and notify those responsible and competent authorities in various situations of school life, such as: entering and leaving the school bus, entering and leaving school, entering the school cafeteria, etc. The first application aims to control access to the school bus through facial recognition. Facial recognition, in turn, employs different artificial intelligence techniques to recognize students, such as: HOG (Histograms of Oriented Gradients), SVM (Support Vector Machine), CNN (Convolutional Neural Network) and KNN (K-Nearest Neighbors). In the tests carried out, the recognition system achieved excellent results in all metrics: accuracy 98.10%, weighted average precision 99%, weighted average recall 98%, and weighted average f1-score 98%.

Keywords: Artificial Intelligence, Disappearance, Facial Recognition, Intelligent Platform and School Dropout.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CNN	Convolutional Neural Network
ECA	Estatuto da Criança e do Adolescente
GB	Gigabytes
HTTP	Hypertext Transfer Protocol
HOG	Histograms of Oriented Gradients
IDH	Índice de Desenvolvimento Humano
IoT	Internet of Things
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbor
LFW	Labeled Faces in the Wild
MCP	McCulloch e Pitts
MLP	Multilayer Perceptron
RAM	Random Access Memory
RBF	Radial Basis Functions
ReDESAP	Rede Nacional de Identificação e Localização de Crianças e Adolescentes Desaparecidos
ResNet	Residual Network
RFID	Radio-Frequency Identification
RNA	Rede Neural Artificial
SBC	Single-Board Computer
SMS	Short Message Service
SVM	Support Vector Machine
TRL	Technology Readiness Level
UFPA	Universidade Federal do Pará
USB	Universal Serial Bus
VGG	Visual Geometry Group

SUMÁRIO

1 INTRODUÇÃO

O conceito de Cidade Inteligente tornou-se muito popular em todo o mundo. Infelizmente, algumas cidades afirmam que são inteligentes apenas com base em algum tipo de tecnologia instalada, como iluminação pública inteligente ou sistema de tráfego inteligente. No entanto, para uma cidade se tornar realmente inteligente, é preciso muito mais que apenas tecnologia. Diversos aspectos sociais e ambientais também devem ser considerados para proporcionar uma maior qualidade de vida para as populações urbanas. As autoridades devem olhar para questões como desigualdade social, falta de acesso à água e esgoto, saúde, energia eficiente, habitação, educação, segurança pública etc.

Este trabalho discute e propõe estratégias para minimizar alguns problemas de acesso e permanência na escola enfrentados por crianças e adolescentes. Sendo assim, ele está situado em uma interseção entre a educação e a segurança pública. Nos dias atuais, são inúmeras as ameaças à segurança e à formação plena de crianças e adolescentes, especialmente nos países em desenvolvimento. Dentre essas ameaças, destacam-se o desaparecimento e a evasão escolar.

No Brasil, a Rede Nacional de Identificação e Localização de Crianças e Adolescentes Desaparecidos (ReDESAP) registra cerca de 40.000 desaparecimentos por ano (Rolim *et al.*, 2018). Segundo especialistas, o desaparecimento de um filho por um período prolongado gera nos pais, particularmente na mãe, uma espécie de luto mais difícil de superar do que o luto gerado pela morte de um filho. A mãe não se desvincula do filho desaparecido e isso impede que ela crie novas relações afetivas ou siga nutrindo relações já existentes (Rolim *et al.*, 2018).

O Estatuto da Criança e do Adolescente (ECA), promulgado em 1990, tornou obrigatória a criação de serviços para a identificação de crianças e adolescentes desaparecidos. Desde então, diferentes serviços foram criados pelo governo federal, como é o caso da ReDESAP, e pelos governos estaduais. Um dos serviços estaduais mais antigos e conhecidos é o SOS Crianças Desaparecidas, criado em 1996 no Rio de Janeiro. Desde o início do seu funcionamento, o SOS utiliza a mesma metodologia: divulgar as imagens de crianças e adolescentes desaparecidos em cartazes e esperar que denúncias feitas pela população ajudem a solucionar os casos. Entretanto, ao longo dos anos, essa metodologia tem se mostrado pouco efetiva (Ferreira, 2017), porque tanto a identificação da possível criança desaparecida, feita pelo denunciante, como o encaminhamento da denúncia para as

autoridades responsáveis, que deve ser feito pelos funcionários do SOS, são fortemente influenciados por juízos subjetivos, especialmente por juízos morais. Isto é, as pessoas geralmente esperam que uma criança desaparecida tenha a aparência suja, irritada e maltratada (Ferreira, 2017), o que resulta em um grande número de crianças e adolescentes falsamente identificados como desaparecidos. Sendo assim, o serviço público brasileiro ainda necessita de um sistema mais eficiente de identificação de pessoas desaparecidas que seja menos afetado por juízos subjetivos.

O índice de evasão escolar é igualmente preocupante no Brasil. A taxa de abandono escolar brasileira é a terceira maior dentre os cem países do mundo com os maiores Índice de Desenvolvimento Humano (IDH) (Silva Filho; Araújo, 2017). Como consequência direta, o Brasil apresenta a menor média de escolaridade entre os países da América do Sul (7,2 anos), onde, 52,6% dos brasileiros com 25 anos ou mais ainda não concluíram o ensino médio, muitos dos quais nem sequer começaram a última etapa do ensino básico (Branco *et al.*, 2020).

Vale ressaltar que no Brasil como um todo a evasão e o atraso escolar aumentam à medida que os estudantes vão ficando mais velhos (Branco *et al.*, 2020). O Estado do Pará, por exemplo, não apenas segue a tendência geral do Brasil, como possui os piores índices de abandono escolar nas três etapas da educação básica: 2,1% nos anos iniciais do ensino fundamental, 5,6% nos anos finais do ensino fundamental e 12,8% no ensino médio (Branco *et al.*, 2020).

Embora o abandono escolar seja um problema complexo com muitas causas e de difícil solução, existem alguns indicadores que podem ser usados para prever uma maior probabilidade de evasão escolar. Dentre esses indicadores está o número de faltas de um estudante, isto é, um estudante com um número grande de faltas tem maiores chances de abandonar os estudos (Silva Filho, Araújo, 2017; Oliveira, Magrone, 2021). Sendo assim, monitorar o número de faltas dos discentes e, com base nisso, agir preventivamente de modo a evitar o abandono escolar mostra-se uma estratégia promissora. Vale lembrar que, na maioria das escolas, a frequência diária dos estudantes é registrada apenas em papel e depois transferida para o sistema computacional no fim de cada bimestre letivo, o que deixa o controle de faltas bastante defasado. Uma outra estratégia eficiente no combate à evasão escolar é fortalecer a rede de proteção da criança e do adolescente nos municípios (Branco *et al.*, 2020).

Tendo em vista os problemas acima expostos, o presente trabalho propõe um sistema de reconhecimento facial em um contexto de cidade inteligente capaz de monitorar o trajeto do estudante desde a entrada no ônibus escolar até a entrada na escola, contemplando também o caminho de volta. O sistema também é preparado para notificar os responsáveis do estudante, a escola, o Ministério Público e o Conselho Tutelar em caso de ausência do estudante para que as devidas providências sejam tomadas, prevenindo, ao mesmo tempo, o desaparecimento de crianças e adolescentes e a evasão escolar. O sistema combina tecnologia da informação com inteligência artificial para automaticamente verificar o comparecimento do estudante à escola e gerar relatórios estatísticos para apoiar a tomada de decisão dos agentes interessados. A primeira parte do sistema, já desenvolvida e testada, corresponde ao monitoramento da entrada no ônibus escolar.

1.1 Motivação

Conforme exposto anteriormente, o desaparecimento e a evasão escolar são dois graves problemas, com grandes repercussões sociais no Estado do Pará e no Brasil como um todo. As práticas normalmente adotadas para enfrentar esses desafios mostram-se, até agora, pouco eficientes. Por essa razão, uma nova abordagem é aqui proposta utilizando modernas técnicas de inteligência artificial e de engenharia de software para mitigar esses dois problemas de grande relevância social.

Contudo, o que se pretende aqui não é apenas produzir uma pesquisa teórica, mas também desenvolver um protótipo eficiente e de baixo custo que possa ser utilizado efetivamente, em um futuro próximo, em uma cidade brasileira para melhorar a educação e a segurança pública.

1.2 Objetivo

A presente dissertação tem como objetivo principal propor uma arquitetura para o monitoramento de estudantes em diferentes momentos do cotidiano escolar. Essa arquitetura possui como base um sistema de reconhecimento facial que, por sua vez, utiliza recentes técnicas de inteligência artificial. A arquitetura proposta implementa conceitos modernos da arquitetura de software, do desenvolvimento web e do desenvolvimento para dispositivos móveis.

Dentre os objetivos específicos, três merecem destaque. O primeiro é o desenvolvimento de uma aplicação de controle de acesso ao ônibus escolar com base no reconhecimento facial. O segundo é desenvolver uma aplicação Web para administrar o sistema de reconhecimento facial e notificar os responsáveis dos estudantes bem como as autoridades envolvidas nas questões educacionais. O terceiro é implementar um aplicativo móvel para ser usado pelos responsáveis dos estudantes para que eles possam acompanhar a entrada dos estudantes no ônibus escolar.

1.3 Organização da Dissertação

Os próximos capítulos estão distribuídos da seguinte maneira:

- Capítulo 2: Esclarece a fundamentação teórica, primeiramente explicando de maneira detalhada o surgimento e o funcionamento das redes neurais artificiais e depois abordando as redes do tipo CNN (*Convolutional Neural Network*) que são as mais usadas em visão computacional. No final do capítulo, os demais algoritmos usados neste trabalho e alguns conceitos adicionais são apresentados.
- Capítulo 3: Apresenta os trabalhos relacionados, subdivididos em dois grupos. O primeiro agrupa os artigos dedicados ao monitoramento de estudantes. O segundo reúne artigos centrados na pesquisa do reconhecimento facial.
- Capítulo 4: Explica a arquitetura proposta, expondo inicialmente uma visão geral da arquitetura e mostrando em seguida os detalhes técnicos do sistema.
- Capítulo 5: Detalha a primeira aplicação que será implantada que é a aplicação de controle do acesso ao ônibus escolar por meio de reconhecimento facial. Nesse tópico também é mostrada primeiramente uma visão geral do sistema e posteriormente os detalhes técnicos.
- Capítulo 6: Mostra os resultados dos testes da primeira aplicação, especificando a base de dados utilizada, o procedimento usado nos teste, as principais métricas e também alguns exemplos da saída do sistema.
- Capítulo 7: Apresenta o estágio atual de desenvolvimento da primeira aplicação, mostrando o hardware do protótipo, a interface gráfica da aplicação web e a interface gráfica da aplicação móvel.
- Capítulo 8: Apresenta a conclusão, ressaltando as contribuições do presente trabalho e apontando possíveis desdobramentos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Considerações Iniciais

Desde o seu surgimento em meados do século XX, o computador moderno superou em muito a capacidade humana de realizar cálculos, sendo usado de forma eficiente já na Segunda Guerra Mundial para calcular trajetórias de mísseis e para decifrar códigos secretos. Todavia, uma capacidade que se desenvolveu de forma espontânea nos seres humanos permaneceu inacessível durante décadas para os computadores: a capacidade de ver, reconhecer e diferenciar objetos, animais e pessoas. Contudo, em anos recentes, a assim chamada visão computacional desenvolveu-se de forma considerável, graças, em grande parte, ao aparecimento de uma técnica de inteligência artificial denominada CNN *(Convolutional Neural Network*) que é uma forma mais complexa de rede neural artificial (Chollet, 2018).

O grande marco histórico desse processo de desenvolvimento foi o desafio ImageNet de 2012 (Chollet, 2018). Esse desafio consistia em classificar em mil categorias diferentes um conjunto de imagens coloridas baseado em um treinamento feito com mais de um milhão de imagens. A equipe de Alex Krizhevsky e Geoffrey Hinton venceu o ImageNet 2012 alcançando uma acurácia de 83,6%. Nos anos seguintes, o desafio ImageNet foi dominado pelas CNNs, que atingiram a acurácia de 96,4% em 2015.

No mesmo ano, um grupo de pesquisadores do Google apresentou ao mundo o FaceNet, um sistema de reconhecimento facial que tinha como núcleo a CNN. Segundo os desenvolvedores do projeto, o sistema atingiu uma acurácia máxima de 99,63% usando a base de dados LFW (*Labeled Faces in the Wild*) com alinhamento da face (Schroff; Kalenichenko; Philbin, 2015). Esse resultado é considerado superior à capacidade humana de reconhecer pessoas. Desde então, muitas empresas passaram a oferecer esse serviço, usando tecnologia semelhante, o que tornou o reconhecimento facial algo presente no dia a dia das pessoas.

Todavia, o presente trabalho tem como objetivo dar um passo além e apresentar uma arquitetura completa de um sistema de reconhecimento facial em um contexto de cidade inteligente. Essa arquitetura contempla uma série de serviços aplicados à área de educação e segurança pública utilizando equipamentos de baixo custo e um paradigma moderno de desenvolvimento web.

Contudo, para um bom entendimento do tema, é preciso antes explicar o que são redes neurais artificiais, suas principais características e princípios de funcionamento. Posteriormente, também faz-se necessário discorrer sobre as redes neurais artificiais do tipo CNN que são as mais usadas em visão computacional. Na parte final do capítulo, os demais algoritmos usados neste trabalho e outros importantes conceitos e métricas são apresentados.

2.2 Redes Neurais Artificiais

Uma das principais técnicas utilizadas atualmente para fazer um computador aprender por meio de exemplos é a Rede Neural Artificial (RNA). Uma RNA pode aprender utilizando três diferentes métodos de aprendizagem: o aprendizado por meio de reforço, o não supervisionado e o supervisionado. Neste ponto, vale ressaltar que a RNA usada no presente estudo foi treinada por meio de um aprendizado supervisionado.

Os tópicos seguintes são dedicados à análise das RNAs. Primeiramente, é apresentada a origem histórica e os conceitos fundamentais. Em seguida, é exposto um tipo específico de RNA chamado de rede Perceptron de múltiplas camadas. Por fim, é explicado em detalhes o algoritmo usado no treinamento da rede Perceptron de múltipla camada denominado *backpropagation.*

2.2.1 Breve Histórico das RNAs

De acordo com Braga, Carvalho e Ludemir (2016), a história das RNAs começou em 1943 quando Warren McCulloch e Walter Pitts criaram o primeiro modelo de neurônio artificial de que se tem notícia. McCulloch era psicólogo e neurofisiologista. Pitts, por sua vez, era matemático. Os dois pesquisadores apresentaram à comunidade científica um modelo lógico-matemático de um neurônio artificial que foi chamado de neurônio MCP (McCulloch e Pitts), junto com a descrição de sua capacidade computacional. O principal objetivo desses pesquisadores era criar um modelo para explicar o funcionamento do sistema nervoso.

Em 1949, Donald Hebb apresentou um estudo detalhado sobre como um neurônio biológico aprende e propôs aplicar o mesmo processo ao neurônio artificial. Segundo ele, um neurônio aprende por meio da variação dos pesos de suas entradas, ou seja, reforçando ou inibindo as conexões sinápticas com outros neurônios.

Em 1958, Frank Rosenblatt apresentou um novo modelo de neurônio artificial, o Perceptron. O Perceptron era basicamente um neurônio MCP com pesos de entrada ajustáveis. Por meio desse artifício, era possível treinar o Perceptron para funcionar como um separador linear, isto é, separar, por meio de uma linha reta, os valores de entrada em duas classes distintas.

Contudo, em 1969, Minsky e Papert publicaram um estudo demonstrando que o Perceptron serviria apenas e tão somente como um separador linear, sendo incapaz de aprender funções mais complexas. Esse estudo desencorajou novas pesquisas em RNA durante toda a década de 70 do século passado.

Foi só em 1982 que as pesquisas em RNA foram retomadas, após John Hopfield publicar um artigo mostrando as propriedades das associações de neurônios artificiais. Isso abriu caminho para o surgimento das redes Perceptron de múltiplas camadas, capazes de aprender funções não-lineares complexas. O desenvolvimento posterior do algoritmo de treinamento *backpropagation*, eficiente para as redes de múltiplas camadas, lançou as bases para todos os avanços que se seguiram.

2.2.2 Neurônios Biológicos X Neurônios Artificiais

Conforme abordado anteriormente, o desenvolvimento dos neurônios artificiais foi baseado nos neurônios biológicos. No entanto, cabe ainda destacar quais são as semelhanças e as diferenças existentes entre esses dois modelos.

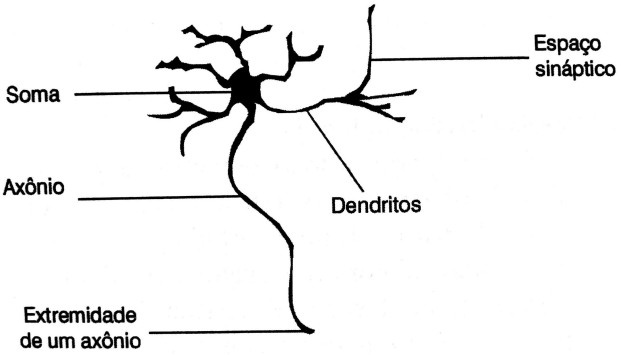
O neurônio biológico possui três partes principais: o corpo celular (soma), os dendritos e o axônio, como mostra a Figura 1. Os dendritos são as entradas do neurônio que recebem os impulsos nervosos vindos de outros neurônios. Dos dendritos, os impulsos passam até chegarem no corpo celular.

No corpo celular, os impulsos são processados e, de acordo com o caso, enviados ou não para outros neurônios. O axônio, por sua vez, é o terminal de saída do neurônio. É por meio dele que o impulso nervoso é enviado para outros neurônios. A conexão entre o axônio de um neurônio anterior e o dendrito de um neurônio posterior é feita por meio de uma sinapse.

Na sinapse, neurotransmissores são liberados pelo axônio anterior e recebidos pelo dendrito do neurônio posterior. Certos tipos de neurotransmissores têm a função de gerar um impulso nervoso, enquanto que outros tipos de neurotransmissores têm a função oposta, ou

seja, inibir a propagação de um impulso nervoso. Sendo assim, “As sinapses funcionam como válvulas, e são capazes de controlar a transmissão de impulsos – isto é, o fluxo da informação – entre os neurônios na rede neural” (Braga; Carvalho; Ludemir, 2016, p. 6).

Figura 1 – Neurônio Biológico.



Fonte: Braga, Carvalho e Ludemir (2016)

É importante ressaltar que o corpo do neurônio combina os diversos impulsos nervosos recebidos pelos dendritos e, caso o resultado ultrapasse um certo limiar de excitação, transmite um impulso por meio do axônio. Em outras palavras, quanto maior o acúmulo de fortes impulsos nas entradas do neurônio, maior é a probabilidade de o neurônio disparar um impulso para o neurônio seguinte.

Por outro lado, o neurônio artificial concebido por McCulloch e Pitts, ou seja, o modelo MCP, reproduz de forma simplificada a estrutura e o funcionamento do neurônio biológico, conforme mostrado na Figura 2. Nele, os dendritos são substituídos por terminais de entrada (X1, X2, X3, …, Xn) e o axônio é substituído por um terminal de saída (Y).

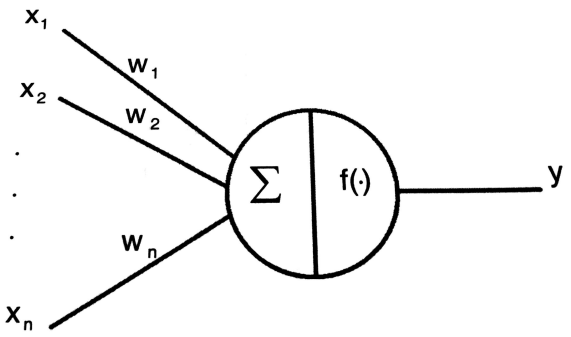
As sinapses são substituídas por pesos associados aos terminais de entrada (W1, W2, W3, …, Wn), sendo que os pesos com valores positivos representam as sinapses excitatórias e os pesos com valores negativos representam as sinapses inibitórias. A combinação dos impulsos nervosos recebidos pelos dendritos é substituída pela soma ponderada das entradas, como mostra a Equação 1. Por fim, o limiar de excitação é substituído por uma função de ativação.

O funcionamento da referida função será objeto de estudo do próximo tópico, junto com as principais arquiteturas de RNAs. Antes, porém, é preciso destacar uma importante diferença entre as redes neurais biológicas e as redes neurais artificiais. Nas redes neurais artificiais, todos os neurônios de uma mesma camada são ativados simultaneamente. Nas redes neurais biológicas, ao contrário, os neurônios de uma mesma camada podem ser

acionados em tempos distintos uma vez que não há um *clock* central para sincronizar todos os neurônios.

$$\sum_{i=1}^{n} XiWi \qquad (1)$$

Figura 2 – Neurônio Artificial.



Fonte: Braga, Carvalho e Ludemir (2016)

2.2.3 Funções de Ativação e Arquiteturas de RNAs

Dentre os diferentes tipos de funções de ativação, quatro merecem destaque: degrau, sigmoidal, linear e gaussiana. A função de ativação usada pelo neurônio MCP é a degrau. Ela funciona como mostrado na Equação 2, na qual q representa o limiar de ativação escolhido.

$$f(x) = 1 \text{ se } \sum_{i=1}^{n} XiWi >= q, \text{ ou } f(x) = 0 \text{ se } \sum_{i=1}^{n} XiWi < q \qquad (2)$$

A função de ativação sigmoidal é uma aproximação contínua da função degrau, ou seja, permite uma transição suave entre o mínimo = 0 e o máximo = 1, sem sofrer nenhuma descontinuidade. Nisso, ela difere da função degrau que é descontínua e apenas serve para ligar ou desligar o neurônio. A Equação 3 descreve a função sigmoidal. Essa função de ativação é a mais usada em RNAs de múltiplas camadas porque facilita a aproximação de funções contínuas.

$$f(x) = 1/(1 + e^{-x}) \qquad (3)$$

Uma função de ativação linear do tipo f(x) = X também pode ser usada para resolver problemas simples, mas, nesse caso, a RNA fica reduzida a uma espécie de calculadora de álgebra linear que multiplica o vetor de entrada pelo vetor dos respectivos pesos e depois soma o resultado.

Por fim, a função gaussiana é usada nas RNAs do tipo RBF (*Radial Basis Functions*), que estão fora do escopo deste trabalho. Em todo caso, a Equação 4 descreve essa função, em que m é o ponto médio e r é o raio de abertura da função.

$$f(x) = e(-(x-m)^2/r^2) \qquad (4)$$

As arquiteturas de RNAs, por sua vez, podem ser divididas em dois grandes grupos: redes *feedforward* e redes com recorrência. As redes *feedforward* podem ter uma ou múltiplas camadas. Elas possuem esse nome porque os sinais de entrada sempre fluem da esquerda para a direita, ou seja, elas sempre são alimentadas para a frente (*feedforward*).

Uma rede *feedforward* de uma única camada já é capaz de resolver problemas com múltiplas variáveis, mas com restrições quanto à complexidade. Por outro lado, basta acrescentar uma camada intermediária para que uma rede *feedforward* seja capaz de aproximar qualquer função contínua, independente da sua complexidade. Essas redes são consideradas estáticas porque as saídas dependem apenas dos valores atuais das entradas. Nas redes recorrentes, ao contrário, os valores de saída dependem dos valores atuais de entrada, assim como dos valores atuais da saída, uma vez que a saída é reconectada com as entradas de neurônios de camadas anteriores.

2.2.4 Aprendizado por Correção de Erro

As RNAs que aprendem de forma supervisionada utilizam um algoritmo de treinamento que busca minimizar o erro, ou seja, a diferença entre a saída esperada e a saída calculada pela rede. Em termos matemáticos, o erro pode ser definido como na Equação 5, na qual $y_d(t)$ é a saída desejada e y(t) é a saída atual calculada pela rede.

$$e(t)= y_d(t) - y(t) \qquad (5)$$

Para minimizar o erro, é necessário ajustar os pesos da rede de forma incremental, ou seja, a cada interação, os pesos sofrem pequenos ajustes. Esses ajustes são calculados em função do erro obtido e(t), da taxa de aprendizado α, que determina o tamanho do passo que será dado em direção ao erro mínimo, e da entrada de cada neurônio.

A Equação 6 é uma forma genérica para atualizar os pesos de uma RNA e é utilizada, com algumas modificações, na maioria dos algoritmos de aprendizado por correção de erro. Nela, $w_i(t)$ e $x_i(t)$ correspondem, respectivamente, ao peso e à entrada i do neurônio. É importante notar que o ajuste do peso deve ser proporcional ao produto do erro pelo valor da entrada. Contudo, como será mostrado no tópico seguinte, RNAs de múltiplas camadas possuem uma equação de atualização dos pesos mais complexa que utiliza o gradiente de erro no lugar do erro e(t).

$$w_i(t+1)= w_i(t) + \alpha e(t)x_i(t) \qquad (6)$$

2.2.5 Rede Perceptron de Múltiplas Camadas

Conforme mencionado na breve história das RNAs exposta anteriormente, a rede Perceptron de múltiplas camadas, em inglês *Multilayer Perceptron* (MLP), foi desenvolvida para superar a limitação do Perceptron de camada única. O Perceptron simples possui apenas uma camada e uma função de ativação degrau. Isso limita o Perceptron simples a ser um mero separador linear.

A explicação é a seguinte: a saída do Perceptron é 1 se $\sum_{i=1}^{n} XiWi >= q$ ou 0 se $\sum_{i=1}^{n} XiWi <$ q, conforme mostrado na Equação 2. Considerando, por exemplo, um Perceptron com duas entradas (x1 e x2), chega-se a equação da reta x1w1 + x2w2 – q = 0, que divide o espaço de entrada em duas partes. A primeira é a classe A1, que está acima da reta, e a segunda é a classe A2, que está abaixo da reta.

Para superar a limitação do Perceptron simples é preciso introduzir a não-linearidade na RNA de duas maneiras: por meio de uma função de ativação não-linear e por meio do uso de múltiplas camadas. A RNA do tipo MLP satisfaz essas duas exigências, o que a torna potencialmente capaz de modelar qualquer fenômeno do mundo real, que são, em sua maioria, não-lineares, desde que se encontre uma estrutura de dados adequada para representá-lo. Uma MLP com função de ativação sigmoidal e uma camada intermediária é

capaz de aproximar qualquer função contínua, linear ou não. Por sua vez, uma MLP com função de ativação sigmoidal e duas camadas intermediárias é capaz de aproximar qualquer função, contínua ou não. Isso acontece, como será detalhado a seguir, porque as camadas intermediárias de uma MLP transformam "o problema descrito pelo conjunto de dados no espaço de entrada em uma representação tratável para a camada de saída da rede" (Braga; Carvalho; Ludemir, 2016, p. 67).

2.2.6 Arquitetura de Redes MLP

Uma rede MLP típica com uma camada de entrada, uma camada intermediária e uma camada de saída realiza duas transformações sucessivas para resolver um problema de classificação não-linear. A primeira transformação é feita na camada intermediária, e nela, uma função do tipo $H(x; w_H)$, em que x representa o vetor com os valores de entrada e w_H o vetor com os pesos da camada intermediária, transforma a superfície não-linear apresentada na entrada da rede em uma superfície linear. Na camada de saída, uma função $Y(H(x; w_H); w_s)$, em que w_s representa o vetor com os pesos da camada de saída e $H(x; w_H)$ a saída da camada anterior, executa a separação linear. É importante destacar que a maioria dos problemas encontrados podem ser resolvidos por uma MLP com apenas uma camada intermediária e que o uso de camadas intermediárias desnecessárias pode dificultar a convergência da rede para um erro mínimo.

Um outro importante parâmetro a ser considerado em uma rede MLP é a função de ativação. Em uma rede MLP, as funções de ativação para as camadas intermediárias devem ser não-lineares. Dentre todas as funções de ativação não-lineares possíveis, a sigmoidal é preferida porque possui uma derivada simples de ser calculada. Essa derivada, por sua vez, é usada no cálculo do gradiente de erro, como será mostrado posteriormente. Por outro lado, a função de ativação da camada de saída pode ser linear ou não-linear, não havendo, nesse caso, nenhuma restrição.

O parâmetro mais complexo a ser definido em uma rede MLP é, sem dúvida, a quantidade de neurônios na camada intermediária. Apesar das inúmeras pesquisas realizadas nessa área, não existe uma resposta definitiva para essa questão. O que se sabe é que o número de neurônios na camada intermediária determina a complexidade da rede e, por conseguinte, a complexidade dos problemas que a rede é capaz de resolver. Sendo assim, a quantidade de neurônios na camada intermediária deve ser proporcional à complexidade do

problema que se quer resolver. Esse princípio geral, contudo, não esgota a questão uma vez que, na maioria dos casos, a complexidade do problema a ser resolvido não é conhecida plenamente até que ele seja resolvido. Para contornar essa dificuldade, foram desenvolvidas diferentes abordagens, dentre as quais vale destacar a abordagem multiobjetivo.

Segundo a abordagem multiobjetivo, o projeto de uma RNA deve ser orientado por dois objetivos básicos: em primeiro lugar, a RNA projetada deve ser capaz de convergir para um erro mínimo; em segundo lugar, a RNA deve possuir a menor complexidade possível capaz de atingir o primeiro objetivo. Para atingir esses dois objetivos simultaneamente, o projetista da RNA deve usar o método das tentativas, isto é, projetar uma RNA com uma complexidade mínima e verificar se ela converge para o erro mínimo. Caso isso não aconteça, o projetista deve aumentar gradativamente a complexidade da rede até que ela convirja para o erro mínimo.

A implementação de uma RNA com complexidade acima do necessário, isto é, o superdimensionamento de uma RNA, traz consigo dois problemas. O primeiro e mais evidente é o desnecessário aumento do custo computacional. O segundo e mais grave é o indesejado aumento do universo das possíveis soluções que atendem ao critério de minimização do erro, sem que a quantidade de possíveis soluções que se aproximam da função geradora dos dados aumente.

Desta forma, quanto maior o número de neurônios desnecessários, maior é a dificuldade para se achar uma solução ótima. Pode-se dizer que o superdimensionamento de uma RNA cria uma espécie de paradoxo da escolha, no qual o aumento das possibilidades de escolha aumenta não apenas a dificuldade da escolha, como também as chances de se fazer uma escolha ruim. Isso reforça a ideia de que funções mais simples costumam ser generalizações melhores, enquanto que funções mais complexas se ajustam melhor aos dados de teste, mas se afastam de uma melhor generalização ao tentarem colocar todos os pontos dentro da curva, inclusive os ruídos presentes na base de dados.

2.2.7 Treinamento de Redes MLP

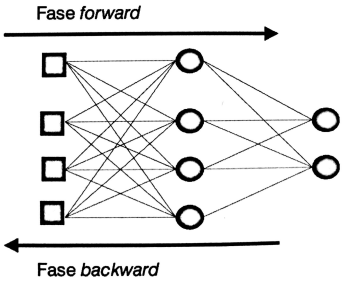
Em linhas gerais, o treinamento de redes MLP é dividido em dois momentos distintos: o primeiro momento é chamado de fase *forward* e o segundo momento é chamado de fase *backward*. Na fase *forward*, os dados de entrada são propagados na rede da esquerda para a direita, passando de camada em camada até que a saída final seja produzida. Já na fase

backward, o erro é propagado da direita para a esquerda e, com base nesse erro, os pesos da rede são ajustados. Esse processo se repete até que um erro mínimo seja atingido.

Para mais detalhes, a fase *forward* pode ser dividida em três passos: 1) As entradas da rede recebem o vetor de entrada e as saídas dos neurônios da primeira camada intermediária são calculadas; 2) As saídas dos neurônios da primeira camada intermediária passam a ser as entradas da camada seguinte, e as saídas dos neurônios dessa camada são calculadas. O processo é repetido até chegar na camada de saída; 3) A saída do neurônio da camada de saída é confrontada com o valor esperado e o erro é calculado.

A fase *backward*, por sua vez, pode ser dividida em quatro passos: 1) O erro encontrado na fase anterior é usado para calcular o gradiente de erro. Em seguida, o gradiente de erro é utilizado para ajustar diretamente os pesos do neurônio da camada de saída; 2) O gradiente de erro da camada de saída é propagado para os neurônios da camada intermediária. Para que isso ocorra com propriedade, o gradiente de erro é multiplicado pelo peso existente entre o neurônio da camada de saída e o neurônio da camada intermediária. O processo se repete para cada neurônio da camada intermediária; 3) O erro propagado no passo anterior é usado para calcular o gradiente de erro de cada neurônio da camada intermediária. Posteriormente, os pesos de cada neurônio da camada intermediária são ajustados com base nos seus respectivos gradientes de erro; 4) O processo se repete até que todos os pesos da rede sejam ajustados. A Figura 3 mostra o sentido do fluxo na fase *forward* e na fase *backward.*

Figura 3 – Fluxos das duas fases do treinamento.



Fonte: Braga, Carvalho e Ludemir (2016)

2.2.8 O Algoritmo *BackPropagation*

O algoritmo mais usado no treinamento de redes MLP é o *backpropagation* e, por isso, ele merece ser analisado com toda a riqueza de detalhes. Para tal fim, são mostrados a seguir todos os passos do referido algoritmo com todas as fórmulas utilizadas por ele, de acordo com as explicações dadas por Rosa (2011), Russell e Norvig (2013) e Braga, Carvalho e Ludemir

(2016). Vale ressaltar que aqui foi usada como base uma rede MLP com uma camada de entrada indexada pela letra i, uma camada intermediária (escondida) indexada pela letra j e uma camada de saída indexada pela letra k.

Passo 1: Inicialização.

Todos os pesos da rede MLP, aqui representados pela letra W, e os níveis das bias (limiar de ativação), aqui representados pela letra θ, são inicializados randomicamente.

Passo 2: Ativação.

A rede é ativada por meio da aplicação das entradas(*input*) x1(p), x2(p), ..., xn(p) e saídas desejadas(*target*) yd1(p), yd2(p), ..., ydn(p). É importante notar que, neste algoritmo, a letra y sempre representa uma saída e a letra x uma entrada. Em seguida:

a) A saída real de cada neurônio da camada escondida é calculada com a seguinte fórmula: yj(p) = sigmoid $[\sum_{i=1}^{n} Xi(p)xWij(p)$- θj], em que n representa o número de entradas do neurônio j da camada escondida e sigmoid é a função de ativação $f(x)=1/(1 + e^{-x})$.

b) A saída real de cada neurônio da camada de saída é calculada usando a seguinte fórmula: yk(p) = sigmoid $[\sum_{i=1}^{m} Xjk(p)xWjk(p)$- θk], em que m representa o número de entradas do neurônio k da camada de saída.

Passo 3: Treinamento dos pesos.

Os pesos da rede MLP são atualizados por meio da propagação *backward* dos erros associados aos neurônios da camada de saída. É importante lembrar que a camada de saída pode ter um ou mais neurônios.

a) O gradiente de erro para cada neurônio da camada de saída é calculado com a fórmula: δk(p) = yk (p) × [1 - yk(p)] × ek(p), em que ek(p) = ydk(p) - yk(p) e "yk(p) × [1 – yk(p)]" é a derivada da função de ativação sigmoidal.

As correções dos pesos de cada neurônio da camada de saída são calculadas por meio da equação: Δwjk(p) = α × yj(p) × δk(p), em que α representa a taxa de aprendizagem. Lembrando que a saída yj da camada escondida é a entrada da camada de saída.

Os pesos de cada neurônio da camada de saída são atualizados usando a equação:

wjk(p + 1) = wjk(p) + Δwjk(p).

b) O gradiente de erro para cada neurônio da camada escondida é calculado com a fórmula: δj(p) = yj(p) × [1-yj(p)] × $\sum_{k=1}^{l} \delta k(p) x Wjk(p)$, em que l representa o número de neurônios da camada de saída e $\sum_{k=1}^{l} \delta k(p) x Wjk(p)$ representa a somatória ponderada dos gradientes de erro dos neurônios da camada de saída.

As correções dos pesos de cada neurônio da camada escondida são calculadas com a equação: Δwij(p) = α × xi(p) × δj(p).

Os pesos de cada neurônio da camada escondida são atualizados usando a fórmula: Wij(p + 1) = wij(p) + Δwij(p).

Passo 4: Iteração.

A iteração p é incrementada de 1, o algoritmo volta para o passo 2(ativação) e repete todo o processo até que o critério de erro selecionado seja satisfeito.

Normalmente, o algoritmo *backpropagation* chega ao fim quando o erro quadrático atinge um valor mínimo previamente determinado ou se, depois de um número determinado de épocas, o erro quadrático não diminuir. O erro quadrático é, basicamente, a média da somatória dos erros obtidos para cada par de saída desejada e saída real elevada ao quadrado, o que é mostrado na Equação 7, na qual p é o número de exemplos de treinamento. É importante destacar que o erro quadrático é calculado para o conjunto completo dos dados de treinamento e que sempre que o passo 4 percorre o conjunto completo de treinamento, uma época é anotada. Sendo assim, o gráfico de convergência da rede deve mostrar a relação época X erro quadrático.

$$e = 1/p \sum_{i=1}^{p} (ydk - yk)^2 \qquad (7)$$

2.3 Redes Neurais Artificiais do Tipo CNN

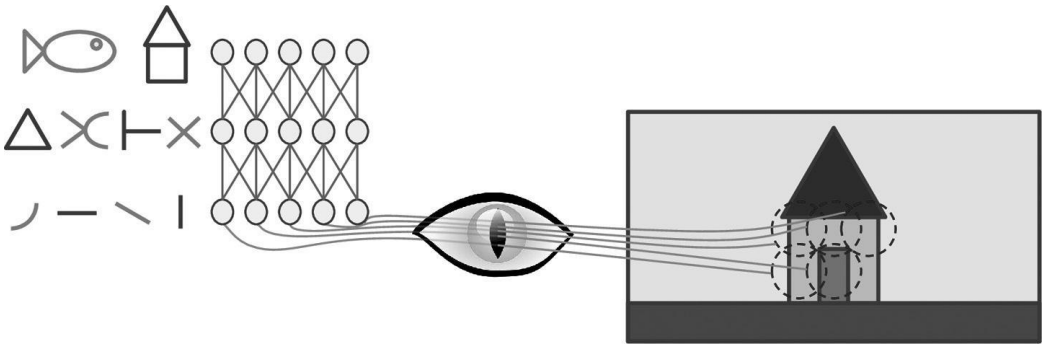
Conforme mencionado antes, nos dias atuais, o campo da visão computacional é quase que inteiramente dominado pelas redes neurais artificiais do tipo CNN. Isso ocorre porque as

CNNs imitam a estrutura do córtex visual do cérebro humano o que torna mais fácil o aprendizado de padrões visuais, mas introduz desafios novos como o uso de uma grande quantidade de camadas, o treinamento de filtros e a utilização de camadas de *pooling*. Todas essas questões serão explicadas de forma pormenorizada nos tópicos seguintes e, ao final, será apresentada a arquitetura completa de uma CNN.

2.3.1 A Estrutura do Córtex Visual e as Camadas Convolucionais

De acordo com Géron (2021) cada neurônio que forma o córtex visual humano reage a estímulos situados em uma pequena região do campo visual. Essa região, que é um subconjunto do campo visual total, é denominada de campo receptivo. Contudo, neurônios com o mesmo campo receptivo podem reagir a padrões visuais diferentes, por exemplo, um neurônio pode reagir a uma linha horizontal e um outro neurônio reagir a uma linha vertical. Ademais, os neurônios das camadas inferiores do córtex visual recebem as imagens captadas pelo olho humano, possuem um campo receptivo menor e reagem a padrões visuais mais simples. Os neurônios das camadas superiores recebem os sinais processados pelas camadas inferiores, possuem campos receptivos maiores e reagem a padrões visuais mais complexos que são combinações dos padrões mais simples. A Figura 4 ilustra a estrutura do córtex visual humano.

Figura 4 – Ilustração do córtex visual humano.



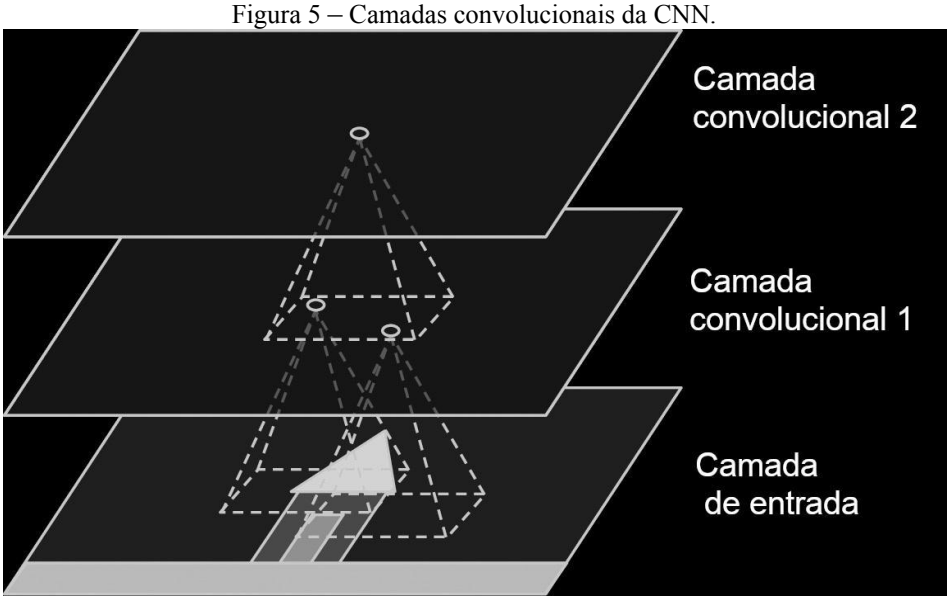
Fonte: Géron (2021)

De forma análoga, em uma CNN, as camadas convolucionais são dispostas da seguinte maneira: os neurônios da primeira camada convolucional são ligados aos pixels da imagem de entrada que pertencem aos seus respectivos campos receptivos. Os neurônios da

segunda camada convolucional, por sua vez, são interligados aos neurônios situados em uma região delimitada da primeira camada.

Para diminuir o custo computacional, uma camada posterior é sempre menor do que a camada anterior, e para tornar possível essa redução, os campos receptivos dos neurônios são deslocados verticalmente e horizontalmente. Esse processo é chamado de *stride.* Nesse caso, o tamanho da saída de uma camada de convolução é igual ao tamanho da entrada dividida pelo *stride*. As bordas da imagem de entrada também podem ser completadas com zeros, possibilitando assim um deslocamento do campo receptivo até o fim da imagem sem que ele entre em um espaço inválido. A Figura 5 mostra duas camadas convolucionais com campos receptivos retangulares.

Figura 5 – Camadas convolucionais da CNN.



Fonte: Géron (2021)

2.3.2 Filtros e Mapas de Características

Como foi dito anteriormente, no córtex visual humano, neurônios com o mesmo campo receptivo podem ser mais sensíveis a determinados padrões visuais e menos sensíveis a outros. Essa característica é imitada em uma CNN com o uso de filtros, que nada mais são do que matrizes numéricas que têm como objetivo destacar padrões específicos em uma imagem.

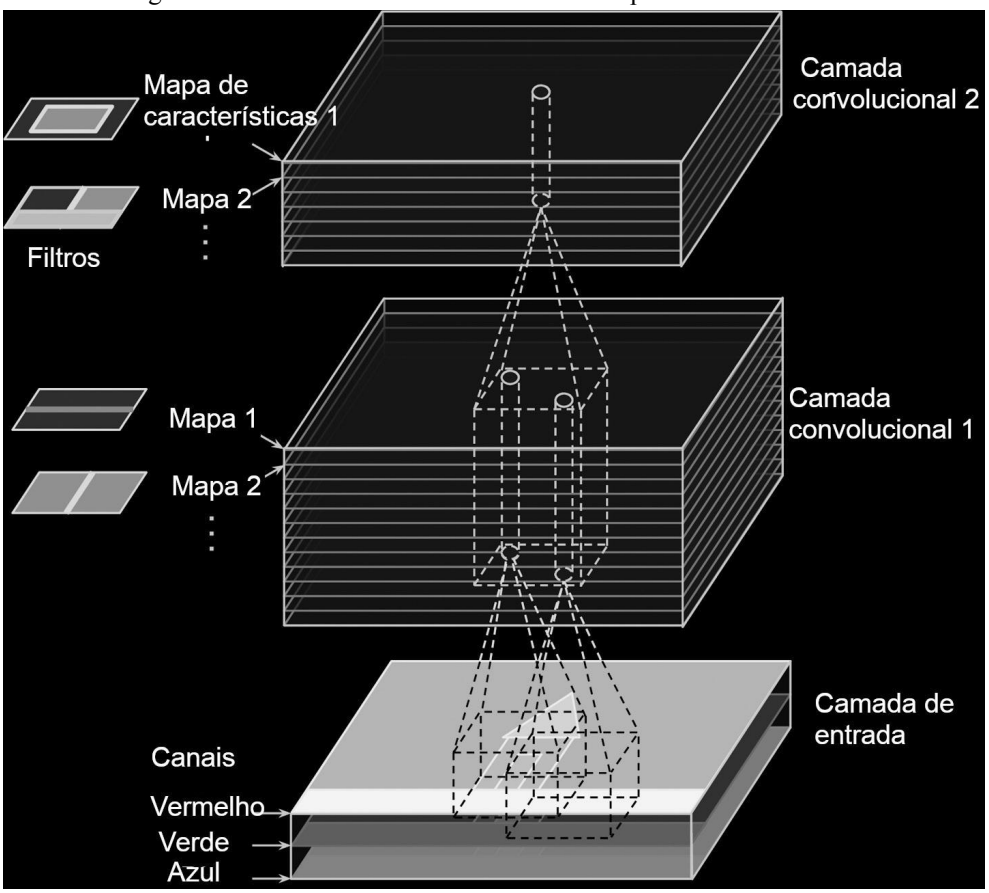
Por exemplo, para destacar linhas verticais deve-se usar uma matriz com 1s na coluna central e 0s nas demais colunas. Dessa forma, quando a matriz que representa a imagem de

entrada for multiplicada pela matriz do filtro, que, nesse caso, representa um quadro preto com uma linha vertical branca no meio, o resultado gerado na saída será uma imagem com todas as suas linhas verticais destacadas.

A multiplicação entre a matriz da imagem e a matriz do filtro é justamente a convolução que dá nome a esse tipo de rede neural. Em termos mais técnicos, "Uma convolução é uma operação matemática que superpõe uma função sobre a outra e calcula a integral de sua multiplicação pontual" (Géron, 2021, p. 347). Durante o treinamento de uma CNN, a rede aprende quais filtros ajudam a solucionar melhor a tarefa proposta, e aprende também a combinar esses filtros para formar padrões mais complexos em camadas superiores. Por esse motivo, pode-se dizer que os filtros estão para as CNN assim como os pesos estão para as redes neurais tradicionais.

Todavia, uma camada convolucional não utiliza apenas um filtro. Ao invés disso, é comum o uso de dezenas ou mesmo de centenas de filtros em uma mesma camada. Para cada filtro, a camada convolucional gera uma saída chamada de mapa de característica. Sendo assim, a saída desse tipo de camada possui três dimensões, como mostra a Figura 6.

Figura 6 – Camadas convolucionais com os mapas de características.



Fonte: Géron (2021)

2.3.3 Camada de *Pooling*

Uma outra camada muito importante em uma CNN é a camada de *pooling*. Ela tem por objetivo reduzir o tamanho da imagem de entrada e, com isso, reduzir também o custo computacional do treinamento da rede. Ao subamostrar a matriz que representa a imagem de entrada, a camada de *pooling* também diminui a possibilidade de um *overfitting*, ou seja, de um sobreajuste no processo de treinamento. Uma vantagem adicional da camada de *pooling* é gerar um certo grau de invariância que permite que um determinado padrão seja identificado em diferentes posições da imagem.

Existem dois tipos principais de camada de *pooling*: a máxima e a média. A máxima é o tipo mais usado e funciona selecionando o maior valor de entrada em um determinado campo receptivo. Por exemplo, uma camada máxima de *pooling* com *Kernel* 2x2 e *stride d*e 2 seleciona o maior valor numérico dentro de uma matriz 2x2, que é uma pequena amostra da imagem de entrada. Posteriormente, essa janela é deslocada lateralmente e uma nova amostragem é feita. Como nesse caso o *stride* é igual a 2*,* a imagem de saída da camada terá a metade da largura e da altura da imagem de entrada. A camada média de *pooling* funciona de forma análoga, sendo a única diferença que ela calcula a média do campo receptivo ao invés de selecionar o valor máximo. Todavia, a camada máxima de *pooling* é preferida principalmente porque conserva as características mais relevantes da imagem de entrada e descarta as características menos relevantes.

2.3.4 A Arquitetura Básica da CNN

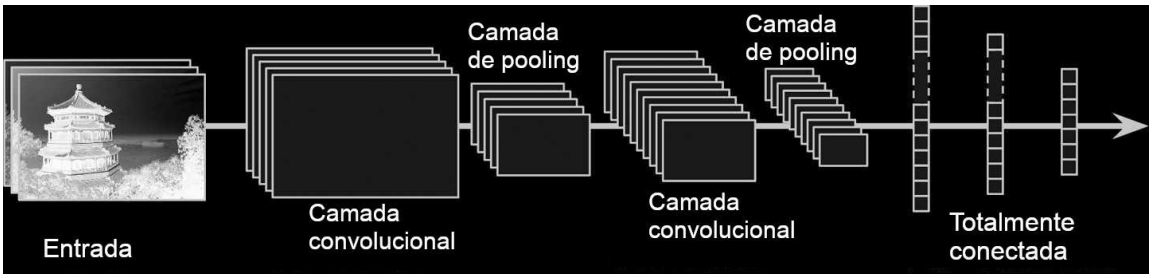
A arquitetura básica de uma CNN é constituída por três grandes seções. O primeiro segmento corresponde à camada de entrada. Esta é a camada mais simples, destinada apenas a receber os dados de entrada, que no caso de uma imagem, é uma matriz numérica.

A segunda seção é a parte central e mais importante de uma CNN, e é composta pelo empilhamento de uma ou duas camadas convolucionais seguido por uma camada de *pooling*. Essa estrutura pode ser repetida diversas vezes, dependendo dos requisitos do problema que se quer resolver. É preciso ressaltar que é comum dobrar o número de filtros a cada bloco de camadas convolucionais, o que também dobra o número de mapas de características. Por

outro lado, a imagem torna-se cada vez menor, porque cada camada de *pooling* divide por dois a altura e a largura da imagem.

Por fim, a terceira e última parte de uma CNN começa com uma camada *Flatten*, que é usada para transformar a matriz que representa os dados em um vetor de uma única dimensão. Logo em seguida, os dados, na forma de um vetor, entram em uma rede neural tradicional formada por uma camada de entrada, uma ou duas camadas escondidas e uma camada de saída, que também é a saída final da CNN. A Figura 7 mostra de maneira simplificada a arquitetura aqui descrita.

Figura 7 – Arquitetura básica da CNN.



Fonte: Géron (2021)

2.3.5 A Rede CNN do Tipo ResNet

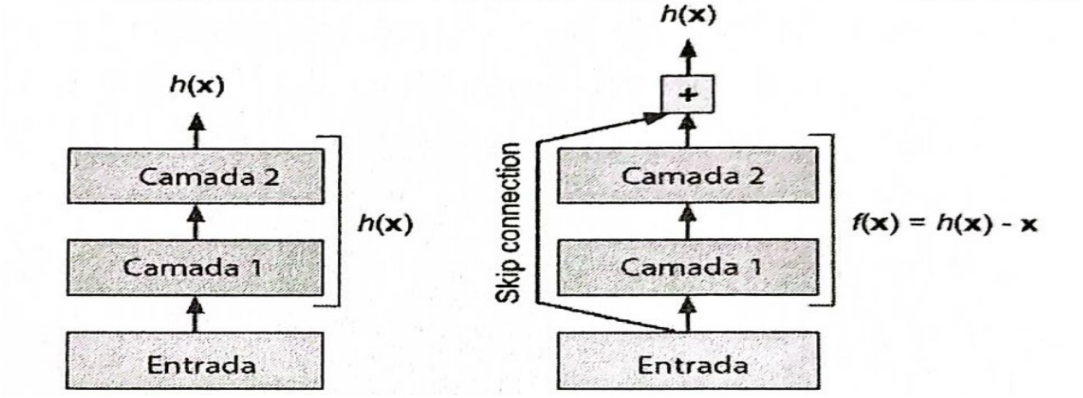
A arquitetura básica de uma CNN possui diversas variações que foram desenvolvidas ao longo dos últimos anos. O presente trabalho utilizou uma variante chamada de ResNet. O nome ResNet vem da abreviação em inglês de “Rede Residual”, o que destaca a maior novidade desse tipo de rede, que é usar o aprendizado residual. Essa técnica consiste em conectar a entrada de uma camada com a saída de uma outra camada localizada mais acima na pilha, o que acelera o processo de treinamento nos casos em que a função alvo é próxima da função identidade.

A explicação é simples: normalmente, no começo do treinamento de uma rede neural, seus pesos são números próximos de zero, o que faz com que a saída da rede também seja um conjunto de números próximos de zero. Por outro lado, quando a camada de entrada é conectada diretamente à saída, a rede inicia o processo de aprendizagem modelando a função identidade.

A Figura 8 mostra a estrutura de uma unidade residual. Nela, a ligação direta entre a entrada de uma camada e a saída de uma outra camada subsequente é chamada de *skip connection*. Na esquerda da Figura 8, é possível ver uma rede convencional que busca

modelar a função f(x) = h(x). Na direita da mesma figura, aparece uma unidade residual, nela a entrada x é adicionada a saída forçando a rede a modelar a função f(x) = h(x) – x em vez de h(x), essa diferença é chamada de aprendizado residual.

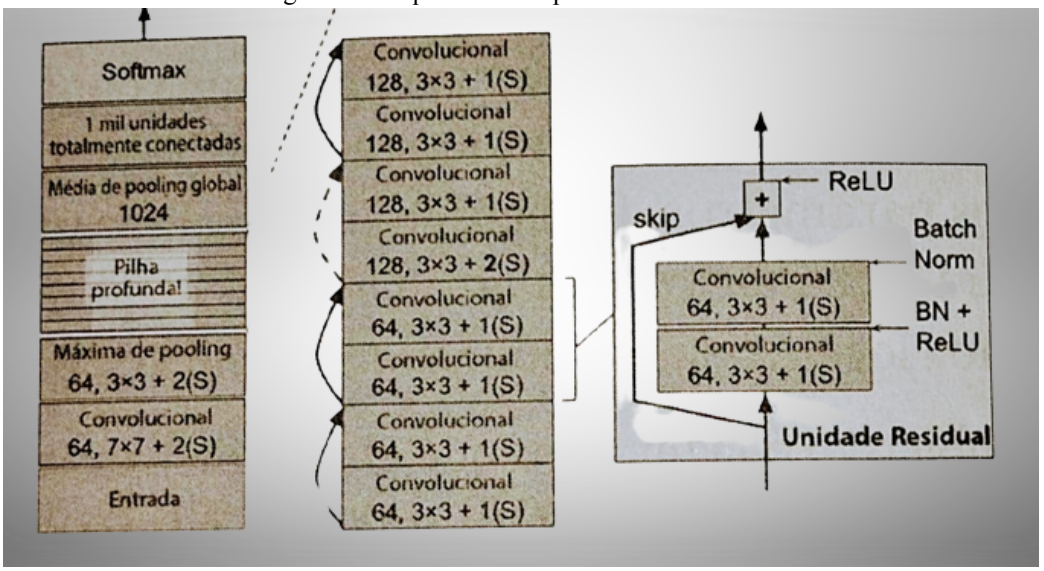
Figura 8 – Unidade residual.



Fonte: Géron (2021)

A Figura 9 mostra a arquitetura completa de uma ResNet. Ela começa com uma camada convolucional seguida por uma camada de *pooling* e termina com uma camada totalmente conectada, exatamente como a arquitetura básica da CNN. A grande diferença está no meio, onde encontra-se a pilha profunda formada por unidades residuais. Essas unidades são constituídas por duas camadas convolucionais que não são seguidas por uma camada de *pooling.* Todas essas unidades usam um kernel 3x3, função de ativação ReLU e *stride* igual a 1.

Figura 9 – Arquitetura completa de uma ResNet.



Fonte: Géron (2021)

Existem também diferentes tipos de ResNets que recebem seus nomes de acordo com a quantidade de camadas que possuem. Por exemplo, uma ResNet-34 possui 34 camadas, contando apenas as camadas convolucionais e a camada totalmente conectada. A ResNet usada neste trabalho é a ResNet-29 que é uma versão reduzida da ResNet-34. Essa rede foi pré-treinada, pelos autores da biblioteca Dlib, com um conjunto de dados de aproximadamente três milhões de faces. Esse conjunto foi formado pela união dos *datasets* VGG Face, Face Scrub e imagens garimpadas da Internet.

O VGG Face é uma base de dados pública, elaborada pelo *Visual Geometry Group*, que contém 2,6 milhões faces de 2.622 pessoas diferentes. O Face Scrub é uma outra base pública com 107.818 faces de 530 pessoas diferentes. O conjunto de dados resultante foi revisado para eliminar os erros de rotulação. Depois da correção, restaram imagens de 7.485 pessoas diferentes.

2.4 Outros Importantes Algoritmos e Conceitos Adicionais

2.4.1 Detalhes Técnicos da Arquitetura Proposta

O Servidor *Back-end* implementa uma API (*Application Programming Interface*) do tipo REST. Trata-se de uma forma moderna e prática de integrar diferentes aplicações sem que seja necessário conhecer os detalhes de como cada aplicação foi implementada. A API aparece então como uma mediadora que recebe e encaminha as requisições e depois devolve uma resposta. Contudo, tanto as requisições como as respostas devem seguir determinados padrões previamente estabelecidos.

A API REST usa o HTTP (*Hypertext Transfer Protocol*), que é o protocolo padrão da Web, e um formato de mensagem chamado de JSON (*JavaScript Object Notation*). O JSON é uma forma simples de registrar as informações e pode ser lido em praticamente qualquer linguagem de programação, o que o torna bastante versátil. Além disso, uma API REST precisa seguir um conjunto de seis restrições ou regras: 1) Deve possuir uma arquitetura cliente-servidor, na qual o cliente solicita os recursos e o servidor fornece os recursos; 2) O estado do cliente não deve ser armazenado no servidor durante a solicitação; 3) Algumas informações de interesse do cliente podem ser armazenadas em cache para agilizar a comunicação entre o cliente e o servidor; 4) As interações entre o cliente e o servidor podem

ser mediada por camadas com recursos adicionais para aumentar a eficiência ou segurança da comunicação; 5) O servidor também pode transferir para o cliente códigos executáveis; 6) A interface deve ser uniforme, ou seja, os recursos devem sempre ser identificados nas solicitações, e as mensagens retornadas devem conter as informações de como os recursos podem ser utilizados.

A API REST aqui proposta utiliza a arquitetura de microsserviços. Tal arquitetura permite subdividir as aplicações em seus elementos constitutivos mínimos, de modo que cada elemento realize uma tarefa específica. Essas tarefas são chamadas de microsserviços e são, ao mesmo tempo, independentes e complementares, compondo uma estratégia que permite um desenvolvimento mais rápido das aplicações, assim como facilita a manutenção do sistema.

Como dito anteriormente, o *Back-end* também utiliza o *framework* Node.js que é um ambiente de execução da linguagem JavaScript que dispensa o uso de um navegador de Internet, tornando assim possível usar essa linguagem em um Servidor Web.

O Node.js possui duas características muito interessantes. A primeira é o processamento de todas as requisições em uma única *thread* chamada de *Event Loop*, o que economiza recursos computacionais. A segunda é a operação de entrada e saída não bloqueante. Isso ocorre porque o *Event Loop* sempre transfere uma operação de entrada e saída para uma *thread* do sistema e não fica aguardando uma resposta e, dessa forma, não bloqueia recursos computacionais. As principais operações de entrada e saída são o acesso ao banco de dados, leitura e escrita de um arquivo do sistema ou o acesso a um serviço externo.

As três principais vantagens do Node.js são: flexibilidade, leveza e facilidade de integração. O Node.js é flexível porque seu gerenciador de pacotes é o maior repositório do mundo. Nele, encontram-se pacotes que podem ser usados nas mais diversas situações. O Node.js é leve porque não utiliza muitos recursos computacionais tanto na instalação quanto na operação. Por fim, o Node.js facilita a integração com o *front-end* (lado do cliente) porque usa a mesma linguagem de programação (JavaScript) e também facilita a integração com bancos de dados não relacionais porque usa a mesma estrutura de dados (JSON).

2.4.2 Detalhes Técnicos da Primeira Aplicação

As duas principais bibliotecas da linguagem Python utilizadas na implementação do sistema de reconhecimento facial são a Dlib e a OpenCV. A biblioteca Dlib é um conjunto de

ferramentas de *machine learning*, *open source*, amplamente usada tanto na academia quanto na indústria. A biblioteca OpenCV, por sua vez, é a mais usada na área de visão computacional, e também é *open source*.

A primeira etapa do reconhecimento facial faz uso do algoritmo HOG (*Histograms of Oriented Gradients*). Esse algoritmo utiliza um filtro para criar duas tabelas com os gradientes da imagem. A primeira tabela guarda as magnitudes, e a segunda, as direções em que os pixels que compõem a imagem variam. Posteriormente, um histograma é gerado com base no ângulo e na magnitude das variações. Como cada padrão de imagem possui um histograma característico, é possível identificar, dessa forma, diferentes padrões de imagens.

A segunda etapa do reconhecimento facial combina os algoritmos HOG e SVM (*Support Vector Machine*). Originalmente, a SVM é um separador linear que classifica os dados de teste com base em uma função linear das principais características dos dados de treinamento. Para encontrar a melhor função linear capaz de separar os dados com a menor quantidade possível de erros de classificação, a SVM inicialmente procura traçar uma faixa com a maior largura possível entre as classes e, em seguida, elege a reta que passa no ponto médio da faixa de separação como o separador linear. Nos casos em que os dados não podem ser linearmente separados, uma penalidade é atribuída para cada dado classificado erroneamente, e o objetivo é combinar uma margem larga com uma penalidade baixa. Sendo a penalidade proporcional à distância da margem, espera-se que na classificação final a SVM cometa apenas erros pequenos (Provost; Fawcett, 2016).

A terceira etapa, por sua vez, usa uma CNN. Conforme dito no tópico que aborda especificamente a rede CNN, esse é o tipo de rede neural artificial mais usado na atualidade em aplicações na área de visão computacional. A principal diferença entre uma CNN e uma rede tradicional é que a CNN aprende padrões locais em uma imagem, enquanto uma rede tradicional aprende padrões globais que envolvem a imagem como um todo (Chollet, 2018). Essa diferença dá à CNN duas vantagens: em primeiro lugar, os padrões aprendidos pela CNN podem ser reconhecidos mesmo quando variam de posição. Em segundo lugar, esses padrões podem ser hierarquizados espacialmente. Isso significa que uma primeira camada da CNN pode aprender um padrão simples em uma imagem e a camada seguinte pode aprender um padrão mais complexo constituído pelo padrão aprendido anteriormente.

Por exemplo: uma primeira camada pode aprender a reconhecer uma curva e uma segunda camada pode aprender a reconhecer os lábios formados por essa curva. A convolução nada mais é do que o produto de duas matrizes, a primeira é a matriz que forma a imagem de

entrada e a segunda é a matriz que forma um filtro. Diferentes filtros foram elaborados para destacar determinados padrões. Dessa forma, uma CNN pode reconhecer diferentes características, indo das mais simples para as mais complexas.

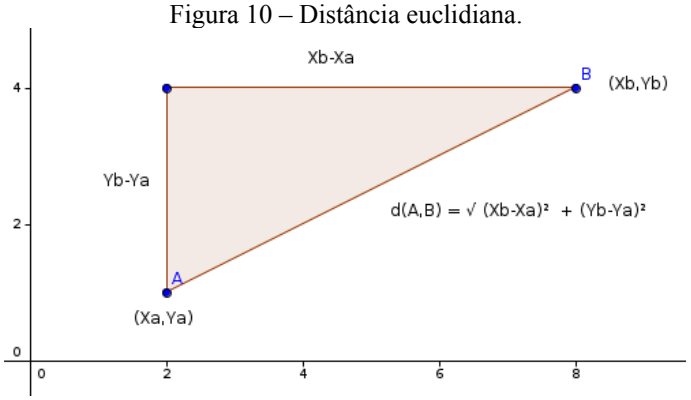
A quarta e última etapa emprega o algoritmo KNN (*K-Nearest Neighbor*), também chamado de algoritmo do vizinho mais próximo, onde k indica o número de vizinhos utilizados na comparação. Vale ressaltar que no presente trabalho k=1, o que significa que apenas o vizinho mais próximo é considerado no momento da classificação. A base desse tipo de algoritmo é a função de distância. Como aqui o objetivo é fazer a comparação entre vetores que são descrições geométricas de faces, todos os valores usados nos cálculos são do mesmo tipo. Nesse caso, a função de distância recomendada é a distância euclidiana, que é simples de entender e fácil de calcular (Provost; Fawcett, 2016). Considerando inicialmente dois vetores com duas posições, cada vetor poderia ser representado em um plano cartesiano. O primeiro vetor seria representado pelo ponto A(Xa, Ya) e o segundo vetor pelo ponto B(Xb, Yb). Também seria possível traçar um triângulo retângulo entre os pontos A e B, conforme mostrado na Figura 10. Nesse triângulo, um cateto seria dado pela diferença Xb-Xa e o outro cateto pela diferença Yb-Ya. Sendo assim, a distância euclidiana entre A e B seria a hipotenusa no triângulo que pode ser diretamente calculada pelo teorema de Pitágoras:

$$d(A,B) = \sqrt{(Xb-Xa)^2 + (Yb-Ya)^2} \qquad (8)$$

Generalizando, para encontrar a distância euclidiana entre dois vetores n-dimensionais, basta usar a seguinte fórmula, que calcula os quadrados das distâncias ao longo de cada dimensão d:

$$D(A,B) = \sqrt{(d_{1,A}-d_{1,B})^2 + (d_{2,A}-d_{2,B})^2 + \ldots + (d_{n,A}-d_{n,B})^2} \qquad (9)$$

Figura 10 – Distância euclidiana.



Fonte: Elaborada pelo autor

2.4.3 Métricas Utilizadas

As métricas apresentadas são: acurácia, precisão, revocação e pontuação f1. A acurácia nada mais é do que o número de acertos dividido pelo número total de elementos do conjunto de teste. A precisão representa a proporção de verdadeiros positivos e falsos positivos, sendo que um valor alto de precisão pressupõe um valor baixo de falsos positivos, conforme podemos inferir da Equação 10:

$$\text{precisão} = TP/(TP+FP) \qquad (10)$$

A revocação, por sua vez, indica a proporção entre verdadeiros positivos e falsos negativos, dessa forma, uma revocação alta implica em um baixo valor de falsos negativos, como mostra a Equação 11:

$$\text{revocação} = TP/(TP+FN) \qquad (11)$$

Por outro lado, a pontuação f1 é uma média harmônica das duas métricas anteriores que só será alta se tanto a precisão como a revocação forem altas. Essa última métrica é calculada pela seguinte fórmula, na Equação 12:

$$F1 = 2/(1/\text{precisão}+1/\text{revocação}) \qquad (12)$$

3 TRABALHOS RELACIONADOS

3.1 Monitoramento de Estudantes

Samhitha e Gaffar (2019) propõem um sistema baseado em IoT (*Internet of Things*) para monitorar o uso de um ônibus escolar. Esse sistema utiliza RFID (*Radio-Frequency Identification*) para identificar os estudantes. Emprega, também, um microcontrolador para processar os sinais e um celular para transmitir as informações via SMS (Serviço de Mensagens Curtas, do inglês *Short Message Service*).

Teja *et al.* (2019) usa um Raspberry Pi, um módulo GPS, um módulo GSM e um leitor de RFID para monitorar o trajeto de crianças. Os leitores são posicionados nos lugares por onde a criança deve passar e uma notificação é mandada para os responsáveis com a posição atual da criança.

Chandrala *et al.* (2019) apresenta um cinto com um microcontrolador preso à criança. Ligado ao microcontrolador, encontra-se um botão de pânico e um transmissor de radiofrequência. Quando a criança entra na escola, um rádio receptor recebe a informação e um SMS é mandado para os responsáveis com a localização da criança. A criança também pode acionar um botão de pânico. Nesse caso, o microcontrolador envia uma mensagem para os responsáveis por meio de um celular.

Pawde *et al.* (2020) utiliza um sistema de reconhecimento facial para registrar a presença dos estudantes. Esse sistema emprega uma rede neural artificial em conjunto com outras técnicas tais como: HOG (*Histograms of Oriented Gradients*) e SVM (*Support Vector Machine*). A acurácia obtida foi de 93%.

Por fim, Vidyasagar, Balaji e Reddy (2015) fazem uso de um sistema com RFID na entrada e na saída de um ônibus escolar. O sinal do RFID é enviado para um microcontrolador e retransmitido via módulo GSM. A direção da escola e os responsáveis recebem uma notificação da entrada e da saída dos estudantes do ônibus escolar.

3.2 Reconhecimento Facial

O sistema FaceNet foi um marco na história do reconhecimento facial, e ainda hoje permanece como uma grande referência nessa área. Schroff, Kalenichenko e Philbin (2015) expõem as principais características desse sistema. Segundo eles, o FaceNet é um sistema

unificado para verificação, reconhecimento e agrupamento facial. Nesse contexto, verificar significa examinar se duas imagens são da mesma pessoa. Reconhecer, por sua vez, é dizer quem é a pessoa que aparece em uma dada imagem. Agrupar, por seu turno, é encontrar pessoas iguais em um conjunto de imagens.

O FaceNet utiliza uma CNN para construir um vetor com os pontos mais significativos da face e, em seguida, usa a distância euclidiana para realizar as três tarefas já mencionadas. Dessa forma, a verificação torna-se a tarefa de examinar se uma determinada distância euclidiana está dentro de um limiar. O reconhecimento transforma-se em um processo de classificação, usando o algoritmo KNN. O agrupamento, por seu lado, passa a ser uma tarefa simples que pode ser feita com o auxílio do algoritmo K-means.

Os bancos de dados utilizados nos testes foram o LFW (*Labelled Faces in the Wild*) e o *YouTube Faces*. O primeiro é um banco de dados de imagens e o segundo é um banco de dados de vídeos. Os resultados foram os seguintes: com o LFW, sem alinhamento da face, 98,87% de acurácia. Com o LFW, com alinhamento da face, 99,63% de acurácia. Com o *YouTube Faces*, usando 100 *frames*, 95,12% de acurácia. Com o *YouTube Faces*, usando 1.000 *frames*, 95,18% de acurácia.

O projeto *HyperFace*, apresentado por Ranjan, Patel e Chellappa (2019), é um sistema baseado em uma CNN capaz de detectar faces humanas, localizar os principais pontos de referência do rosto, estimar a posição da cabeça da pessoa e diferenciar o gênero masculino do feminino. O *HyperFace* parte do princípio de que o aprendizado de uma CNN é hierarquizado, ou seja, as camadas mais rasas detectam características mais simples e podem ser melhor utilizadas para localizar pontos de referência e para estimar a posição de uma cabeça. As camadas mais profundas, por sua vez, apreendem características mais gerais e podem ser melhor utilizadas nas tarefas de detecção de faces e distinção de gêneros.

Sendo assim, os pesquisadores construíram uma CNN padrão e, em seguida, extraíram características de diferentes camadas dessa rede. Mas, para realizar o aprendizado multitarefa, as características extraídas de diferentes camadas foram fundidas em uma nova CNN. Essa nova rede, por último, foi treinada simultaneamente com diferentes funções de erro, uma para cada tipo de tarefa. Os autores chegaram a duas importantes conclusões: 1) A fusão de camadas intermediárias melhorou o desempenho de todas as tarefas; 2) Todas as tarefas se beneficiaram do uso da estrutura de aprendizado multitarefa.

O estudo apresentado por Deb, Nain e Jain (2018) analisa a influência da idade das pessoas e da passagem do tempo no processo de reconhecimento facial. Ele utilizou os

sistemas COTS-A e o FaceNet para realizar os reconhecimentos, juntamente com um banco de dados formado com imagens de crianças e adolescentes de dois a 18 anos.

Os melhores resultados foram obtidos com a combinação dos dois sistemas. Para um lapso de tempo de um ano entre o conjunto de imagens de treinamento e o conjunto de imagens de teste, foi atingida uma acurácia de 90,18% no reconhecimento. Os autores também indicaram uma acurácia de 80% no reconhecimento com um lapso de tempo de 2,5 anos entre o conjunto de imagens de treino e o conjunto de dados de teste.

Huang, Li, Loy e Tang (2020) estudam formas de melhorar o treinamento de classes desbalanceadas, e indicam duas estratégias tradicionais: reamostragem e aprendizagem sensível ao custo. A primeira consiste em subamostrar a classe majoritária ou superamostrar a classe minoritária, enquanto a segunda atribui um custo maior ao erro de classificação da classe minoritária. Os autores propõem um novo método que divide a base de dados em *clusters* no interior das classes e entre as classes, o que resulta em fronteiras de classes equilibradas. Experimentos foram feitos com reconhecimento facial e previsão de um atributo em uma face, e resultados semelhantes aos tradicionais foram encontrados.

3.3 Análise dos Artigos Relacionados

Os sistemas já elaborados para monitorar os trajetos dos estudantes estão, em sua maioria, dentro de um contexto de IoT, mas não utilizam inteligência artificial, o que limita esses sistemas a aplicações mais tradicionais. Ademais, na literatura disponível, esses sistemas aparecem como aplicações isoladas, desprovidas de uma arquitetura mais abrangente, capaz de abarcar diferentes aplicações em um contexto de cidades inteligentes.

O sistema aqui proposto é uma arquitetura completa e inova, em relação a Samhitha e Gaffar (2019), Teja *et al.* (2019) e Vidyasagar, Balaji e Reddy (2015), pois substitui o RFID por um sistema de reconhecimento facial baseado em inteligência artificial, que é bem menos vulnerável à fraude. Também possui maior praticidade que Samhitha e Gaffar (2019) e Chandrala *et al.* (2019), porque não precisa de um aparelho celular para enviar as notificações. Por último, supera Pawde *et al.* (2020), uma vez que apresenta maior acurácia, ou seja, 98,1% contra 93%.

O presente trabalho também apresenta uma acurácia muito próxima a do sistema criado por Schroff, Kalenichenko e Philbin (2015) com a vantagem de ser um sistema

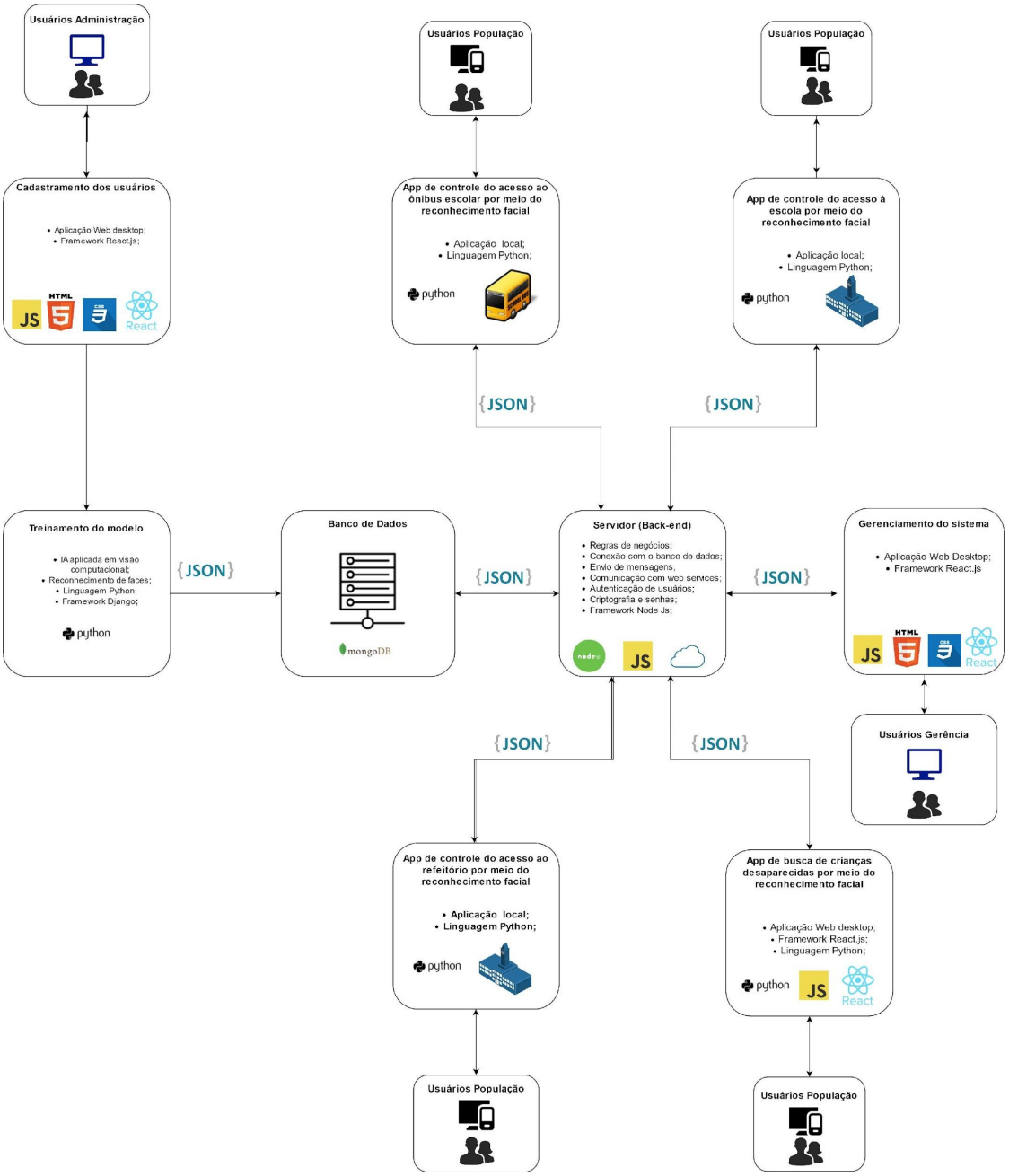
completo (com aplicação local, web e móvel) no contexto de uma cidade inteligente e não apenas um sistema de reconhecimento facial isolado.

O sistema proposto também utiliza uma rede mais simples e eficiente do que a rede apresentada por Ranjan, Patel e Chellappa (2019). Optou-se também por usar a técnica da reamostragem nos testes do sistema, que é mais simples e apresenta resultados semelhantes à técnica proposta por Huang, Li, Loy e Tang (2020). Por fim, o trabalho apresentado por Deb, Nain e Jain (2018) alerta para a necessidade de atualização anual das imagens usadas para o treinamento do sistema, o que será feito por ocasião das matrículas dos estudantes.

4 VISÃO GERAL DA ARQUITETURA

A Figura 11 mostra a arquitetura completa do sistema de reconhecimento facial. Nela, é possível ver os atores envolvidos, as principais aplicações e as tecnologias utilizadas.

Figura 11 – Arquitetura do sistema de reconhecimento facial.



Fonte: Elaborada pelo autor

Da esquerda para a direita, aparece primeiramente o “Cadastramento dos Usuários”, que se trata de uma aplicação web desenvolvida com o *framework* React.js que será usada

pelos administradores do sistema para cadastrar os alunos das escolas públicas. Neste cadastro, serão inseridos os dados pessoais de cada aluno e algumas fotos, capturadas no mesmo momento por uma webcam. Seguindo logo abaixo, surge o "Treinamento do modelo", que é um microsserviço desenvolvido com a linguagem Python. Esse microsserviço tem como entrada as fotos dos alunos, e faz uso de uma série de algoritmos de inteligência artificial para criar um modelo capaz de reconhecer as faces dos alunos cadastrados. Na sequência, o resultado do treinamento do modelo é armazenado no MongoDB, que é um banco de dados não relacional.

No núcleo da arquitetura encontra-se o Servidor *Back-end*, que realiza as principais funções que tornam possível o funcionamento articulado das várias partes do sistema. Dentre essas funções, destacam-se: o controle do acesso ao banco de dados, o envio de notificações aos usuários e a comunicação com os serviços web. O Servidor de *Back-end* usa a linguagem JavaScript e o *framework* Node.js.

Ao lado do Servidor está o "Gerenciamento do Sistema" que é uma aplicação web também implementada com o *framework* React.js. O "Gerenciamento do Sistema" supervisiona todo o sistema e, entre outras coisas, cria os perfis dos administradores do sistema.

Na parte superior central da Figura 11, pode-se observar a primeira aplicação: "App de controle de acesso ao ônibus escolar por meio de reconhecimento facial". Essa aplicação será a primeira a ser implantada, e tem como objetivo principal impedir que pessoas que não são estudantes regularmente matriculados utilizem o ônibus escolar que, por lei, é destinado apenas aos alunos da rede pública municipal. O funcionamento desta aplicação será detalhado em um tópico posterior.

Na parte superior direita, encontra-se a aplicação "App de controle de acesso à escola por meio de reconhecimento facial". Essa aplicação usa a linguagem Python e roda em um dispositivo local. Ela possui diversas semelhanças de hardware e software com a primeira aplicação. A principal diferença está na interação com outros atores, tais como a secretaria municipal de educação, o conselho tutelar e o ministério público.

A secretaria de educação receberá um relatório com a frequência dos alunos. O conselho, mediante um cadastro prévio, obterá um relatório com as faltas dos alunos que recebem auxílio do governo e, por essa razão, não podem deixar de ir à escola. Por fim, o ministério público, também mediante um cadastro prévio, receberá um relatório com as faltas dos alunos que cumprem medidas socioeducativas e que, por isso, são obrigados a frequentar

a escola. Dessa forma, a referida aplicação não será apenas um controle da entrada dos alunos, será também um instrumento de combate à evasão escolar.

Na parte inferior esquerda pode ser visto o “App de controle do acesso ao refeitório por meio do reconhecimento facial”. Essa também é uma aplicação desenvolvida em Python que roda em um dispositivo local. Além de controlar o acesso ao refeitório, essa aplicação, por meio do Servidor *Back-end*, notifica os responsáveis quando um aluno entra no refeitório para comer a merenda escolar, e também pode ser usada pela secretária de educação para fazer o controle do estoque de merenda.

Finalmente, na parte inferior direita aparece o “App de busca de crianças desaparecidas por meio do reconhecimento facial”. Essa é uma aplicação Web programada em Python que faz uso do modelo treinado com as imagens dos alunos das escolas públicas, e por meio das câmeras de segurança instaladas em locais públicos da Cidade Inteligente, pode buscar reconhecer crianças desaparecidas.

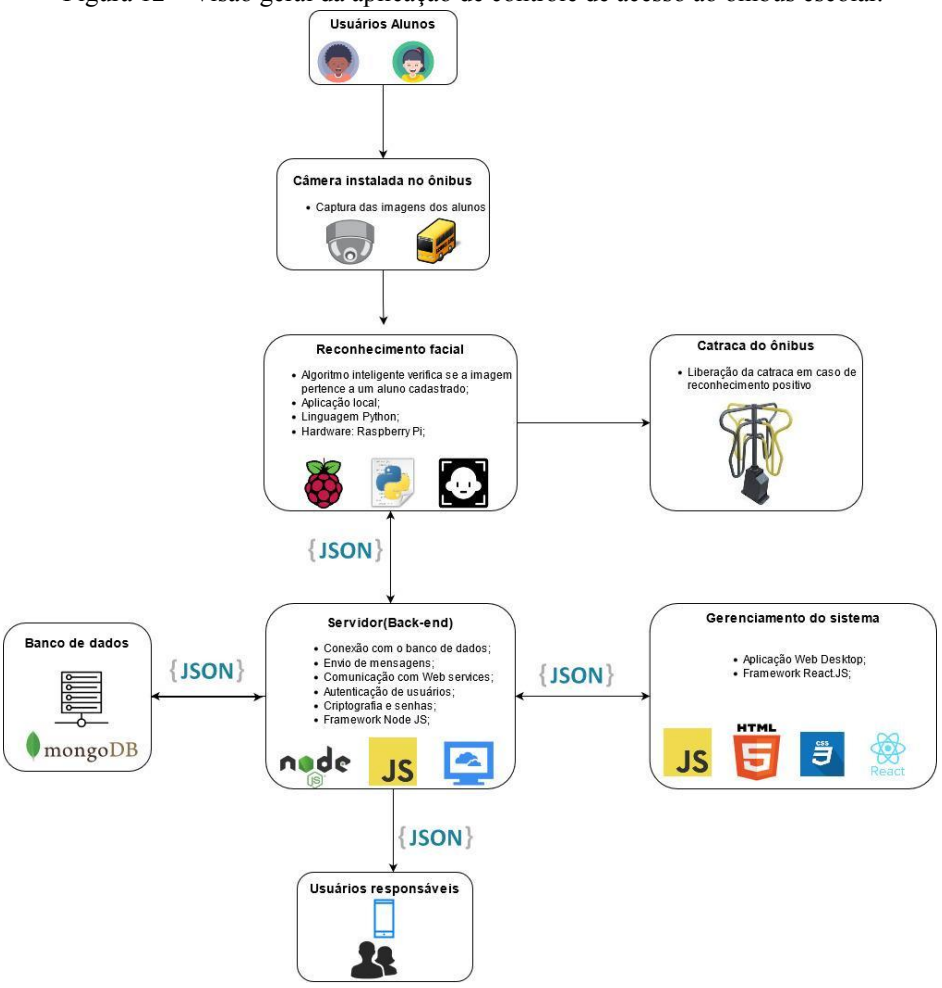
5 PRIMEIRA APLICAÇÃO

O “App de controle de acesso ao ônibus escolar por meio de reconhecimento facial” tem como primeiro objetivo controlar o acesso ao ônibus escolar no município paraense de Canaã dos Carajás. Com essa finalidade, utiliza um moderno sistema de reconhecimento facial que rapidamente verifica se a pessoa em questão é um aluno previamente cadastrado e, em caso positivo, libera a catraca permitindo o embarque. O sistema conta também com um serviço baseado na internet que envia uma mensagem informando aos responsáveis a hora que o aluno entrou no ônibus.

5.1 Visão Geral do Funcionamento da Primeira Aplicação

A Figura 12 mostra todas as etapas do sistema de controle de acesso ao ônibus escolar, desde a entrada no veículo até o envio da notificação.

Figura 12 – Visão geral da aplicação de controle de acesso ao ônibus escolar.



Fonte: Elaborada pelo autor

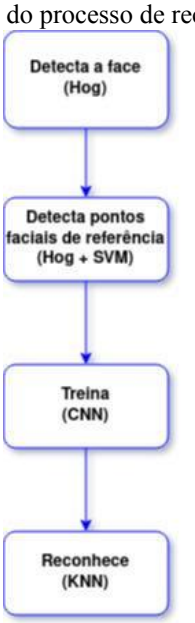
Inicialmente, uma pessoa tenta acessar o ônibus. Nesse momento, uma câmera estrategicamente posicionada dentro do ônibus captura a imagem do rosto dessa pessoa e, em seguida, esta imagem capturada é processada por um algoritmo inteligente escrito na linguagem Python e embarcado em um Raspberry Pi 4 instalado no ônibus. O algoritmo capaz de reconhecer faces humanas verifica se o rosto pertence a um aluno previamente cadastrado, e se o reconhecimento for positivo, a catraca é liberada para a entrada do aluno.

Ao mesmo tempo, o dispositivo inteligente manda a informação da entrada do aluno no ônibus, contendo data e hora, para um servidor na Internet. Posteriormente, o serviço da internet manda a informação da entrada do aluno para o responsável, que também foi previamente cadastrado, o qual recebe a notificação diretamente no aplicativo desenvolvido para smartphones. Por fim, a Figura 12 ilustra a aplicação para a Internet que gerencia o sistema de reconhecimento facial, e tem como principal função cadastrar os alunos e seus respectivos responsáveis.

5.2 Fluxograma do Processo de Reconhecimento Facial

A Figura 13 mostra as quatro principais etapas do processo de reconhecimento facial.

Figura 13 – Fluxo do processo de reconhecimento.

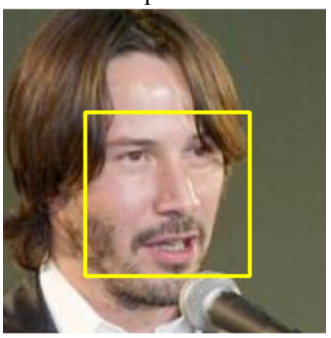


Fonte: Elaborada pelo autor

Na primeira etapa do processo, o algoritmo HOG é utilizado para buscar, na imagem de entrada, um padrão correspondente a uma face humana. Caso esse padrão seja encontrado, um retângulo é desenhado para delimitar o espaço onde a face está.

A Figura 14 mostra um exemplo de saída da primeira etapa, com a face delimitada. A imagem pertence ao dataset público LFW.

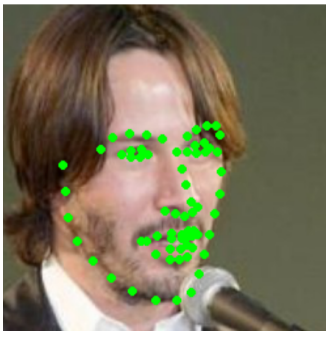
Figura 14 – Exemplo de face detectada.



Fonte: Elaborada pelo autor

Na segunda etapa, o algoritmo HOG é usado em conjunto com outra técnica, chamada de SVM. O objetivo dessa etapa é detectar, no espaço anteriormente delimitado da face, 68 pontos de referência. A Figura 15 mostra um exemplo da saída da segunda etapa, com os pontos de referência marcados em verde.

Figura 15 – Exemplo de pontos faciais de referência.



Fonte: Elaborada pelo autor

Na terceira etapa, os 68 pontos faciais de referência são usados como entrada de uma rede neural artificial treinada previamente, que se trata de uma CNN do tipo ResNet. Essa rede neural gera em sua saída um descritor facial, que é um vetor numérico de 128 posições que descreve geometricamente as principais características da face. Essa etapa também salva um arquivo com os descritores de cada imagem, e um outro arquivo com os respectivos

rótulos das imagens. A Figura 16 mostra um exemplo do arquivo de descritores, que possui a estrutura de uma matriz.

Figura 16 – Descritores faciais.

"recursos/descritores_lfw.npy"

<class 'numpy.ndarray'>

[[0.0231774 -0.03540148 0.00637923 ... 0.08052176 0.00873458
-0.00103766]
[-0.03958605 0.00413402 0.03178674 ... 0.00810013 0.04620028
0.01335262]
[-0.03044262 0.00194865 0.07012942 ... -0.07528858 0.0729827
0.00713028]
...
[-0.09771828 0.0051564 0.1593315 ... -0.00383048 0.04726707
0.01650582]
[-0.09172633 0.02785334 0.14442033 ... 0.00488575 0.0184319
0.01051828]
[-0.1049312 0.05075648 0.14799951 ... -0.0034761 0.06202971
-0.00356548]]

Fonte: Elaborada pelo autor

A Figura 17 exemplifica um arquivo de rótulos com a estrutura de um dicionário da linguagem Python.

Figura 17 – Rótulos das imagens.

"recursos/indices_lfw.pickle"

<class 'dict'>

{0: '1LFW/Treinamento\\Abdullah_Gul_0001.jpg', 1: '1LFW/Treinamento\\
Abdullah_Gul_0002.jpg', 2: '1LFW/Treinamento\\Abdullah_Gul_0003.jpg', 3:
'1LFW/Treinamento\\Abdullah_Gul_0005.jpg', 4: '1LFW/Treinamento\\
Abdullah_Gul_0007.jpg', 5: '1LFW/Treinamento\\Abdullah_Gul_0012.jpg', 6:
'1LFW/Treinamento\\Abdullah_Gul_0013.jpg', 7: '1LFW/Treinamento\\
Abdullah_Gul_0014.jpg', 8: '1LFW/Treinamento\\Abdullah_Gul_0015.jpg', 9:
'1LFW/Treinamento\\Abdullah_Gul_0017.jpg', 10: '1LFW/Treinamento\\
Adrien_Brody_0001.jpg', 11: '1LFW/Treinamento\\Adrien_Brody_0005.jpg', 12:
'1LFW/Treinamento\\Adrien_Brody_0007.jpg', 13: '1LFW/Treinamento\\
Adrien_Brody_0008.jpg', 14: '1LFW/Treinamento\\Adrien_Brody_0009.jpg', 15:
'1LFW/Treinamento\\Alejandro_Toledo_0001.jpg', 16: '1LFW/Treinamento\\
Alejandro_Toledo_0002.jpg', 17: '1LFW/Treinamento\\Alejandro_Toledo_0004.jpg',
18: '1LFW/Treinamento\\Alejandro_Toledo_0005.jpg', 19: '1LFW/Treinamento\\
Alejandro_Toledo_0006.jpg', 20: '1LFW/Treinamento\\
Alejandro_Toledo_0009.jpg'...}

Fonte: Elaborada pelo autor

Na quarta etapa, é utilizado o algoritmo KNN, que funciona da seguinte maneira: Inicialmente, é gerado um descritor facial para a imagem de teste. Em seguida, é calculada a distância euclidiana entre o descritor da imagem de teste e cada um dos descritores das imagens de treinamento. Por fim, é selecionada a menor distância, e se essa distância estiver dentro de um limiar previamente estabelecido, a imagem de teste recebe o mesmo rótulo da imagem de treinamento correspondente. Em sucessivos experimentos, um limiar de 0,5 mostrou-se suficiente para determinar com precisão a identidade da grande maioria das

imagens usadas nos testes. A Figura 18 mostra um exemplo da saída da quarta etapa. O número na imagem representa a distância euclidiana.

Figura 18 – Exemplo de face reconhecida.

Fonte: Elaborada pelo autor

6 RESULTADOS DOS TESTES

A base de dados LFW foi utilizada no primeiro teste do sistema de reconhecimento facial. Essa base foi escolhida por ser muito usada em testes e *benchmarks*, e por ser citada frequentemente em publicações na área de reconhecimento facial. Trata-se de uma base de dados pública criada pela University of Massachusetts a partir de imagens de pessoas famosas tiradas da internet. A base é constituída por imagens de políticos, esportistas, artistas e empresários de diferentes partes do mundo, homens e mulheres de diferentes grupos étnicos. Na maioria das imagens, as pessoas não estão posando para as fotos, mas sim em suas atividades habituais. Ao todo, são 13.233 imagens de 5.749 pessoas, entretanto, a maioria das pessoas possui apenas uma ou duas imagens.

Inicialmente, foi separado um subconjunto com as pessoas que possuem sete ou mais imagens, o que totaliza 158 pessoas. Em seguida, esse subconjunto foi dividido em duas pastas, uma para o treinamento e a outra para o teste. A pasta de treinamento possui entre cinco a dez imagens de cada pessoa. A pasta de teste, por outro lado, possui duas outras imagens de cada uma das 158 pessoas. Todas as imagens foram rotuladas com o nome da pessoa, o sobrenome e um número sequencial.

O resultado obtido neste teste foi o seguinte: 310 acertos de um total de 316 imagens testadas, o que significa uma acurácia de 98,10%. Esse resultado é bem próximo do resultado obtido pelo sistema FaceNet desenvolvido pelos pesquisadores do Google (Schroff; Kalenichenko; Philbin, 2015), levando em consideração que, para reduzir o custo computacional, aqui não foi utilizado o alinhamento da imagem.

A Figura 19 apresenta as principais métricas relativas ao primeiro teste. Para facilitar a visualização, são mostradas apenas as 16 primeiras classes e as médias finais. A média ponderada da precisão (do inglês, *precision*) para todas as classes é de 99%. A média ponderada da revocação (*recall*) para todas as classes é de 98%. Por fim, a média ponderada da pontuação f1 (*f1-score*) para todas as classes é de 98%.

Os pesos da média ponderada correspondem ao número de ocorrências de cada classe no conjunto dos valores esperados. O número de ocorrências, por sua vez, é chamado de suporte. Deve-se ressaltar que existem dois casos de baixa precisão e um caso de baixa revocação que são devidos ao baixo número de amostras, ou seja, como foram utilizadas apenas duas imagens de teste para cada pessoa, um único erro altera bastante a métrica relativa a uma única pessoa, mas sem comprometer a média geral.

Figura 19 – Resumo das principais métricas de classificação.

	precision	recall	f1-score	support
Abdullah Gul	1.00	1.00	1.00	2
Adrien Brody	1.00	1.00	1.00	2
Alejandro Toledo	1.00	1.00	1.00	2
Alvaro Uribe	1.00	1.00	1.00	2
Amelie Mauresmo	1.00	1.00	1.00	2
Andre Agassi	0.67	1.00	0.80	2
Andy Roddick	1.00	0.50	0.67	2
Angelina Jolie	1.00	1.00	1.00	2
Ann Veneman	1.00	1.00	1.00	2
Anna Kournikova	1.00	1.00	1.00	2
Ari Fleischer	1.00	1.00	1.00	2
Ariel Sharon	1.00	1.00	1.00	2
Arnold Schwarzenegger	0.67	1.00	0.80	2
Atal Bihari	1.00	1.00	1.00	2
Bill Clinton	1.00	1.00	1.00	2
Bill Gates	1.00	1.00	1.00	2
accuracy			0.98	316
macro avg	0.98	0.97	0.97	316
weighted avg	0.99	0.98	0.98	316

Fonte: Elaborada pelo autor

Nas Figuras 20 e 21 aparecem alguns exemplos de pessoas que foram reconhecidas pelo sistema proposto. Na Figura 20, é possível perceber que o sistema é capaz de reconhecer faces em diferentes ângulos e com variadas expressões.

Figura 20 – Exemplos de pessoas reconhecidas.

Fonte: Elaborada pelo autor

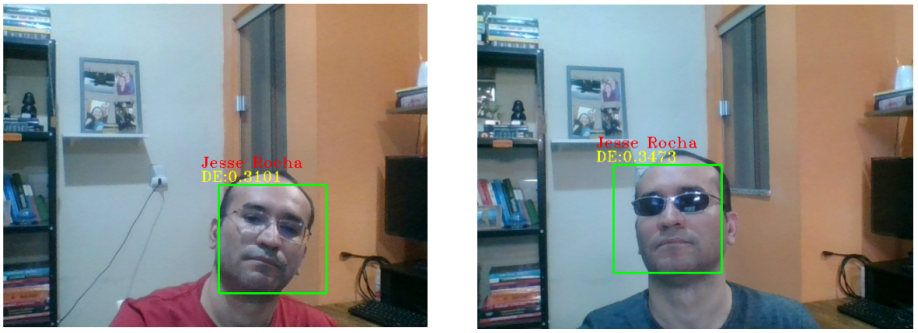
Na Figura 21, é possível notar que o sistema reconhece pessoas de diferentes grupos étnicos independentemente do uso ou não de protetor solar ou adereços de cabeça.

Figura 21 – Outros exemplos de pessoas reconhecidas.

Fonte: Elaborada pelo autor

Em um segundo teste, o sistema foi treinado com fotos de arquivo pessoal e, em seguida, testado com o uso de uma webcam em um laptop Acer Nitro 5 com processador Intel Core i7 e 8 GB (*Gigabytes*) de memória RAM (*Random Access Memory*), o mesmo usado no primeiro teste. Em ambos os testes cada reconhecimento ocorreu em uma fração de segundo, mostrando que o sistema é eficiente e eficaz. O resultado do segundo teste aparece na Figura 22.

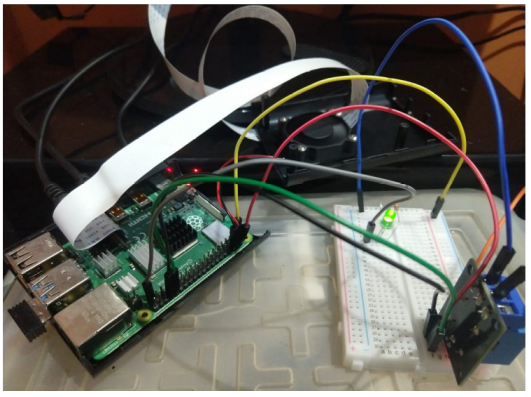
Figura 22 – Resultado do segundo teste.



Fonte: Elaborada pelo autor

Por fim, o sistema de reconhecimento facial foi testado em um Raspberry Pi 4 com processador ARM Cortex-A72, 4 GB de memória RAM e um módulo de câmera desenvolvido para o Raspberry Pi. Nesse teste, o reconhecimento foi feito com a ajuda do módulo de câmera e usado para acionar um módulo relé que, no produto final, irá acionar a trava elétrica da catraca do ônibus escolar. Também nesse teste, o reconhecimento foi realizado em uma fração de segundo e o acionamento do módulo relé ocorreu de forma correta. A Figura 23 mostra o Raspberry Pi com o módulo relé em funcionamento durante o último teste.

Figura 23 – Raspberry Pi e módulo relé.



Fonte: Elaborada pelo autor

7 ESTÁGIO ATUAL DE DESENVOLVIMENTO DA PRIMEIRA APLICAÇÃO

O “App de controle de acesso ao ônibus escolar por meio de reconhecimento facial” já foi apresentado e aprovado em um encontro com a prefeitura de Canaã dos Carajás. A Figura 24 mostra o hardware do referido sistema, montado em um tripé para melhor visualização dos interessados. O próximo passo será a instalação desse sistema em um ônibus escolar da rede municipal de Canaã.

Figura 24 – Hardware do sistema de reconhecimento facial.

Fonte: Elaborada pelo autor

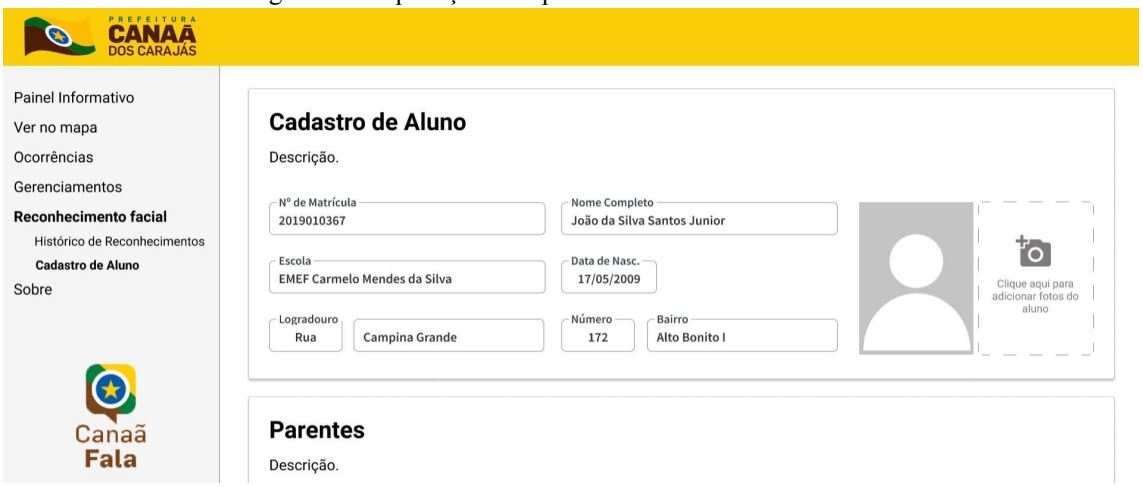
A aplicação web para o acompanhamento dos alunos que entram no ônibus escolar está pronta, e aparece na Figura 25. Na parte superior da imagem, é possível ver a identificação da última pessoa reconhecida, e na parte inferior, encontra-se uma lista com as identificações mais recentes.

Figura 25 – Aplicação web para o acompanhamento dos alunos.

Fonte: Elaborada pelo autor

A aplicação web para o cadastramento dos alunos também já foi desenvolvida. Sua interface gráfica aparece na Figura 26. Nessa aplicação, é possível cadastrar os alunos e seus responsáveis e salvar as imagens dos alunos, necessárias para o treinamento do sistema de reconhecimento facial.

Figura 26 – Aplicação web para o cadastramento dos alunos.



Fonte: Elaborada pelo autor

Por fim, na Figura 27, pode ser visto o protótipo da interface da aplicação móvel, na qual os responsáveis irão fazer o acompanhamento dos alunos. Essa aplicação terá funcionalidades parecidas com a aplicação web de acompanhamento dos alunos.

Figura 27 – Aplicação móvel para o acompanhamento dos alunos.

Fonte: Elaborada pelo autor

7.1 Especificações dos Equipamentos Utilizados na Primeira Aplicação

Conforme citado anteriormente, a primeira aplicação foi testada com um Raspberry Pi 4 com processador ARM Cortex-A72 e 4 GB de memória RAM. Um Raspberry 4 com 2 GB de RAM também serviria a esse propósito, apenas deixando o sistema um pouco mais lento. Existem vários outros modelos de SBC (*Single-Board Computer*), isto é, computador de placa única, similares ao Raspberry Pi 4 no mercado que poderiam ser testados. Alguns apresentam preços menores do que o Raspberry e outros preços maiores.

O sistema de reconhecimento facial apresentou um bom desempenho com o uso de uma câmera especialmente desenvolvida para o Raspberry Pi que utiliza um cabo flexível de 15 pinos e tem uma resolução de 5 megapixels. O sistema também foi testado com uma webcam comum de 3 megapixels com saída USB (*Universal Serial Bus*, em português, porta serial universal). Nesse segundo caso, os resultados foram bem semelhantes ao primeiro, o que mostra que usar uma câmera desenvolvida para o Raspberry ou uma webcam comum é uma escolha de projeto sem grandes impactos no resultado final. Vale lembrar que câmeras com resoluções menores do que 3 megapixels não foram testadas.

Um cartão de memória classe dez de 64 GB foi utilizado para gravar o sistema operacional do Raspberry (Raspberry OS), a aplicação e os arquivos com os descritores faciais e os rótulos para testes. Como apenas 25% da capacidade total do cartão foi usada, pode-se afirmar que um cartão de 32 GB também poderia ter sido usado, o que reduziria os custos.

8 CONCLUSÃO

O sistema de reconhecimento aqui proposto atingiu, nos testes realizados, excelentes resultados em todas as métricas. Ele também mostrou-se capaz de reconhecer pessoas com diferentes expressões faciais e em diferentes ângulos. Os reconhecimentos ocorreram em frações de segundo, o que demonstra que o referido sistema pode ser incorporado à rotina escolar sem provocar atrasos no embarque nos ônibus ou na entrada da escola. O hardware utilizado possui pequenas dimensões e baixo consumo de energia, podendo ser instalado em praticamente qualquer lugar. Ele já dispõe de conexão Wi-Fi e Bluetooth e também pode receber um módulo com GPS e GSM possibilitando o rastreamento dos ônibus escolares em tempo real.

O sistema de monitoramento conta com aplicações Web e aplicações para dispositivos móveis permitindo que tanto os responsáveis como a direção da escola monitorem os trajetos dos estudantes de casa para a escola e da escola para casa. Essas informações também podem ser automaticamente repassadas para outras autoridades competentes como o Ministério Público e o Conselho Tutelar. Dessa forma, o sistema pode ser utilizado como uma ferramenta de combate à evasão escolar e ao desaparecimento de crianças e adolescentes.

Os principais resultados deste trabalho foram publicados na forma de artigo na *International Conference on Smart Applications, Communications and Networking (SmartNets 2023)* com o título "*A Platform for Monitoring Student Commuting in the Use of School Transport in Smart Cities – A Facial Recognition Based Approach*". Esse artigo encontra-se em anexo.

Por fim, a primeira aplicação, isto é, a aplicação de controle de acesso ao ônibus escolar por meio de reconhecimento facial está em consonância com todos os requisitos apresentados pelos Stakeholders e encontra-se no Nível de Maturidade Tecnológica TRL 6, ou seja, já foi feita a demonstração de funções críticas do protótipo em ambiente relevante. O próximo passo será a instalação do protótipo do sistema em oito ônibus escolares da rede municipal de Canaã dos Carajás que servirão como projeto piloto. Para essa nova fase o sistema já conta com o aval do conselho de ética em pesquisa da Universidade Federal do Pará (UFPA) para garantir que todos os direitos dos participantes sejam respeitados.

REFERÊNCIAS

BRAGA, Antônio de Pádua; LUDERMIR, Teresa Bernarda; CARVALHO, André Carlos Ponce de Leon Ferreira. **Redes Neurais Artificiais:** Teoria e Aplicação. Rio de Janeiro: LTC, 2016.

BRANCO, Emerson Pereira; ADRIANO, Gisele; BRANCO, Alessandra Batista de Godoi; IWASSE, Lilian Fávaro Alegrâncio. Evasão Escolar: Desafios para Permanência dos Estudantes na Educação Básica. **Revista Contemporânea de Educação**, Rio de Janeiro, v. 15, n. 33, mai/ago. 2020.

CHANDRALA, Vibha; NIVEDITHA, N.; REDDY, Nega B.; URMILA, N.; DEEPAK, G. Child Monitoring System Using IoT. **International Journal of Advance Research, Ideas and Innovations in Technology**, [s.l.], v. 15, n. 3, p. 416-419, 2019.

CHOLLET, François. **Deep Learning with Python**. Shelter Island: Manning Publications, 2018.

DEB, Debayan; NAIN, Neeta; JAIN, Anil K. Longitudinal Study of Child Face Recognition. *In:* 2018 International Conference on Biometrics, 11., 2018, [s.l.]. **Anais** [...] [s.l.]: IAPR, 2018. p. 225-232.

FERREIRA, Letícia Carvalho de Mesquita. Faces da Desigualdade: Os Efeitos da Circulação de Retratos de Crianças Desaparecidas no Brasil. **International Journal on Collective Identity Research**, Lejona, v. 2017/1, p. 1-25, 2017.

SILVA FILHO, Raimundo Barbosa; ARAÚJO, Ronaldo Marcos. Evasão e Abandono Escolar na Educação Básica no Brasil: Fatores, Causas e Possíveis Consequências. **Educação Por Escrito**, Porto Alegre, v. 8, n. 1, p. 35-48, 2017.

GÉRON, Aurélien. **Mãos à obra:** aprendizado de máquina com Scikit-Learn, Keras & TensorFlow: Conceitos, ferramentas e técnicas para a construção de sistemas inteligentes. Rio de Janeiro: Alta Books, 2021.

HUANG, Chen; LI, Yining; LOY, Chen Change; TANG, Xiaoou. Deep Imbalanced Learning for Face Recognition and Attribute Prediction. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 42, n. 11, p. 2781-2794, 2019.

OLIVEIRA, José Adelmo Menezes de; MAGRONE, Eduardo. Evasão Escolar: Apreensões e Compreensões em Contexto Adverso. **Revista Labor**, Fortaleza, v. 1, n. 26, p. 11-32, 2021.

PAWDE, Parikshit; BHARGAVA, Anjali; DUBEY, Anuradha; HABLANI, Sheetal; JAIN, Sweta. Real Time Attendance System Using One Shot Learning for Face Recognition. **International Journal of Advanced Trends in Computer Science and Engineering**, Tirupur, v. 9, n. 2, p. 1986-1990, mar. 2020.

PROVOST, Foster; FAWCETT, Tom. **Data Science para Negócios**. Rio de Janeiro: Alta Books, 2016.

RANJAN, Rajeev; PATEL, Vishal; CHELLAPPA, Rama. Hyperface: A Deep Multi-task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 41, n. 1, p. 121-135, 2019.

ROLIM, Gisleila da Silva; TARDIVO, Leila Salomão de La Plata Cury; RADZEVICIUS, Letícia da Costa; SALDANHA, Mariana Fernandes; SALLES, Rodrigo Jorge. Análise do Luto de Mães de Crianças e Adolescentes Desaparecidos. **Psicologia: Ciência e Profissão**, Brasília, v. 38, n. 3, p. 507-521, jul/set. 2018.

ROSA, João Luís Garcia. **Fundamentos da Inteligência Artificial**. Rio de Janeiro: LTC, 2011.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**. Rio de Janeiro: Elsevier, 2013.

SAMHITHA, K. Sree Veda; GAFFAR, Abdul. Acute School Bus Surveillance and Notification System. **International Journal of Advance Research, Ideas and Innovations in Technology**, Vellore, v. 5, n. 2, p. 1672-1676, abr. 2019.

SCHROFF, Florian; KALENICHENKO, Dmitry; PHILBIN, James. Facenet: a Unified Embedding for Face Recognition and Clustering. IEEE xplore, p. 815-823, 2015.

TEJA, Appadi Sumanth; REDDY, Devaram Gnanendra; KUMAR, Dhinesh; JEYARAMYA, V. Child in Safety with Localization Awareness. **International Journal of Advance Research, Ideas and Innovations in Technology**, Vellore, v. 5, n. 2, p. 315-317, 2019.

VIDYASAGAR, K.; BALAJI, G.; REDDY, Narendra. RFID-GSM Imparted School Children Security System. **Communications on Applied Electronics**, Nova Iorque, v. 2, n. 2, p. 17-21, jun. 2015.

ANEXO – A Platform for Monitoring Student Commuting in the Use of School Transport in Smart Cities - A Facial Recognition Based Approach

A Platform for Monitoring Student Commuting in the Use of School Transport in Smart Cities - A Facial Recognition Based Approach

Jessé da Costa Rocha
Graduate Program in Electrical Engineering, Institute of Technology, Federal University of Pará (UFPA)
Belém, Pará, Brazil
jesse.rocha@icen.ufpa.br

Marcela Alves de Souza
Information and Communication Technology Center
Federal University of the South and Southeast of Para (Unifesspa),
Marabá, Pará, Brazil
marcela.alves@unifesspa.edu.br

Evelin Helena Silva Cardoso
Department of Computer Science
Federal Rural University of the Amazon (UFRA)
Capitão Poço, Pará, Brazil
evelin.cardoso@ufra.edu.br

Nandamudi Vijaykumar
Coordination of Applied Research and Technological Development
National Institute for Space Research (INPE)
São José dos Campos, São Paulo, Brazil
vijay.nl@inpe.br

Jasmine Priscyla Leite de Araújo
Graduate Program in Electrical Engineering, Institute of Technology, Federal University of Pará (UFPA)
Belém, Pará, Brazil
jasmine@ufpa.br

Renato Lisboa Francês
Graduate Program in Electrical Engineering, Institute of Technology, Federal University of Pará (UFPA)
Belém, Pará, Brazil
rfrances@ufpa.br

***Abstract*—This article proposes an intelligent platform for monitoring students' steps on their way to school until they leave the school to their homes. This platform can identify students and notify those responsible and competent authorities in various situations of school life, such as: entering and leaving the school bus, entering and leaving school, entering the school cafeteria, etc. The first application aims to control access to the school bus through facial recognition. In the tests carried out, the recognition system achieved excellent results in all metrics.**

Keywords—intelligent platform, facial recognition, student monitoring

I. Introduction

Since its emergence in the mid-twentieth century, the modern computer has far surpassed the human ability to perform calculations, being used efficiently as early as World War II to calculate missile trajectories and to decipher secret codes. However, a capacity that develops spontaneously in humans remained inaccessible to computers for decades: the ability to see, recognize and differentiate between objects, animals, and people. However, in recent years, the so-called computer vision has developed considerably, thanks in large part to the emergence of an artificial intelligence technique called CNN (Convolutional Neural Network).

The great milestone of this development process was the 2012 ImageNet challenge [1]. This challenge consisted of classifying a set of colored images in 1000 different categories based on training conducted with more than 1 million images. The team of Alex Krizhevsky and Geoffrey Hinton won an ImageNet Challenge 2012 achieving an accuracy of 83.6%. In the following years, the ImageNet challenge was dominated by CNNs that reached an accuracy of 96.4% in 2015.

Also in 2015, a group of Google researchers introduced FaceNet to the world, a facial recognition system that had a CNN as its core. According to the project developers, the system achieved a maximum accuracy of 99.63% using the LFW (Labeled Faces in the Wild) database with face alignment [2]. This result is considered superior to the human ability to recognize people. Since then, many companies have started offering this service, using similar technology, which has made facial recognition something present in people's daily lives.

However, this article goes a step further and presents a complete architecture of a facial recognition system in a smart city context. This architecture includes a series of services applied to the area of education and public safety using low-cost equipment and a modern web development paradigm.

Furthermore, a smart city is essentially a connected city, in which any device can connect to the network anytime, anywhere. In addition, the mobile ecosystem is currently heterogeneous and comprises a multitude of networks with different technologies, such as 5G, LTE, Wi-Fi, Bluetooth, and WiMax. The heterogeneity of a wireless environment offers the opportunity to evaluate and selects the best network among several possible options, to best meet the requirements of the different smart applications incorporated in the day-to-day life of cities, such as the smart application presented in this article. Therefore, the issue addressed in [3] will also be considered in the implementation of the architecture described in this paper to guarantee the quality of the connection and at the same time optimize energy consumption.

The next topics are distributed as follows: Sections II briefly addresses the related works, III explains the proposed architecture, IV details the first application that will be implemented, V shows the test results of the first application, and VI presents the conclusions.

II. Related Work

The FaceNet system was a milestone in the history of facial recognition and even today remains a major reference in this area. [2] exposes the main characteristics of this system. According to that article, FaceNet is a unified system for facial verification, recognition, and clustering. In this context, verifying means examining whether two images are

979-8-3503-0252-3/23/$31.00 ©2023 IEEE

of the same person. To recognize, in turn, is to say who is the person that appears in a given image. Clustering, on the other hand, is to find similar people in a set of images. FaceNet uses a CNN to build a vector of the most significant points on the face and then uses the Euclidean distance to perform the three tasks already mentioned. In this way, verification becomes the task of examining whether a given Euclidean distance is within a threshold. The recognition turns into a classification process using the KNN (K Nearest Neighbor) algorithm. Clustering, on the other hand, becomes a simple task that can be done with the aid of the k-means algorithm. The databases used in the tests were LFW and YouTube Faces. The first is an image database and the second is a video database. The results were as follows: with LFW, without face alignment, 98.87 accuracy. With LFW, with face alignment, 99.63 accuracy. With YouTube Faces, using 100 frames, 95.12% accuracy. With YouTube Face, using 1000 frames, 95.18% accuracy.

The HyperFace project, presented in [4], is a system based on a CNN capable of detecting human faces, locating the main landmarks of the face, estimating the position of the person's head and differentiating between male and female genders. HyperFace assumes that the learning of a CNN is hierarchical, that is, the shallower layers detect simpler features and can be better used to locate reference points and to estimate the position of a head. The deeper layers, in turn, apprehend more general characteristics and can be better used in face detection and gender distinction tasks. Therefore, the researchers built a standard CNN and then extracted features from different layers of this network. But to realize multitasking learning, the extracted features from different layers were merged into a new CNN. This new network, finally, was trained simultaneously with different error functions, one for each type of task. The authors reached two important conclusions: 1) Merging intermediate layers improved the performance of all tasks. 2) All tasks benefited from using the multitasking learning framework.

[5] proposes an integrated IoT (Internet of Things) environment based on RFID (Radio-Frequency Identification) and sensor networks. They also propose an architecture to articulate the varied uses of RFID and sensor networks in different areas, such as: health, education, security, climate monitoring, etc. As an addition, they suggest a monitoring and processing center planned to use cloud computing. The proposed architecture consists of four layers, namely: 1st) Perception layer (devices, sensors, and RFID tags). 2nd) Data conversion layer (Middle ware). 3rd) Network layer (routers, gateways, and switches). 4th) Application layer (intelligent services and applications from different domains). The main challenges pointed out by the developers of this system are related to compatibility and security. They conclude that the said system could serve the general public and can be implemented in different regions.

The study presented in [6] analyzes the influence of people's age and the passage of time in the facial recognition process. It used the COTS-A and FaceNet systems to carry out the recognitions, along with a database consisting of images of children and adolescents from 2 to 18 years old. The best results were obtained with the combination of the two systems. For a one-year time lapse between the training image set and the test image set, a recognition accuracy of 90.18% was achieved. The authors also indicated an 80% recognition accuracy with a time lapse of 2.5 years between the training image set and the test data set.

There are studies [7] showing means to improve the training of unbalanced classes. It indicates two traditional strategies: resampling and cost-sensitive learning. The first consists of undersampling the majority class or oversampling the minority class. The second assigns a higher cost to the minority class misclassification. The authors propose a new method that divides the database into clusters within classes and between classes, which results in balanced class boundaries. Experiments were done with facial recognition and prediction of an attribute on a face. Results similar to the traditional ones were found.

The system proposed here presents an accuracy very close to the system created by [2] with the advantage of being a complete system (with local, web and mobile application) in the context of a smart city and not just an isolated facial recognition system. The proposed system also uses a simpler and more efficient network than the one presented in [4]. In relation to the work published in [5], the present work innovates by replacing RFID monitoring with facial recognition which, in this context, proves to be safer and more reliable. It was also decided to use the resampling technique in the system tests, which is simpler and presents results like the technique proposed by [7]. Finally, the work presented by [6] warns of the need to annually update the images used for system training, which will be done when students enroll.

III. Overview Of The Student Path Monitoring Platform

Fig. 1 shows the overview of the student trajectory monitoring platform. It is possible to see the actors involved, the main applications and the technologies used. From left to right, the “User Registration” appears first. It is a web application developed with the React.js framework that will be used by system administrators to register students from public schools. In this register, the personal data of each student and some photos captured on the spot by a webcam will be inserted. Following just below, the “Model Training” appears, which is a microservice developed with the Python language. This microservice takes photos of students as input and makes use of a series of artificial intelligence algorithms to create a model capable of recognizing the faces of registered students. Next, the model training result is stored in MongoDB, which is a non-relational database.

At the core of the architecture is the Back-end Server that performs the main functions that make possible the coordinated functioning of the various parts of the system. Among these functions, the following stand out: controlling access to the database, sending notifications to users, and communicating with web services. The Back-end Server uses the JavaScript language and the Node.js framework. Next to the Server is the “System Management” which is a web application also implemented with the React.js framework. “System management” oversees the entire system and, among other things, creates administrator profiles.

At the top center of Fig. 1, the first application can be observed: “App for controlling access to the school bus through facial recognition”. This application will be the first to be deployed and aims to monitor the use of the school bus. How this application works will be detailed in a later topic.

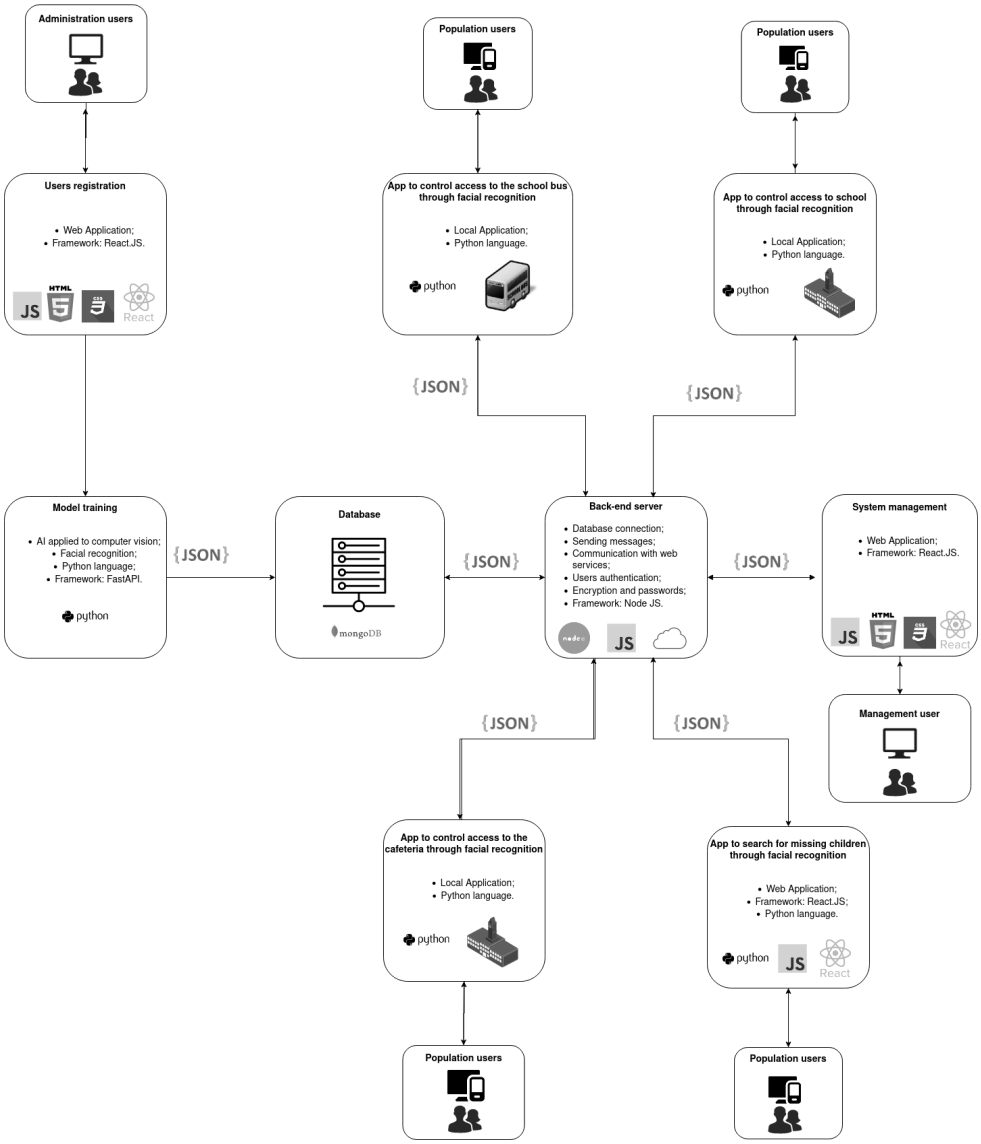


Fig. 1. Facial recognition system architecture.

At the top right is the application “School access control app through facial recognition”. This application uses the Python language and runs on a local device. It has several hardware and software similarities with the first application. The main difference is in the interaction with other actors such as the municipal education department, the Guardianship Council, and the Prosecution Office. The education department will receive a report on student attendance. The Guardianship Council and the Prosecution Office, upon prior registration, will obtain a report with the names of students who have been absent three times or more in a month. In this way, the referred application will not only be a control of the entrance of students, but also be an instrument to mitigate school dropout.

At the bottom left one can see the “App for controlling access to the cafeteria through facial recognition”. This is also an application developed in Python that runs on a local device. In addition to controlling access to the cafeteria, this application, through the Back-end Server, notifies those responsible when a student enters the cafeteria to eat school lunch and can also be used by the education department to control the lunch stock.

Finally, at the bottom right appears the “App to search for missing children through facial recognition”. This is a web application programmed in Python that makes use of the model trained with the images of students from public schools and through security cameras installed in public places of the Smart City to seek to recognize missing children.

IV. APPLICATION OF SCHOOL BUS ACCESS CONTROL THROUGH FACIAL RECOGNITION

Fig. 2 shows all the steps of the school bus access control system. Initially a person tries to get on the bus. At that moment, a camera strategically placed inside the bus captures the image of that person's face. Then, the captured image is processed by an intelligent algorithm written in the Python language and embedded in a Raspberry Pi 4 installed on the bus. The algorithm capable of recognizing faces checks if it belongs to a previously registered student. If the recognition is positive, the turnstile is released for the student to enter. At the same time, the Raspberry Pi sends the information about the student boarding the bus to a server on the Internet, containing a timestamp with date and time. Subsequently, the Internet service sends the student's entry information to the person in charge who was also previously registered. The person in charge receives the notification directly in the application developed for smartphones, which today is the main way to access the Internet in Brazil, widely used by practically all social classes. Finally, Fig. 2 illustrates the application for the Internet that manages the facial recognition system and whose main function is to register students and their respective guardians.

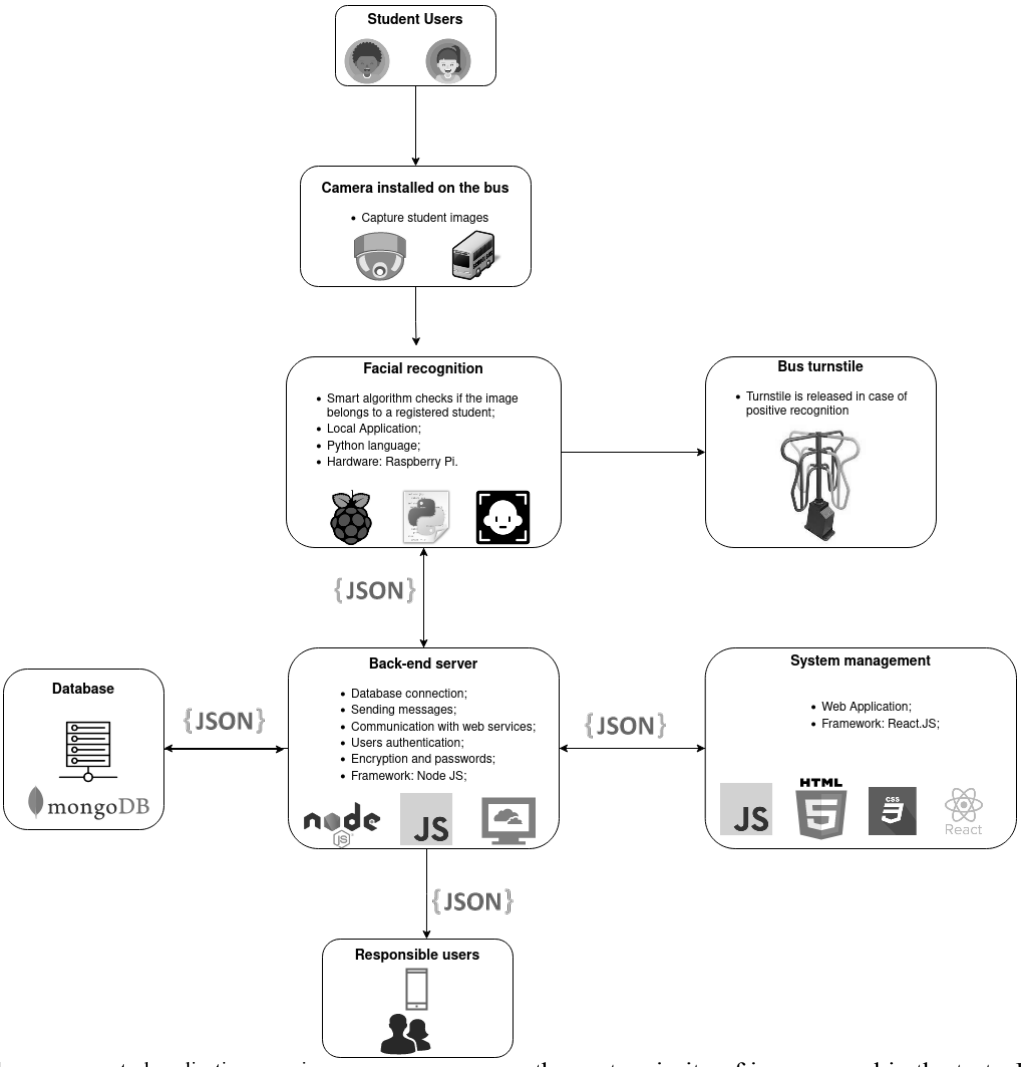


Fig. 2. School bus access control application overview.

Fig. 3 shows the four main steps of the facial recognition process [2]. In the first step, the HOG algorithm is used to search the input image for a pattern corresponding to a human face. If this pattern is found, a rectangle is drawn to delimit the space where the face is located. Fig. 4 (left) shows an example output from the first step, with the face delimited. The image belongs to the LFW public dataset.

In the second step, the HOG and SVM algorithms are used together. The objective of this step is to detect, in the previously delimited space of the face, 68 reference points. Fig. 4 (right) shows an example of the second step output, with the reference points marked in gray.

In the third step, the 68 facial landmarks points are used as input to a previously trained artificial neural network. It is a CNN of the ResNet type [1]. This neural network generates in its output a facial descriptor that is a numerical vector of 128 positions that geometrically describes the main features of the face. This step also saves a file with the descriptors of each image and another file with the respective image labels.

In the fourth step, the KNN algorithm is used, which works as follows: Initially, a facial descriptor is generated for the test image. Then, the Euclidean distance between the test image descriptor and each of the training image descriptors is calculated [8]. Finally, the smallest distance is selected and if this distance is within a previously established threshold, the test image receives the same label as the corresponding training image. In successive experiments, a threshold of 0.5 proved to be sufficient to accurately determine the identity of the vast majority of images used in the tests. Fig. 4 shows an example of the fourth step output. The number in the image represents the Euclidean distance.

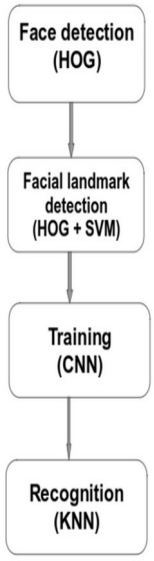


Fig. 3. Recognition process pipeline.

Fig. 4. Example of detected face, facial landmarks and recognized face

V. RESULTS

The LFW database was used in the first test of the facial recognition system. This base was chosen because it is widely used in benchmarks and because it is frequently cited in publications in facial recognition. It is a public database created by the University of Massachusetts from images of famous people taken from the Internet. The base consists of images of politicians, sportsmen, artists, and businessmen from different parts of the world, men, and women of different ethnic groups. In most of the images people are not posing for the photos, but just being themselves. Altogether there are 13233 images of 5749 people, however, most people only have one or two images.

Initially, a subset was separated with those with 7 or more images, which totals 158 people. Then this subset was divided into two categories, one for training and the other for testing. The training set has between 5 to 10 images of each person. The test set, on the other hand, has 2 other images of each of the 158 people. All images were labeled with the person's first name, last name, and a sequential number. The result obtained in this test was the following: 310 correct answers from a total of 316 images tested, which means an accuracy of 98.10%. This result is very close to the one presented in [2].

Table I presents the main metrics related to the first test. To facilitate the visualization, only the first 16 classes and the final averages are shown. The weighted average accuracy (precision) for all classes is 99%. The weighted average recall for all classes is 98%. Finally, the weighted average f1-score for all classes is 98%.

Precision represents the proportion of true positives and false positives, so a high value of precision presupposes a low value of false positives.

The recall, in turn, indicates the proportion between true positives and false negatives, so a high recall implies a low value of false negatives.

On the other hand, the f1-score is a harmonic average of the two previous metrics that will be high only if both accuracy and recall are high.

TABLE I. SUMMARY OF THE MAIN CLASSIFICATION METRICS.

	Precision	Recall	F1-score	Support
Abdullah Gul	1.00	1.00	1.00	2
Adrien Brody	1.00	1.00	1.00	2
Alejandro Toledo	1.00	1.00	1.00	2
Alvaro Uribe	1.00	1.00	1.00	2
Amelie Mauresmo	1.00	1.00	1.00	2
Andre Agassi	0.67	1.00	0.80	2
Andy Roddick	1.00	0.50	0.67	2
Angelina Jolie	1.00	1.00	1.00	2
Ann Veneman	1.00	1.00	1.00	2
Anna Kournikova	1.00	1.00	1.00	2
Ari Fleischer	1.00	1.00	1.00	2
Ariel Sharon	1.00	1.00	1.00	2
Arnold Schwarzenegger	0.67	1.00	0.80	2
Atal Bihari	1.00	1.00	1.00	2
Bill Clinton	1.00	1.00	1.00	2
Bill Gates	1.00	1.00	1.00	2
Accuracy			0.98	316
Macro avg	0.98	0.97	0.97	316
Weighted avg	0.99	0.98	0.98	316

In Fig. 5 there are some examples of people who were recognized by the proposed system. It is possible to see that the system recognizes faces at different angles and with different expressions.

In a second test, the system was trained with photos from personal files and then tested using a webcam on an Acer Nitro 5 laptop with an Intel core i7 processor and 8 GB of RAM, the same used in the first test. In both tests, each recognition occurred in a fraction of a second, showing that the system is efficient and effective.

Fig. 5. Examples of recognized people.

Finally, the facial recognition system was tested on a Raspberry Pi 4 with an ARM Cortex-A72 processor, 4 GB of RAM and a camera module developed for the Raspberry Pi. In this test, the recognition was made with the help of the camera module and used to activate a relay module that in the final product will activate the electric lock of the school bus turnstile. Also in this test, the recognition was performed in a fraction of a second and the relay module was activated correctly.

VI. Conclusions

The recognition system proposed here achieved, in the tests carried out, excellent results in all metrics. It also showed to be able to recognize people with different facial expressions and at different angles. The recognition takes fractions of a second, which demonstrates that the referred system can be incorporated into the school routine without causing delays in boarding buses or entering the school. The hardware used has small dimensions and low power consumption and can be installed almost anywhere. It already has both Wi-Fi and Bluetooth connections and can also receive a module with GPS and GSM enabling the tracking of school buses in real time.

The monitoring platform can be accessed through an Internet browser or through mobile devices allowing both parents and school management to monitor students' journeys from home to school and from school to home. This information may also be automatically passed on to other competent authorities such as the Prosecution Office and the Guardianship Council. Thus, the system can be used as a tool to mitigate school dropout and the disappearance of children and adolescents.

Finally, the first application, that is, the application of access control to the school bus through facial recognition, is in line with all the requirements presented by the stakeholders and is at maturity level TRL 6, that is, it has already been demonstrated the validity of critical prototype functions in relevant environments.

Acknowledgment

This work was supported by the Coordination for the Improvement of Higher Education Personnel—CAPES, the National Council for Scientific and Technological Development— CNPq, the Municipal Fund for Sustainable Development of Canaã dos Carajás —FMDS, and the Support Program for Qualified Production—PROPESP/UFPA (PAPQ) (notice 02/2022). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

[1] F. Chollet, Deep learning with python, Shelter Island, 2018.

[2] D. Schroff, D. Kalenichenko, J. Philbin, “Facenet: a unified embedding for face recognition and clustering”, IEEE explore, pp. 815-823, 2015.

[3] T. Coqueiro, J. Jailton , T. Carvalho, and R. Francês,"A fuzzy logic system for vertical handover and maximizing battery lifetime in heterogeneous wireless multimedia networks", Hindawi Wireless Communications and Mobile Computing, Volume 2019.

[4] R. Ranjan, V. Patel, R. Chellappa, “Hyperface: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition”, IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 121-135, 2019.

[5] V. Jerald, A. Rabara, D. Bai, “Internet of things (IOT) based smart environment integrating various business applications”, International Journal of Computer Applications, pp. 32-37, 2015.

[6] D. Deb, N. Nain, K. Jain, "Longitudinal study of child face recognition", International Conference on Biometrics, pp. 225-232, 2018.

[7] K. Huang, Y. Li, C. Loy, and X. Tang, “Deep imbalanced learning for face recognition and attribute prediction”, IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 2781-2794, 2020.

[8] F . Provost, T. Fawcett, Data science para negócios. Alta Books, 2016.