UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS I – CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIAS DA COMPUTAÇÃO

ALEKSANDRO DA COSTA FABRÍCIO

RECONHECIMENTO FACIAL:
UTILIZANDO UM SISTEMA DE CHAMADAS AUTOMÁTICO PELO SMARTPHONE

CAMPINA GRANDE - PB

2020

ALEKSANDRO DA COSTA FABRÍCIO

RECONHECIMENTO FACIAL:
UTILIZANDO UM SISTEMA DE CHAMADAS AUTOMÁTICO PELO SMARTPHONE

Trabalho de Conclusão de Curso apresentado pela Universidade Estadual da Paraíba (UEPB), como requisito para conclusão do curso de Bacharelado em Ciências da Computação.

Orientador: Prof. Dr. Robson Pequeno de Souza

CAMPINA GRANDE - PB

2020

F126r Fabricio, Aleksandro da Costa.
Reconhecimento facial [manuscrito] : utilizando um sistema de chamadas automático pelo smartphone / Aleksandro da Costa Fabricio. - 2020.
59 p. : il. colorido.
Digitado.
Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências e Tecnologia , 2021.
"Orientação : Prof. Dr. Robson Pequeno de Souza , Coordenação do Curso de Computação - CCT."
1. Inteligência Artificial. 2. Redes Neurais. 3. Reconhecimento Facial. I. Título
21. ed. CDD 006.3

Elaborada por Camile de A. Gomes - CRB - 15/559 BC/UEPB

ALEKSANDRO DA COSTA FABRÍCIO

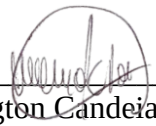
RECONHECIMENTO FACIAL:
UTILIZANDO UM SISTEMA DE CHAMADAS AUTOMÁTICO PELO SMARTPHONE

Trabalho de Conclusão de Curso apresentado pela Universidade Estadual da Paraíba (UEPB), como requisito para conclusão do curso de Bacharelado em Ciências da Computação.

Aprovada em 07 de Dezembro de 2020.

Prof. Dr. Robson Pequeno de Souza (DC - UEPB)
Orientador(a)

Prof. Dr. Paulo Eduardo e Silva Barbosa (DC - UEPB)
Examinador(a)



Prof. Dr. Wellington Candeia de Araújo (DC - UEPB)
Examinador(a)

Dedico este trabalho aos meus pais Teobaldo e Laurenice, que me mostraram o caminho para chegar até aqui e à minha esposa Mikaella, por estar sempre ao meu lado.

AGRADECIMENTOS

Quero expressar meus agradecimentos,

Ao coordenador Paulo Eduardo Barbosa, do curso de Ciências da Computação, por sempre buscar me engajar nos projetos durante o curso.

Ao professor Robson Pequeno de Souza, por estar sempre disposto a ajudar, principalmente ao longo desse período de orientação.

Ao meus pais Teobaldo Filgueira Fabrício e Maria Laurenice da Costa Fabrício, por nunca desistirem de mim e sempre me incentivar em todos os projetos da minha vida.

Aos meus irmãos Alessandra da Costa Fabrício e Alex da Costa Fabrício, por todo o apoio e momentos de amizade.

À minha esposa Mikaella Santos de Souza, por estar sempre ao meu lado e me apoiando em todo o meu percurso.

Aos colegas de classes, em especial a Alex, Eduardo, Thiago, Lucas, Lancelote, Caio, Marcos e todos os outros que estiveram sempre ao meu lado nos momentos de quebra-cabeça com projetos do curso, em tempos de desespero para as provas e pelos momentos de amizade.

À ECIT Alfredo Pessoa de Lima, local que serviu de laboratório para esta pesquisa, na pessoa do gestor escolar Valdeci Alves Diniz.

Aos alunos do 2ºA, do curso Técnico de Manutenção e Suporte em Informática, por terem participado desta pesquisa.

A todos que participaram desta minha jornada, agradeço.

"Eu imagino um mundo onde a inteligência artificial nos permitirá ser mais produtivos, viver mais, e ter energia mais limpa."

Fei-Fei Li

RESUMO

O uso da Visão Computacional atualmente vem se tornando cada vez mais versátil, podendo trazer diversas soluções que automatizam processos diários. Além disso, graças à evolução tecnológica, podemos encontrar Smartphones com um bom poder de processamento capaz até de executar tarefas complexas. A proposta deste trabalho foi desenvolver um protótipo de software para Smartphone que utiliza algoritmos de reconhecimento facial para facilitar o processo de registro de chamadas de presença em salas de aula, sem a necessidade de conexão com à Internet, ou seja, utilizando apenas o poder de processamento do Smartphone. Para isso, foi utilizado um modelo já treinado e uma biblioteca contendo algoritmos específicos para tal finalidade. Foram feitos testes tanto de reconhecimento facial quanto de identificação facial em diversas imagens e os resultados obtidos demonstraram que o algoritmo tem uma boa eficiência, tornando-o bastante promissor para ser usado não só em reconhecimento facial, mas também utilizando de outros modelos treinados para identificar outros tipos de padrões.

Palavras-Chave: Inteligência Artificial; Redes Neurais; Reconhecimento Facial.

ABSTRACT

The use of Computational Vision is currently becoming increasingly versatile and can bring several solutions that automate daily processes. In addition, thanks to technological evolution, we can find Smartphones with a good processing power capable of even performing complex tasks. The purpose of this work was to develop a prototype software for Smartphone that uses facial recognition algorithms to facilitate the process of recording presence calls in classrooms, without the need for internet connection, that is, using only the processing power of the Smartphone. For this, we used a trained model and a library containing specific algorithms for this purpose. Both facial recognition and facial identification tests were performed on several images and the results showed that the algorithm has a good efficiency, making it very promising to be used not only in facial recognition, but also using other models trained to identify other types of patterns.

Keywords: Artificial Intelligence; Neural Networks; Facial Recognition.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CRUD	Create, Read, Update, Delete
MLP	Multilayer Perceptron (Peceptron Multicamadas)
MVP	Minimal Viable Product (Mínimo Produto Viável)
QR	Quick Response
RAM	Read Access Memory (Memória de Acesso Aleatório)
RFID	Radio Frequency Identification
RNA	Rede Neural Artificial
RNC	Rede Neural Convolucional
RNC	Rede Neural Convolucional
RNN	Rede Neural Natural

SUMÁRIO

1. INTRODUÇÃO

Com o avanço populacional no mundo está cada vez mais necessário que soluções mais eficientes estejam atuando para facilitar as ações na sociedade. Assim como Ferreira (2017) diz que “a tecnologia e a inovação são dois itens que proporcionam evolução e revolução. Quem não acompanhar esse ritmo de transformação fica desatualizado e fora do contexto social.”.

Graças às tecnologias atuais – que estão cada vez mais poderosas – é possível criar diversas soluções que tornam nossas atividades mais eficientes, que mudam a forma como lidamos com grupos de pessoas ou até a sociedade como um todo.

Dentre estas tecnologias, está a evolução dos computadores, especialmente os Smartphones. Facilmente encontramos modelos até de baixo custo com um considerável espaço de armazenamento interno e sendo comum aparelhos de no mínimo 4 gigabytes de memória RAM, além de ter processadores com vários núcleos e uma boa frequência de processamento, sendo bem fácil de encontrar processadores com 4 até 8 núcleos e acima de 2 giga hertz de processamento (SCHERER, 2020).

Uma das tecnologias mais marcantes e que se mostra cada vez mais útil na construção dessas soluções é a Visão Computacional, que se trata da capacidade do computador em identificar formas geométricas presentes em uma imagem através de padrões que representam tais elementos e, assim, possibilitando a extração de suas informações dentro do conteúdo da imagem atual (ALVES, 2005, p. 14).

Diversas são as aplicações que podem ser desenvolvidas para este tipo de tecnologia, como na área automação de sistemas de direção e de aeronaves, detecção de sorrisos para fotos automáticas, identificação de pessoas, carros, qualquer objeto que possua um determinado padrão construído e identificado através de técnicas de Inteligência Artificial (MIRANDA, 2011, p. 13-14).

Também podem ser citados como exemplos o processamento de classificação em imagens para sistemas de vigilância em vídeo, permitindo que a máquina consiga reconhecer e melhorar constantemente o reconhecimento facial, servindo para identificar criminosos conhecidos, ou sendo capazes de identificar comportamentos e atividades que estão fora da norma ou quebram a lei (DATA SCIENCE ACADEMY, 2018), ou até algoritmos capazes de detectar em vídeos ou imagens emoções das pessoas como raiva, alegria, tristeza, nojo, surpresa e medo (EXPERT ACADEMY, 2020).

Além disso, Ponti & Costa (2017, p. 63) afirma que:

> Nos últimos anos, técnicas de Aprendizado Profundo tem revolucionado diversas áreas de aprendizagem de máquina, em especial a visão computacional. Isso ocorreu principalmente por dois motivos: a disponibilidade de bases de dados com milhões de imagens, e de computadores capazes de reduzir o tempo necessário para realizar o processamento dessas bases de dados.

Tais soluções são produtos de algoritmos capazes de processar informações de imagens, inclusive muitos destes funcionam junto com algoritmos de Aprendizado de Máquina para se ter soluções com bom desempenho, assim conforme Miranda (2011, p. 14) apresenta que:

> O aprendizado é um aliado valioso no desenvolvimento de sistemas de visão computacional voltados ao reconhecimento e classificação de imagens, visto que além de aprender, as técnicas de aprendizado de máquinas são capazes de generalizar o conhecimento adquirido durante o processo de treinamento.

No Brasil podemos encontrar em diversas empresas o uso da Visão Computacional junto com algoritmos de Aprendizado de Máquina para, por exemplo, o reconhecimento facial. São empresas como a ViaQuatro, que instalou em plataformas de embarque e desembarque um sistema que identifica as emoções das pessoas diante de determinada publicidade; a Hering que adotou um sistema de identificação para traçar o perfil do público que mais frequenta suas lojas através do gênero, faixa etária e humor; ou a Zaitt, que utilizou o reconhecimento facial para autenticação de clientes (SIMÃO, 2020).

Soluções que envolvem estas tecnologias estão se tornando cada vez mais confiáveis, tanto que existem algoritmos capazes de processar imagens, através de reconhecimento de padrões, para identificar se um paciente possui lesões a partir de imagens térmicas (ALVES, A., 2018); ou até como uma “segunda opinião” para auxiliar no diagnóstico médico, melhorando a qualidade do serviço médico prestado (SILVA, 2016).

1.1. PROBLEMÁTICA

Em diversos setores profissionais, a verificação da presença é algo essencial, seja para registro de cumprimento do expediente, como até para registro de presença em grupos de pessoas.

Observando a realidade de registros em salas de aula, em que os professores sempre precisam registrar a presença dos alunos. Diversas técnicas foram desenvolvidas para garantir que todos os presentes sejam registrados, porém as práticas que já são adotadas não apresentam ser completamente eficazes. Conforme Mattos (2017, p. 12):

> [...] durante a chamada, um professor pode vir a “pular” um nome, ou, até mesmo, um aluno distraído pode não perceber que foi chamado. No caso de uma lista de presença, algumas pessoas podem acabar não tendo contato com ela, caso circule de maneira irregular.

Além disso, neste caso o professor precisa dedicar parte do horário da aula para verificar a presença dos alunos e em momentos que é passada a lista para os alunos assinarem, os mesmos podem burlar as assinaturas, assinando no nome de alguém que não está presente em aula para não ser penalizado.

Como formas alternativas, existem tecnologias que procuram tornar mais fácil ou até tornar este processo mais eficiente. Dentre elas, podem ser citadas como exemplo o uso de cartões de identificação utilizando RFID (*Radio Frequency Identification*) ou códigos impressos (código de barras ou código QR), biometria digital e até reconhecimento facial dos alunos (MATTOS, 2017, p. 12).

No entanto, até estes métodos possuem fatores limitantes que o tornam inviáveis ou apresentam falhas e limitações que, inclusive, o aluno pode usar para burlar e garantir a presença, mesmo que de outro aluno, vejamos a seguir:

- **Cartões de identificação:** para chips RFID há a necessidade de hardware específico para programar o chip e ler as informações gravadas, dispositivo que muitas vezes não está ao alcance da instituição ou do professor. Neste caso os alunos podem perder ou danificar o seu cartão, ou até mesmo podem usar o cartão de outro aluno que não está presente para contabilizar a presença, tornando necessária a atenção do professor para exercer tal controle e, provavelmente, pode até não ser mais rápido que a chamada convencional (TEIXEIRA, 2011, p. 68);
- **Biometria digital:** possui a necessidade de hardware específico para leitura da digital do aluno. Uma vantagem é que os alunos não conseguem registrar a presença de outro que não está na aula, tirando a necessidade da intervenção do professor. No entanto, neste caso ainda há a intervenção do aluno, precisando ir até o dispositivo biométrico para registrar sua presença. Com isso, pode acontecer de alunos esquecerem de fazer o registro de sua presença, além de ser um método não higiênico, pois muitos alunos terão que tocar na mesma superfície (MOVIO, 2015, p. 20);
- **Reconhecimento facial:** esta é uma solução mais eficiente que as anteriores, pois é mais rápida e não precisa da intervenção do professor ou aluno. No mercado existem soluções para sala de aulas utilizando câmeras fixas que são ligadas a um computador com o software dedicado ligado a um servidor com os algoritmos de reconhecimento da biometria facial (FaceID), ou soluções que utilizam a câmera do próprio smartphone do professor ligado a um servidor através da internet. Para o primeiro tipo de solução é necessário investimento de câmeras, de um computador e de fiação para a conexão dos dispositivos, o que torna até inviável para algumas realidades de instituições onde existem muitas salas

> de aula. Já para a solução que utiliza um smartphone para coletar a imagem dos alunos e enviar a um servidor, apenas funciona através de uma aplicação móvel que se comunica ao servidor, sendo necessário ter internet disponível para sua execução, ou seja, o professor precisa estar sempre conectado durante o processo de chamada (MATTOS, 2017, p. 25).

Observando esta última solução, a realidade para a maioria dos professores e instituições de ensino é que não possuem condições de investimento em muitos equipamentos, além de não terem acesso à Internet – ou até não possuindo computador em alguns casos. Tais condições são vitais para o funcionamento das soluções apresentadas anteriormente, portanto, não seriam aplicadas ou precisam de um grande processo de instalação para entrarem em vigor nestes locais de trabalho. Por isso, podemos levantar a seguinte questão norteadora **"É possível construir uma solução tecnológica viável para este caso que seja independente de conexão com a Internet?"** e, além dessa, considerando que os smartphones atuais – mesmo sendo de baixo custo – possuem um considerável poder de processamento, podemos também levantar a seguinte questão **"É possível construir uma solução de reconhecimento de imagens para Smartphones?".**

1.2. JUSTIFICATIVA

A escolha do desenvolvimento desta pesquisa e protótipo de software é definida pelo pesquisador estar trabalhando como professor e como o mesmo possui contato direto com esta realidade, percebe esta necessidade básica em seu ramo de trabalho. Sabendo que é, de fato, não produtivo ter que fazer a chamada de presença dos alunos – através dos métodos tradicionais como falar o nome de cada aluno – em todas as aulas que vai lecionar, perdendo um tempo precioso que muitas vezes é bem curto para a aula.

Por isso é válida uma solução que facilite este processo de chamadas, considerando também que como não há sempre a presença de conexão com à Internet, se torna um requisito não funcional uma solução sem a necessidade (ou a mínima necessidade) de intervenção do professor e do aluno, além de não precisar de conexão com à Internet para realizar o registro da presença dos alunos.

Escolher desenvolver uma aplicação para smartphone coincide com a realidade de praticamente todos professores, já que estes têm acesso a tal tecnologia, principalmente possuindo, na grande parte destes, um smartphone com sistema Android e que, mesmo sendo um de baixo custo, apresenta um poder de processamento considerável. Dessa forma, há uma maior chance desta tecnologia ser implantada em qualquer instituição de ensino e os professores podem utilizá-la sem qualquer investimento em dispositivos.

1.3. OBJETIVO GERAL

Desenvolver um protótipo de software para smartphone que possua execução de chamada de presença por meio de reconhecimento facial, independente de Internet ou qualquer conexão para identificação do aluno.

1.4. OBJETIVOS ESPECÍFICOS

- Investigar sobre as soluções já existentes no mercado;
- Determinar a tecnologia para a implantação da aplicação móvel;
- Determinar as bibliotecas necessárias para a utilização do reconhecimento facial;
- Desenvolver o protótipo;
- Comprovar que um smartphone é capaz de processar imagens para reconhecimento facial;
- Testar a eficácia do classificador utilizado no reconhecimento facial;

2. FUNDAMENTAÇÃO TEÓRICA

2.1. VISÃO COMPUTACIONAL

Um dos processos mais complexos já implementados foi a Visão Computacional, que se trata da capacidade do computador em reconhecer elementos no espaço (em três dimensões) através de suas imagens (em duas dimensões) (DATA SCIENCE ACADEMY, 2018). Além disso, Mesquita (2015, p. 28) diz que:

> A captura de uma imagem digital é realizada por dispositivos capazes de discretizar os dados de cor, iluminação ou níveis de cinza e acomodá-los em uma ou várias matrizes [...]. Cada elemento da matriz, conhecido como 'pixel' (abreviação de Picture element), possui um valor associado a ele que representa a intensidade de luz refletida naquele ponto.

Dessa forma, o algoritmo faz um processamento nas matrizes que representam a imagem em busca de sequências de pixels que se aproximam dos padrões presentes nos modelos testados para a identificação e classificação dos objetos, podendo ter até o auxílio de técnicas de Inteligência Artificial para melhorar este processo. Miranda (2011, p. 17) acrescenta que "essa forma particular é chamada de reconhecimento de padrões, classificando dados visuais numéricos ou simbólicos baseando-se em informações contidas em bancos de dados de padrões.".

Ainda em Miranda (2017, p. 20, apud GONZALES e WOODS, 2000), é apresentado que nas características representadas por estes padrões – elementos necessários para a classificação do objeto na imagem – estão as bordas (ou descontinuidades), característica importante para delimitar o perímetro do objeto, acrescentando que "descontinuidades que destacam-se pela presença de pixels que estão na fronteira entre duas regiões com intensidades luminosas relativamente distintas.".

2.1.1. Reconhecimento Facial

Bicalho (2013, p. 13) categoriza o processo de reconhecimento facial em dois cenários:

- **Reconhecimento de rostos:** as faces presentes na imagem são identificadas através da aproximação com um modelo contendo pesos treinados a partir de um banco de dados contendo milhares de imagens já previamente classificadas.
- **Identificação da pessoa:** é feita a comparação entre duas imagens, sendo uma já previamente identificada e a outra processada e comparada com a primeira para encontrar o seu nível de proximidade.

Para ambos os casos, as mesmas características de uma face são consideradas, sendo denominadas de marcações ou descritores, que são nada mais do que pontos bidimensionais que

identificam o formato dos **olhos**, **nariz**, **boca** e **contorno do queixo** e **maxilar**, como apresentado nas faces encontradas na **Figura 1** abaixo (BICALHO, 2013, p. 17).

Figura 1 - Representação dos descritores de faces em uma imagem

Fonte: https://github.com/justadudewhohacks/face-api.js.

2.2. APRENDIZAGEM DE MÁQUINA

Também conhecido como *Machine Learning*, se trata uma subárea da Inteligência Artificial e conta com um conjunto de técnicas de algoritmos utilizado para tornar a máquina capaz de "aprender" através da experiência, sendo possível devido à sua capacidade de armazenar e recuperar grandes quantidades de informações, adaptando o comportamento da solução de acordo com o tipo de dado processado (SHAI & SHAI, 2014).

Este tipo de algoritmo pode ser considerável dinâmico por se adaptar ao problema que precisa resolver, procurando ter o melhor desempenho possível. Dessa forma, Miranda (2011, p. 24) diz que:

> Uma forma de avaliar o desempenho de uma técnica de aprendizado de máquina aplicada a um determinado problema consiste no cálculo estatístico da taxa de acertos para cada técnica em questão [...]. Este método consiste em subdividir o conjunto de exemplos utilizados para o sistema realizar o aprendizado em dois grupos, um para treinamento e outro para avaliação, sendo que as estimativas para o desempenho futuro da técnica são realizadas com base no segundo.

Sendo assim, um algoritmo de Aprendizagem de Máquina atua com base na experiência adquirida, que por sua vez esta experiência é construída através de tarefas que o algoritmo deve executar e, a partir disso, é calculado o erro do qual irá determinar se será necessário aprimorar os

parâmetros do algoritmo, repetindo este processo até que o valor mínimo aceitável do erro seja atendido. Sendo que dentre as possíveis tarefas que um algoritmo pode executar nesse contexto, a mais comum é a **Classificação**, em que o software irá analisar uma determinada quantidade de categorias de dados na entrada e ao concluir este processo, um modelo será gerado com as características dos elementos identificados (GOODFELLOW et al, 2016).

Além disso, Júnior (2018, p. 36), apresenta que o aprendizado de máquina pode ser classificado em três tipos principais:

- **Aprendizado supervisionado:** ocorre quando os dados de entrada já possuem classificações pré-estabelecidas. Com isso o algoritmo pode treinar modelos que ao inserir novos dados, ele irá pertencer a uma destas classificações já estabelecidas pelos dados iniciais;
- **Aprendizado não-supervisionado:** quando os dados de entrada não possuem classificações pré-estabelecidas. Dessa forma eles serão agrupados de acordo com a proximidade de seus valores e pela quantidade de grupos definida pelo desenvolvedor;
- **Aprendizado por reforço:** além dos dados de entrada já possuírem rótulos pré-determinados para a criação do modelo, a cada entrada de novos dados ele irá adquirir novas informações para se auto avaliar, ajustando seu próprio modelo para atingir um determinado objetivo.

2.3. REDES NEURAIS ARTIFICIAIS

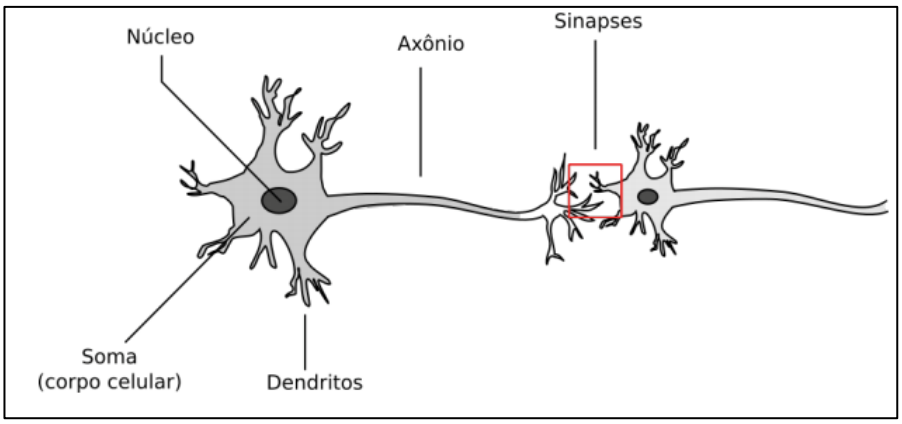
Inspiradas no funcionamento de sistemas nervosos e na forma como os neurônios comunicam entre si, a criação das RNAs (Redes Neurais Artificiais) trouxe uma forma de processar dados em alta performance para a construção de soluções para diversos tipos de problemas como reconhecimento de padrões, diagnósticos médicos, reconhecimento de fala e escrita, mineração de dados, dentre outros (AZEVEDO, 2016, p. 7). Seguindo este conceito, Miranda (2011, p. 25, apud HAYKIN, 2001) complementa que:

> Uma rede neural é uma máquina projetada para modelar a maneira como o cérebro realiza uma tarefa ou função de interesse, podendo ser vista como um processador extremamente paralelizado e distribuído, constituído de unidades de processamento simples (neurônios), normalmente implementada utilizando-se componentes eletrônicos ou simulada por software em computador.

Dessa forma, enquanto que na RNN (Rede Neural Natural) seu sistema nervoso é constituído por cerca de 86 bilhões de células nervosas, nas quais a entrada de informações ocorrem pelos dendritos e a saída pelos axônios, além destas informações serem passadas por meio

de sinapses de efeito variável entre os neurônios (**Figura 2**) – células que podem se adaptar de acordo com o tipo de informação (AZEVEDO, 2016, p.8; RUFINO 2017, p.13; MIRANDA, 2011; p. 25). As RNAS possuem conexões mais simples chamados de nodos (ou nós) e estas conexões possuem uma variável chamada peso, esta que é determinada de acordo com o processamento durante o treinamento (RUFINO, 2017, p. 13).

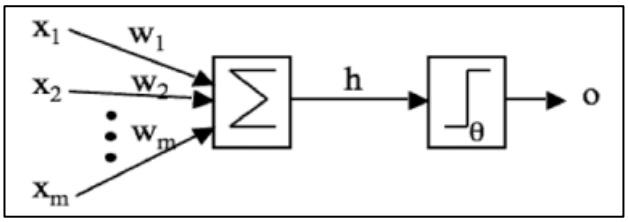
Figura 2 - Modelo de um neurônio humano



Fonte: (AZEVEDO, 2016, p. 20)

De acordo com Miranda (2011, p. 28), o modelo matemático que representa um neurônio artificial é o modelo de McCulloch e Pitts (**Figura 3**), que é composto pelas entradas $\boldsymbol{X_1, X_2, \ldots, X_N}$ que representam conexões de outros neurônios e os pesos destas conexões $\boldsymbol{W_1, W_2, \ldots, W_N}$ como as sinapses de uma RNN.

Figura 3 - Modelo matemático do neurônio de McCulloch e Pitts



Fonte: (MIRANDA, 2011, p. 29, apud MARSLAND, 2008)

Para este caso, o somatório do produto da entrada pelos pesos irá resultar em um valor resultante $\boldsymbol{h}$, ou seja:

$$\boldsymbol{h} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \boldsymbol{w_i x_i}$$

(2.1)

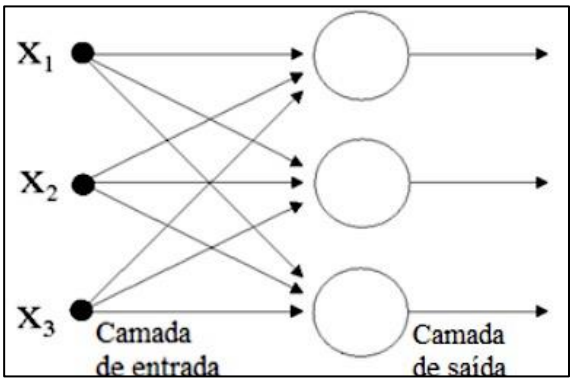
Como neste modelo o valor resultante é um binário (0 ou 1), então a sua função de ativação pode ser reescrita através da seguinte equação:

$$\boldsymbol{a} = \boldsymbol{g(h)} = \begin{cases} \mathbf{1}\ \boldsymbol{se}\ \boldsymbol{h} > \mathbf{0} \\ \mathbf{0}\ \boldsymbol{se}\ \boldsymbol{h}\ \leq \mathbf{0} \end{cases}$$

2.3.1. Perceptron

Este tipo de RNA é o mais básico dentre todos e consiste basicamente em uma rede neural com apenas uma camada de entrada e uma camada de saída (**Figura 4**), sendo utilizadas para separação (ou classificação) de elementos linearmente separáveis (RUFINO 2017, p. 17).

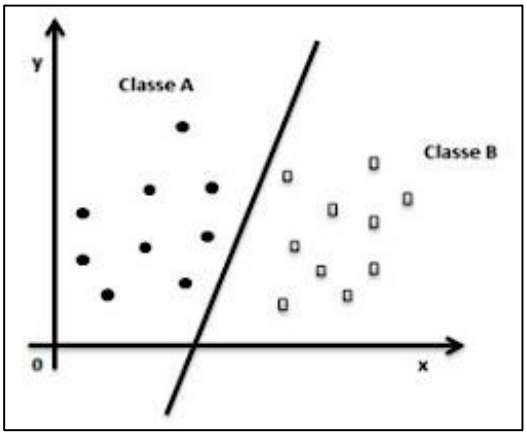
Figura 4 - Exemplo de RNA de camada única



Fonte: (RUFINO, 2017, p. 17, apud PAULA, 2002)

Miranda (2011, p. 31-32) apresenta que “se os padrões utilizados para treinar o Perceptron são retirados de duas classes linearmente separáveis, então o algoritmo converge e posiciona a superfície de decisão na forma de um hiperplano entre as duas classes”. Esta definição é chamada de *teorema da convergência do Perceptron*, provada por Rosenblatt, em 1958.

Figura 5 - Classes linearmente separáveis

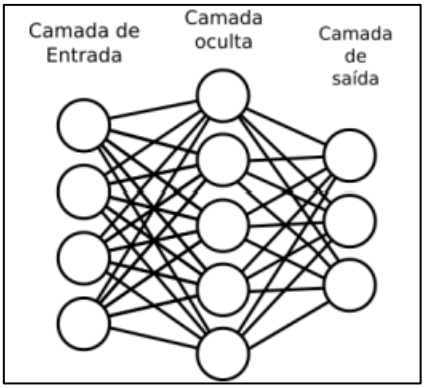


Fonte: (MIRANDA, 2011, p. 32)

2.3.2. Perceptron Multicamadas

Enquanto que o Perceptron de camada única é alimentado apenas uma única vez com os dados de entrada e consegue separar classes de forma linear, o Perceptron de Múltiplas Camadas(*Multilayer Perceptron* – MLP), contém a camada de entrada, de saída e uma ou mais camadas ocultas entre a de entrada e saída, e consegue separar de forma não linear (AZEVEDO, 2016, p. 23).

Figura 6 - Modelo de uma rede do tipo perceptron com múltiplas camadas

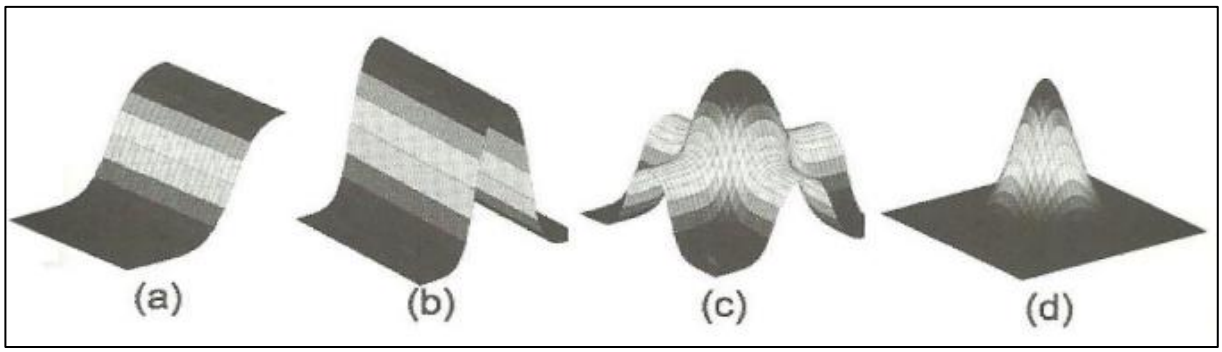


Fonte: (AZEVEDO, 2016, p. 21)

Segundo Miranda (2011, p. 33), “normalmente não é necessário utilizar mais do que três camadas, duas ocultas e uma de saída, ao modelar o MLP para realizar o reconhecimento de um determinado conjunto de padrões.”.

Para o MLP, quando se aplica mais camadas ocultas, como se trata de uma função sigmoidal como resultante da função de ativação, mais cristas são adicionadas à função, que se combinadas podem gerar funções com um único ponto de máximo (MIRANDA, 2011, p. 34).

Figura 7 - O aprendizado do MLP pode ser vista em (a) como saída única de uma única função sigmoidal, que pode servir como entrada para outra camada, gerando a curva (b). Adicionando outra curva de 90º nós produzimos (c) uma crista, que pode ser aperfeiçoado (d) para a forma que queremos



Fonte: (MIRANDA, 2011, p. 33, apud MARSLAND, 2008)

2.3.3. *Deep Learning*

Semelhante à estrutura seguida pelo MLP, o *Deep Learning* (Aprendizado profundo) possui uma maior quantidade de camadas de processamento de informações e uma maior entrada de dados. Além disso Júnior (2018, p. 47) complementa que há

> Maior número de neurônios, aumentando a complexidade dos modelos, maiores níveis de conexão entre as camadas, permitindo maior quantidade de parâmetros para gerenciar, necessitando maior poder computacional para o treinamento e extração automática de características (filtros).

Portanto, graças à essa grande quantidade de informações que são processadas por esta arquitetura de RNA, é possível reconhecer objetos visuais sem a necessidade de extração prévia de suas características (AZEVEDO, 2016, p. 24).

2.3.4. Redes Neurais Convolucionais (RNC)

Sendo uma variante da MLP, a principal funcionalidade de uma Rede Neural Convolucional (*Convolutional Neural Networks* – CNN) é a capacidade de reconhecer formas bidimensionais e identificar padrões a partir dos pixels de entrada. Geralmente, este tipo de RNA é utilizado em problemas de Visão Computacional, principalmente porque seus dados são um conjunto de informações na forma de um plano (AZEVEDO, 2016, p. 24).

Basicamente, é dito que uma rede neural se torna Convolucional quando pelo menos uma de suas camadas faz a convolução, que Júnior (2018, p. 48) define como “uma operação matemática entre duas funções f e g, produzindo uma terceira função, que pode ser chamada de função modificada de f, utilizada para detecção de bordas, suavização de imagens, extração de atributos, dentre outros usos”.

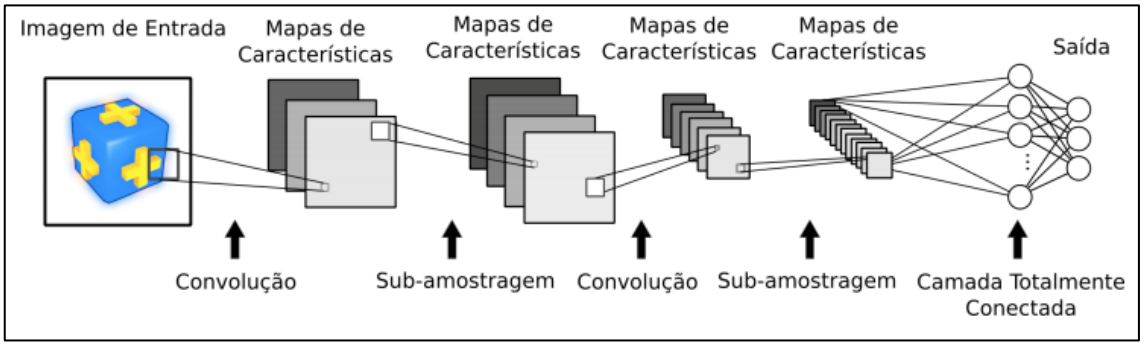
De acordo com Azevedo (2016, p. 24, apud O’SHEA e NASH, 2015), a arquitetura de uma RNC pode ser caracterizada da seguinte forma:

- **Convolutional Layers:** o neurônio recebe uma seção bidimensional da camada anterior (uma parte da imagem da camada anterior);
- **Pooling Layers:** executa o processo de redução da complexidade do modelo construído através da redução da dimensionalidade espacial dos dados de entrada;
- **Fully-connected layers:** geralmente são as camadas finais da RNC. Neste caso ocorre o processo de treinamento e classificação, e seus neurônios estão conectados a todos da camada anterior.

Além disso, Júnior (2018, p. 48, apud MARTINS, 2017), aponta que as principais diferenças de uma RNC em relação à outras RNAs são:

- **Tensores k-dimensionais de neurônios:** quando os neurônios são 3D (largura, altura e profundidade), estes recebem apenas uma parte da região presente na camada anterior;
- **Conectividade local:** através da correlação espacial, a RNC consegue garantir um padrão de conectividade entre os neurônios de camadas adjacentes, garantindo uma rápida resposta espacial;
- **Pesos compartilhados:** os valores dos pesos sempre são constantes, garantindo que as características de entrada sempre sejam detectadas quando encontrados os padrões desejados.

Figura 8 - Modelo de uma rede do tipo convolutiva



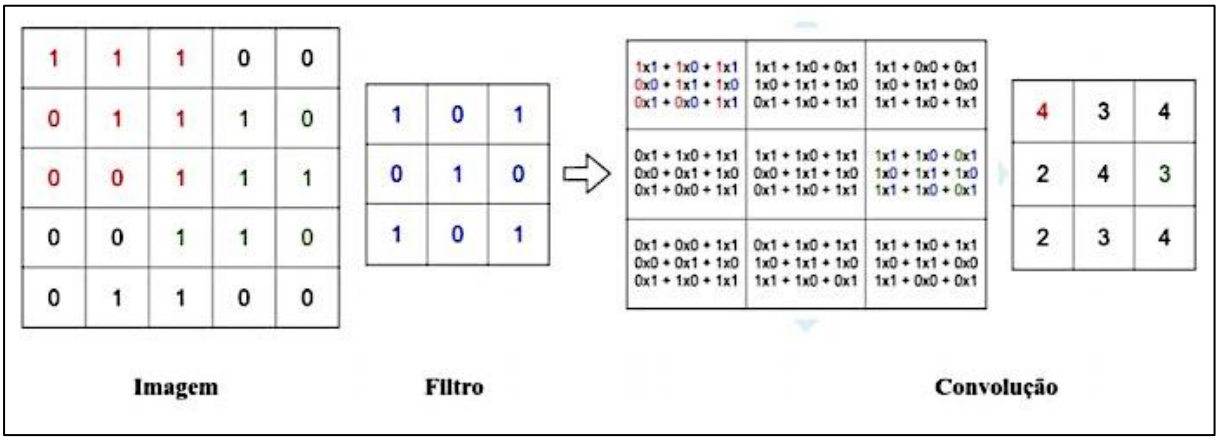
Fonte: (AZEVEDO, 2016, p. 26)

Com o intuito de detalhar como os valores dos pesos interagem com os pixels, Júnior (2018, p. 53) mostra que:

> A rede é inicializada com um conjunto de pequenos pesos e vieses aleatórios, sendo alimentada em seguida, com os dados de entrada. Na camada em que ocorre a convolução, esta pode ser interpretada como sendo o somatório da multiplicação de cada elemento da imagem e seus vizinhos locais, pelos elementos da matriz filtro da convolução.

Ou seja, ocorre um processo semelhante a percorrer o filtro por toda a imagem, porém, todo o processo ocorre de forma paralela, como acontece na **Figura 9** a seguir:

Figura 9 - Processo hipotético de convolução de uma imagem 5x5 recebendo um filtro 3x3



Fonte: Adaptado de (JÚNIOR, 2018, p. 53, apud FERREIRA, 2017)

De acordo com Gonzalez & Woods (2018, p. 161), pode-se notar que a RNC que opera através do processamento das camadas vizinhas se trata de uma convolução utilizando um filtro[1] $\boldsymbol{w}$ em um ponto (x, y), sendo sua expressão dada por:

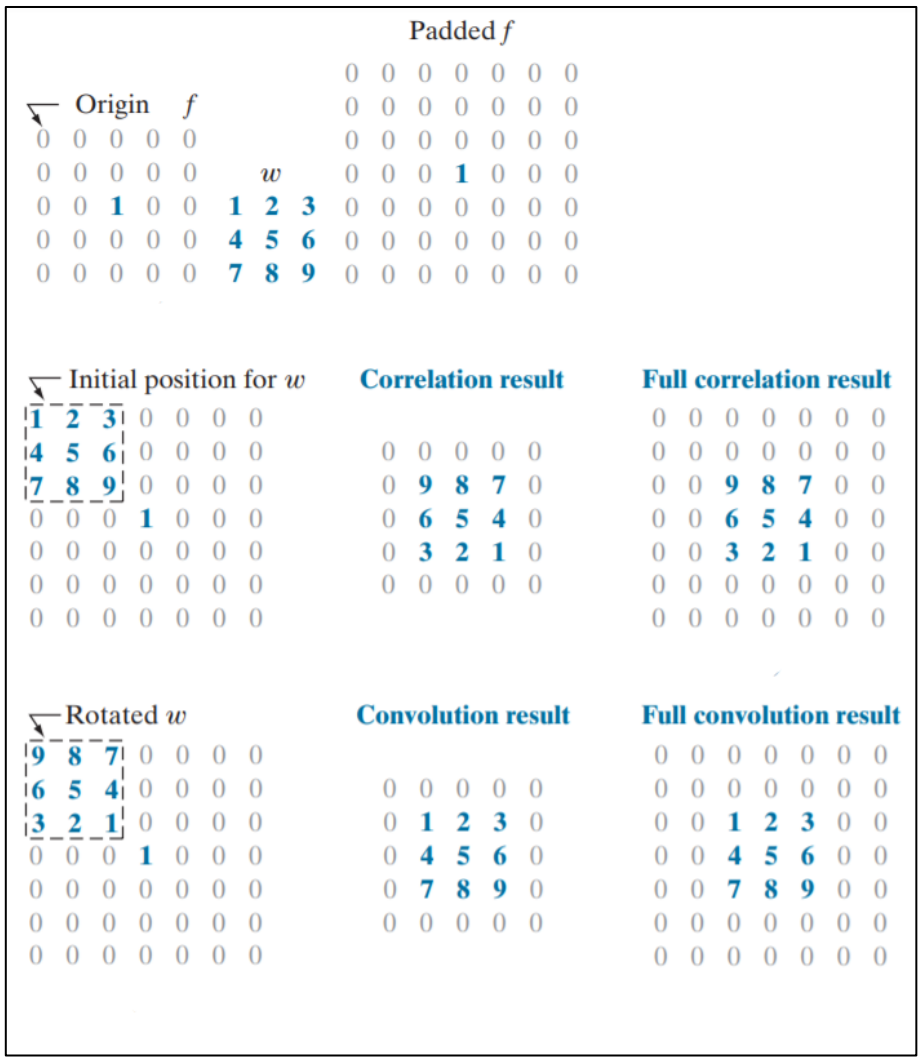
$$\boldsymbol{w} * \boldsymbol{f}_{x,y} = \sum_{l=-a}^{a} \sum_{k=-b}^{b} \boldsymbol{w}_{s,t} * \boldsymbol{f}_{x-l,y-k} \tag{2.2}$$

, onde o operador da convolução é representado por $*$ e $\boldsymbol{l}$ e $\boldsymbol{k}$ são os valores da dimensão do filtro.

Antes de iniciar o processamento, o filtro aplicado precisa ser rotacionado 180º para ser uma convolução, caso contrário irá ocorrer uma correlação (GONZALEZ & WOODS, 2018, p. 159). Podemos ver esse comportamento na **Figura 10** a seguir:

[1] Na literatura, a denominação "filtro" também pode ser encontrada como "kernel" (JÚNIOR, 2018) ou "filter kernel" (GONZALEZ & WOODS, 2018).

Figura 10 – Exemplo de processo de correlação e convolução em uma imagem f



Fonte: Adaptado de (GONZALES & WOODS, 2018, p. 160)

Substituindo $\boldsymbol{f_{x,y}}$ por $\boldsymbol{a_{x,y}}$ e considerando um filtro $\boldsymbol{w}$ com dimensões $3x3$, ao expandir a equação **(**2.2) apresentada anteriormente, temos:

$$\boldsymbol{w * a_{x,y} = w_{1,1} * a_{x-1,y-1} + w_{1,2} * a_{x-1,y-2} + \cdots w_{3,3} * a_{x-3,y-3}}$$

(2.3)

De acordo com Júnior (2018, p. 49), para definir a equação de entrada de dados no neurônio e, ao final, definir a função de ativação $\boldsymbol{h(z)}$**,** devemos seguir os seguintes passos:

a) Renomeia-se os índices de $\boldsymbol{w}$ e $\boldsymbol{a}$**:**

$$\boldsymbol{w} * \boldsymbol{a}_{x,y} = \boldsymbol{w}_1 * \boldsymbol{a}_1 + \boldsymbol{w}_2 * \boldsymbol{a}_2 + \cdots \boldsymbol{w}_9 * \boldsymbol{a}_9$$

(2.4)

, então temos:

$$\boldsymbol{w} * \boldsymbol{a}_{x,y} = \sum_{i=1}^{9} \boldsymbol{w}_i \boldsymbol{a}_i$$

(2.5)

b) Sendo as equações **(**2.1) e **(**2.5) idênticas e adicionando o viés (*bias*), encontramos a equação resultante $\boldsymbol{z}$**:**

$$\boldsymbol{z} = \boldsymbol{w} * \boldsymbol{a}_{x,y} + \boldsymbol{b} = \sum_{i=1}^{9} \boldsymbol{w}_i \boldsymbol{a}_i + \boldsymbol{b}$$

(2.6)

c) Considerando $\boldsymbol{h}$ como a função de ativação, a saída do neurônio é obtida através da seguinte equação:

$$\boldsymbol{a} = \boldsymbol{h}(\boldsymbol{z})$$

Ainda em Júnior (2018, p. 49), é dito que "inserindo os kernels na equação [...], onde os índices sobrescritos são referentes aos *feature maps*, os valores de l, k, x, e y são os mesmos em todas as três equações, porque todos o três kernels possuem o mesmo tamanho e se movem conjuntamente", ressaltando que os *feature maps* são os Mapas de Características apresentados na **Figura 8**. Portanto, considerando a equação expandida **(**2.4), temos:

$$\overset{(\mathbf{1})}{\boldsymbol{w}_{l,k} * \boldsymbol{a}_{x,y}} + \overset{(\mathbf{2})}{\boldsymbol{w}_{l,k} * \boldsymbol{a}_{x,y}} + \overset{(\mathbf{3})}{\boldsymbol{w}_{l,k} * \boldsymbol{a}_{x,y}} = \sum_{l}\sum_{k} \overset{(\mathbf{1})}{\boldsymbol{w}_{l,k}} * \boldsymbol{a}_{x-l,y-k} + \sum_{l}\sum_{k} \overset{(\mathbf{2})}{\boldsymbol{w}_{l,k}} * \boldsymbol{a}_{x-l,y-k} + \sum_{l}\sum_{k} \overset{(\mathbf{3})}{\boldsymbol{w}_{l,k}} * \boldsymbol{a}_{x-l,y-k}$$

Ou seja, ao resumir esse processo e adicionar o viés $\boldsymbol{b}$ para camada seguinte, temos:

$$\boldsymbol{z}_{x,y} = \sum_{l}\sum_{k} \boldsymbol{w}_{l,k} \boldsymbol{a}_{x-l,y-k} + \boldsymbol{b}$$

$$\boldsymbol{z_{x,y} = w * a_{x,y} + b}$$

(2.7)

Chegando no valor correspondente $\boldsymbol{a_{x,y}}$ e encontrando a função de ativação:

$$\boldsymbol{a_{x,y} = h(z_{x,y})}$$

(2.8)

Júnior (2018, p. 50) acrescenta que para diferenciar os valores das camadas anteriores, é necessário adicionar uma notação para evidenciar essa diferenciação, abordando que em uma camada totalmente conectada é utilizada uma notação $\boldsymbol{\ell}$, reescrevendo as equações **(**2.7) e **(**2.8):

$$\boldsymbol{z_{x,y}(\ell) = w(\ell) * a_{x,y}(\ell - 1) + b(\ell)}$$

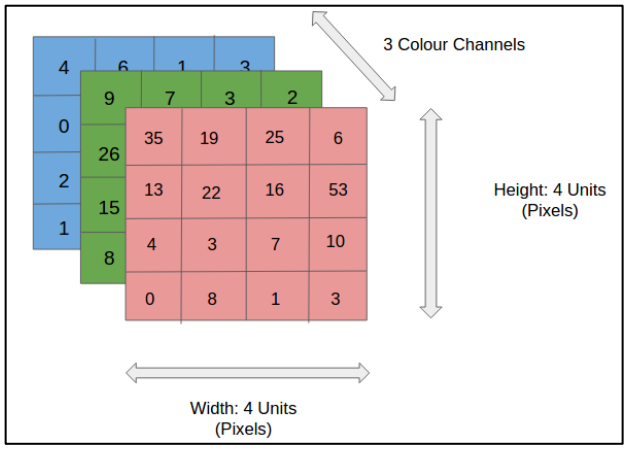
(2.9)

$$\boldsymbol{a_{x,y}(\ell) = h(z_{x,y}(\ell))}$$

(2.10)

Para o treinamento do modelo deste projeto, os dados de entrada são basicamente matrizes de três dimensões (**Figura 11**) de acordo com o tamanho da imagem, sendo que a largura e altura são as dimensões da imagem, a profundidade é a quantidade de canais de cores que normalmente encontramos em imagens que possuem os três canais RGB (*Red Green Blue*).

Figura 11 – Exemplo de matrizes de entrada que representam a imagem



Fonte: (ALVES, G., 2018)

Alves (G., 2018) acrescenta que “quanto mais profundas são as camadas das convoluções, mais detalhados são os traços identificados com o *activion map*.”. Além disso, a definição da

quantidade de camadas e os pesos iniciais são definidos manualmente, e quanto maior a quantidade de camadas (maior profundidade), mais detalhados são os traços identificados, porém custando um maior processamento.

Sobre a saída da convolução, na última camada da RNC é colocada uma camada completamente conectada (*Fully Connected*) e é calculado o seu erro (ALVES, G., 2018).

De acordo com Júnior (2018, p. 50):

> Redes neurais profundas utilizam treinamento por retropropagação, que consiste em quatro passos básicos: (1) inserção dos valores de entrada; (2) treinamento da rede por propagação, para definir todos os padrões de treinamento e determinar o erro de classificação ou regressão; (3) o retorno por retropropagação, que avalia a saída de erro através da rede e calcula as mudanças requeridas par atualização dos parâmetros; e (4) atualização dos pesos e vieses (bias) da rede. Esses passos são repetidos até que a rede atinja erros em níveis aceitáveis.

Então, para calcular a retropropagação para cada neurônio em cada *feature map*, na camada $\boldsymbol{\ell}$, utiliza-se da definição da equação para o erro, em função de $\boldsymbol{x}$ e $\boldsymbol{y}$ (JÚNIOR, 2018, p. 50):

$$\boldsymbol{\delta_{x,y}(\ell)} = \frac{\boldsymbol{\delta E}}{\boldsymbol{\delta z_{x,y}(\ell)}}$$

(2.11)

A partir disso que se define a seguinte equação do cálculo da retropropagação (JÚNIOR, 2018, p. 54):

$$\boldsymbol{\delta_{x,y}(\ell)} = h'\left(\boldsymbol{z_{x,y}(\ell)}\right)\left[\boldsymbol{\delta_{x,y}(\ell+1)} * \boldsymbol{rot180}\left(\boldsymbol{w(\ell+1)}\right)\right]$$

(2.12)

Por fim, caso o erro não atinja os níveis desejados de aceitação, atualiza-se os pesos $\boldsymbol{w_{l,k}(\ell)}$ e vieses (bias) $\boldsymbol{b(\ell)}$ para cada *feature map* (JÚNIOR, 2018, p. 54):

$$\boldsymbol{w_{l,k}(\ell)} = \boldsymbol{w_{l,k}(\ell)} - \boldsymbol{\alpha\delta_{x,y}(\ell)} * \boldsymbol{rot180(a(\ell-1))}$$

(2.13)

$$\boldsymbol{b(\ell)} = \boldsymbol{b(\ell)} - \boldsymbol{\alpha}\sum_{\boldsymbol{x}}\sum_{\boldsymbol{y}}\boldsymbol{\delta_{x,y}(\ell)}$$

(2.14)

3. METODOLOGIA

Neste capítulo será abordado sobre as tecnologias utilizadas para o desenvolvimento do protótipo de software de alta fidelidade, como ferramentas utilizadas, arquitetura implementada, linguagens e bibliotecas utilizadas.

Sendo que, de acordo com os conceitos de Engenharia de Software, um protótipo é a representação limitada de um software, podendo ser um protótipo de **baixa fidelidade**, que focam mais em construir telas para identificar os processos de interação do usuário com o sistema, por exemplo tendo o esboço das telas em papel; ou um protótipo de **alta fidelidade** que busca se aproximar bem mais do software final, apresentando telas com elementos de layout, cores, páginas com hiperlinks que fazem o usuário transitar entre elas, ou até a ferramenta já funcional porém não finalizada o suficiente para ser considerada um MVP (*Mínimal Viable Product* – Mínimo Produto Viável) (ROSEMBERG et al, 2008).

Para este trabalho, o protótipo desenvolvido foi uma aplicação híbrida para smartphones, mais voltada especificamente para a plataforma Android, em que o usuário tem acesso a este software em seu próprio smartphone e todas as funcionalidades incluídas nele não necessitam de conexão com à Internet.

3.1. FERRAMENTAS E LINGUAGENS UTILIZADAS

Para o desenvolvimento desse protótipo foi utilizado um framework voltado para a construção de aplicações híbridas denominado Ionic. Como a base do Ionic é Angular, então o desenvolvimento é próximo ao de um *Software Web*. No **Quadro 1** estão as ferramentas e tecnologias utilizadas no desenvolvimento do protótipo.

Quadro 1 - Ferramentas utilizadas para o desenvolvimento do protótipo

Nome	Versão	Descrição	URL
Typescript	3.9.5	Derivado do Javascript, porém suas variáveis são tipadas	www.typescriptlang.org
HTML (HyperText Markup Language)	5	Linguagem de Marcação	www.w3c.br
SCSS (Syntactically Cascade Style Sheets)	-	Linguagem de estilização	www.sass-lang.com
Angular	10	Framework web para front-end	www.angular.io
Ionic	5	Framework para desenvolvimento mobile	www.ionicframework.com

Visual Studio Code	1.51.1	Editor de texto para códigos	www.code.visualstudio.com
Android Studio	4.1.1	IDE para desenvolvimento Android	www.developer.android.com

Fonte: do autor.

3.2. BIBLIOTECAS UTILIZADAS

Utilização de bibliotecas é uma estratégia de reuso de código bem utilizada em todas as linguagens de programação, pois além de ser fácil sua implantação no projeto, garante que o desenvolvedor consiga implementar funções complexas em seu projeto sem precisar fazer desde o início (FILHO, 2013).

Quadro 2 - Bibliotecas utilizadas no desenvolvimento do protótipo

Nome	Versão	Descrição	Comando de instalação NPM
Capacitor	2.4.1	Framework que facilita a execução em tempo real da aplicação móvel de forma nativa	@capacitor/core e @capacitor/cli
Progressive Web Apps (PWA) Elements	3.0.1	Framework que torna o desenvolvimento de um site para se comportar como uma aplicação móvel	pwa-elements
Face Api	0.22.2	Biblioteca responsável por fazer todo o procedimento de reconhecimento facial	face-api.js

Fonte: do autor.

3.3. ARQUITETURA DO SOFTWARE

Uma das principais vantagens no desenvolvimento utilizando o framework Ionic é a estruturação feita por componentes, dos quais podem ser um componente de Página, Serviço, Interface, etc.

Os componentes de Página são compostos por arquivos HTML, SCSS e Typescript, assemelhando-se ao desenvolvimento de uma aplicação nativa para Android. Além disso, graças ao sistema de rotas, o desenvolvedor pode organizar em pastas específicas criadas por ele todos estes componentes criados.

3.3.1. Estruturação do código

Para todas as páginas do protótipo desenvolvido para este projeto, a estratégia de organização escolhida foi organizar no diretório **pages** conforme o seguinte: **pages/aluno-cadastro**, **pages/chamada**, **pages/home**, **pages/turma**, **pages/turma-cadastro**.

Além disso, para os componentes de serviços deste projeto, foram organizados no diretório **services**, em que cada serviço representa:

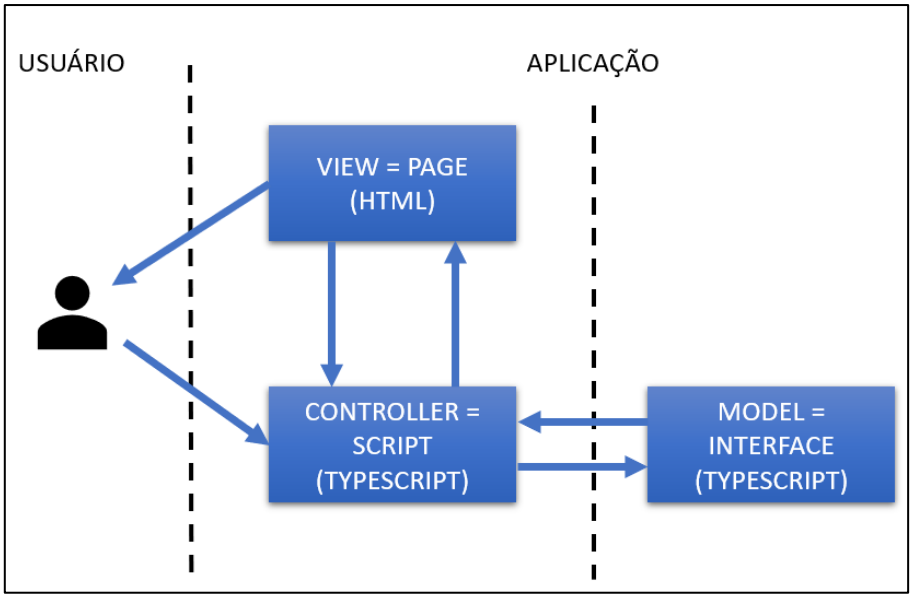
- **Aluno, Chamada e Turma:** contendo os *scripts* de armazenamento no banco de dados destes modelos, além de regras específicas para requisitos funcionais.
- **Foto:** contendo scripts de captação e armazenamento de imagens pela câmera do dispositivo;
- **Recognition:** contendo scripts relacionados ao reconhecimento facial.

3.3.2. Arquitetura

Apesar do framework Ionic ser bem flexível quanto ao desenvolvimento da sua arquitetura, por padrão é utilizado o modelo MVC (Model-View-Controller) presente no AngularJS (TUTORIALSPOINT, 2018, p. 9).

Este modelo pode ser facilmente identificado, de acordo com a **Figura 12**:

Figura 12 - Representação da arquitetura MVC em um projeto Ionic



Fonte: do autor

Além desses componentes apresentados (Script, Page e Interface), foi utilizado o Service, que se trata do responsável por questões como requisitos funcionais, armazenamento das informações de um determinado Model, etc.

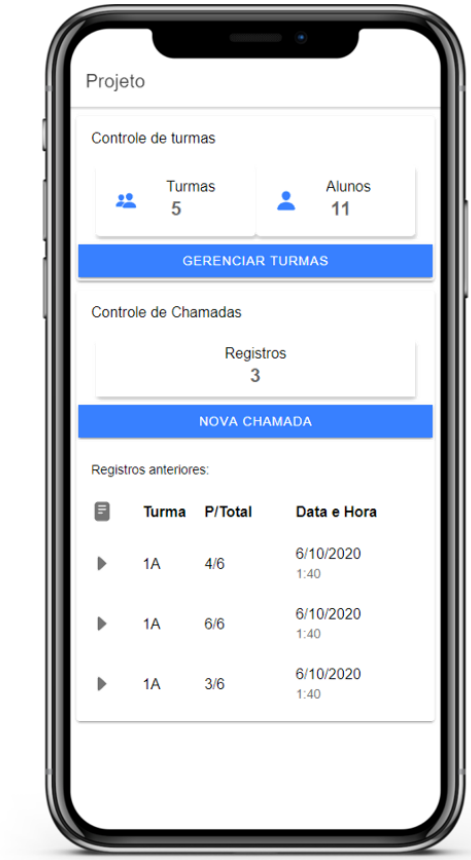
3.4. APRESENTAÇÃO DO SOFTWARE

3.4.1. Apresentação da Interface

A **Figura 13** apresenta a tela inicial da aplicação, esta tela tem como objetivo apresentar ao usuário uma visualização rápida dos seus registros, além dos seguintes recursos:

- **Gerenciar Turmas:** Ao clicar o usuário pode manter as turmas e manter os alunos;
- **Nova chamada:** Ao clicar o usuário pode fazer uma nova chamada;
- **Manter chamada:** Na lista de registros anteriores, ao clicar em um item o usuário pode visualizar e editar informações das chamadas registradas, e ao arrastar o item para esquerda poderá excluir um registro de chamada.

Figura 13 - Representação da tela principal do protótipo em um smartphone



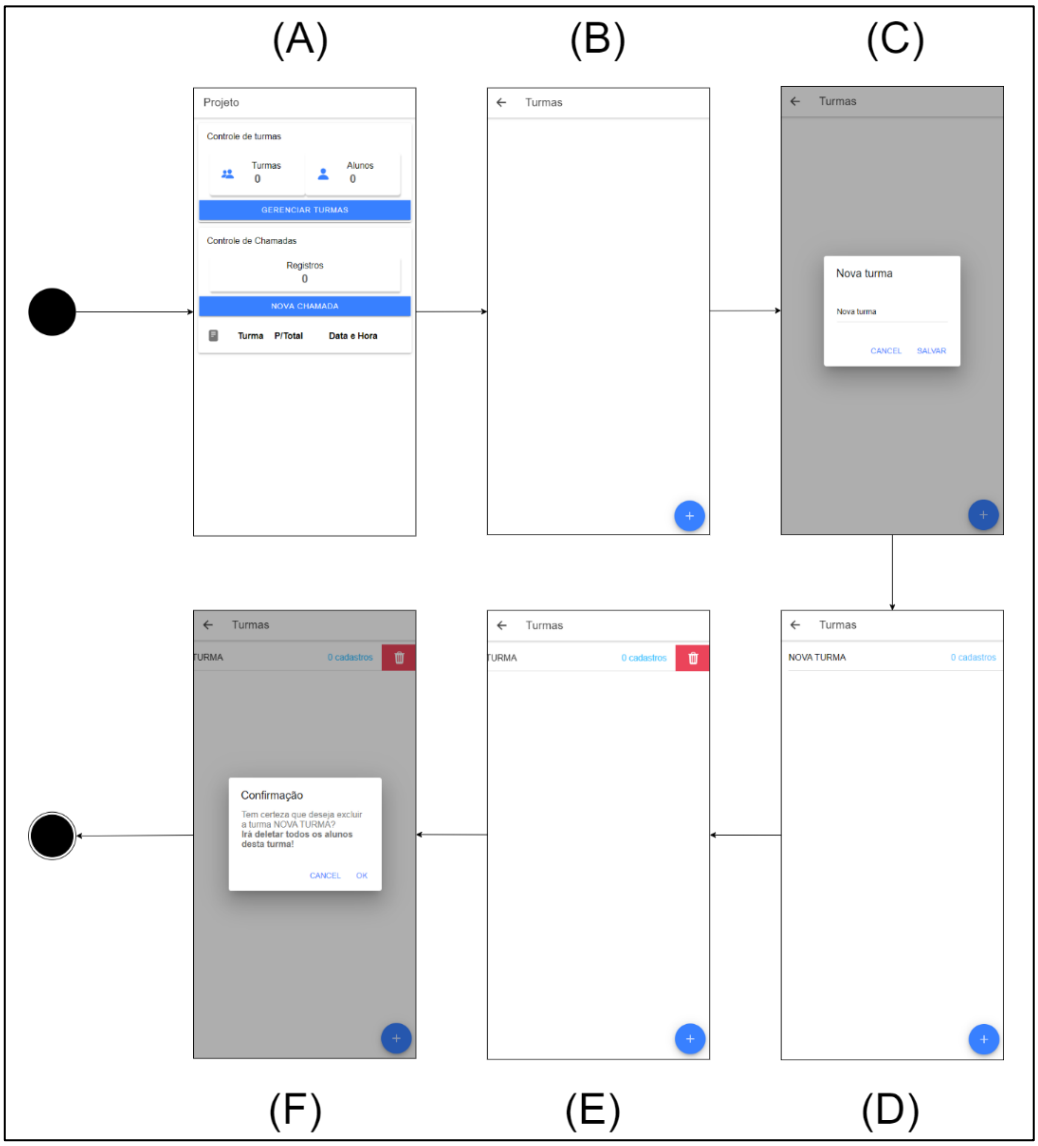
Fonte: do autor

3.4.2. Fluxo de manter Turma

A **Figura 14** apresenta o fluxo de manter turma, seguindo os seguintes passos:

- **Para inserção de uma nova turma:** (A) Clica-se em Gerenciar Turmas; (B) Clica no botão de "+" no canto inferior direito da tela; (C) Insere o nome da turma e clica em salvar;
- **Caso o usuário queira excluir uma turma:** (D) Arrasta o item da lista para a esquerda até aparecer o botão da lixeira; (E) Clica na no botão da lixeira; (F) Confirma a exclusão.

Figura 14 - Fluxo de Registrar e Deletar Turma

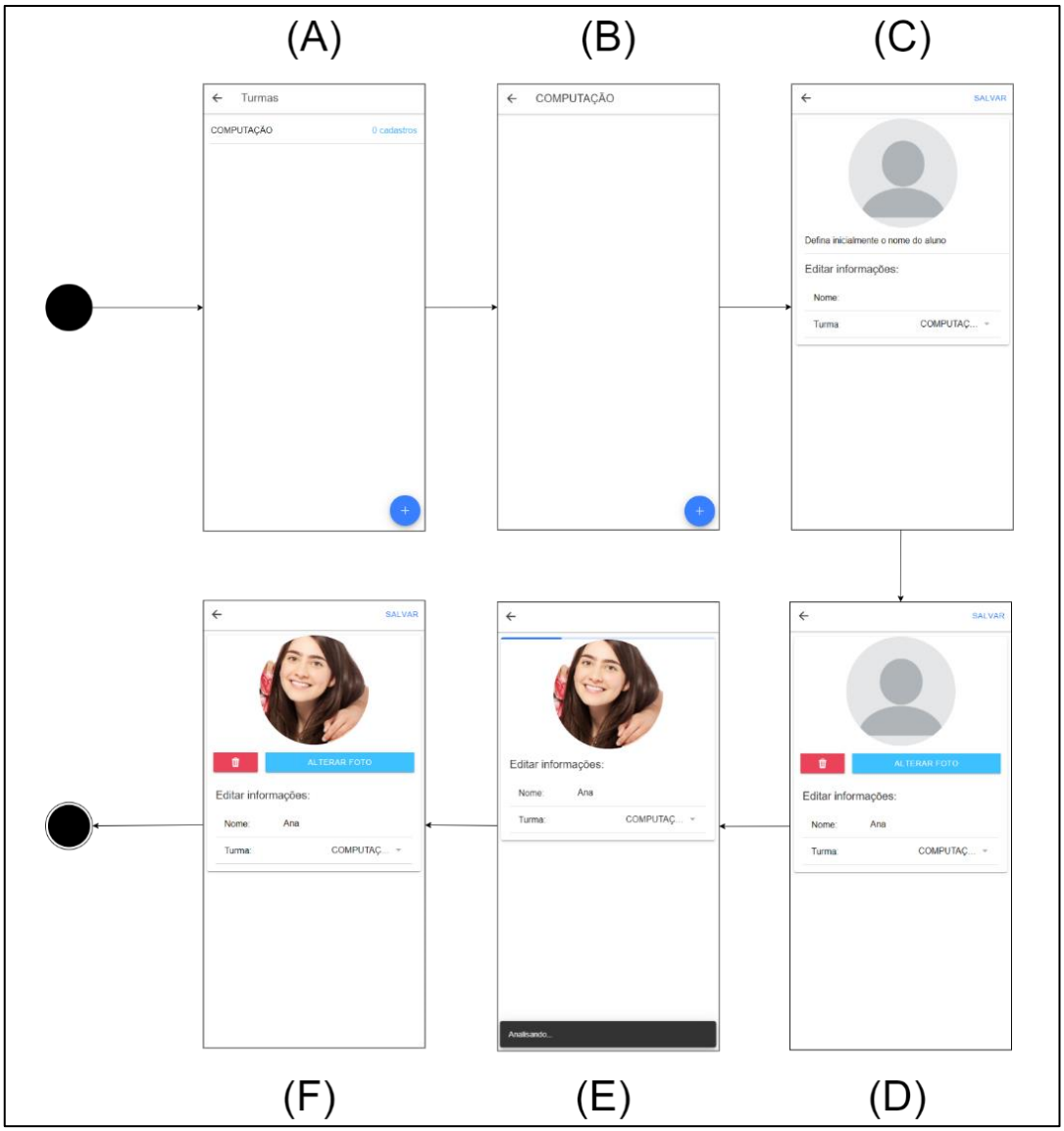


Fonte: do autor.

3.4.3. Fluxo de manter Aluno

Na **Figura 15** pode ser visto o processo para inserir um novo aluno, para que este processo seja feito precisa-se ter uma turma cadastrada. Os passos seguidos para o novo aluno são: (A) Seleciona o item da lista de turmas cadastradas; (B) Clica no "+" no canto inferior direito da tela; (C) Preenche o campo do nome do aluno; (D) Ao preencher o nome irá aparecer os botões para adicionar uma nova foto e excluir; (E) Quando adicionar uma nova foto será feita uma análise no rosto do aluno para coletar os descritores do seu rosto, portanto deve aguardar o processo; (F) Clica no botão de Salvar, no canto superior direito da tela.

Figura 15 - Fluxo de Registrar Aluno



Fonte: do autor.

3.4.4. Fluxo de nova chamada

Neste fluxo, o usuário poderá fazer uma nova chamada desde que tenha a turma e os alunos cadastrados, além disso, para funcionar o reconhecimento facial, os alunos precisam ter em seu cadastro sua foto registrada. No fluxo apresentado na **Figura 16** o usuário poderá registrar uma nova chamada seguindo os seguintes passos:

(A) Clica no botão de Nova Chamada;

(B) Clica em "Selecione" na barra superior;

(C) Selecione uma turma dentre as cadastradas que foram apresentadas na lista;

(D) Irá aparecer a turma cadastrada, para fazer a análise adicione a foto da turma;

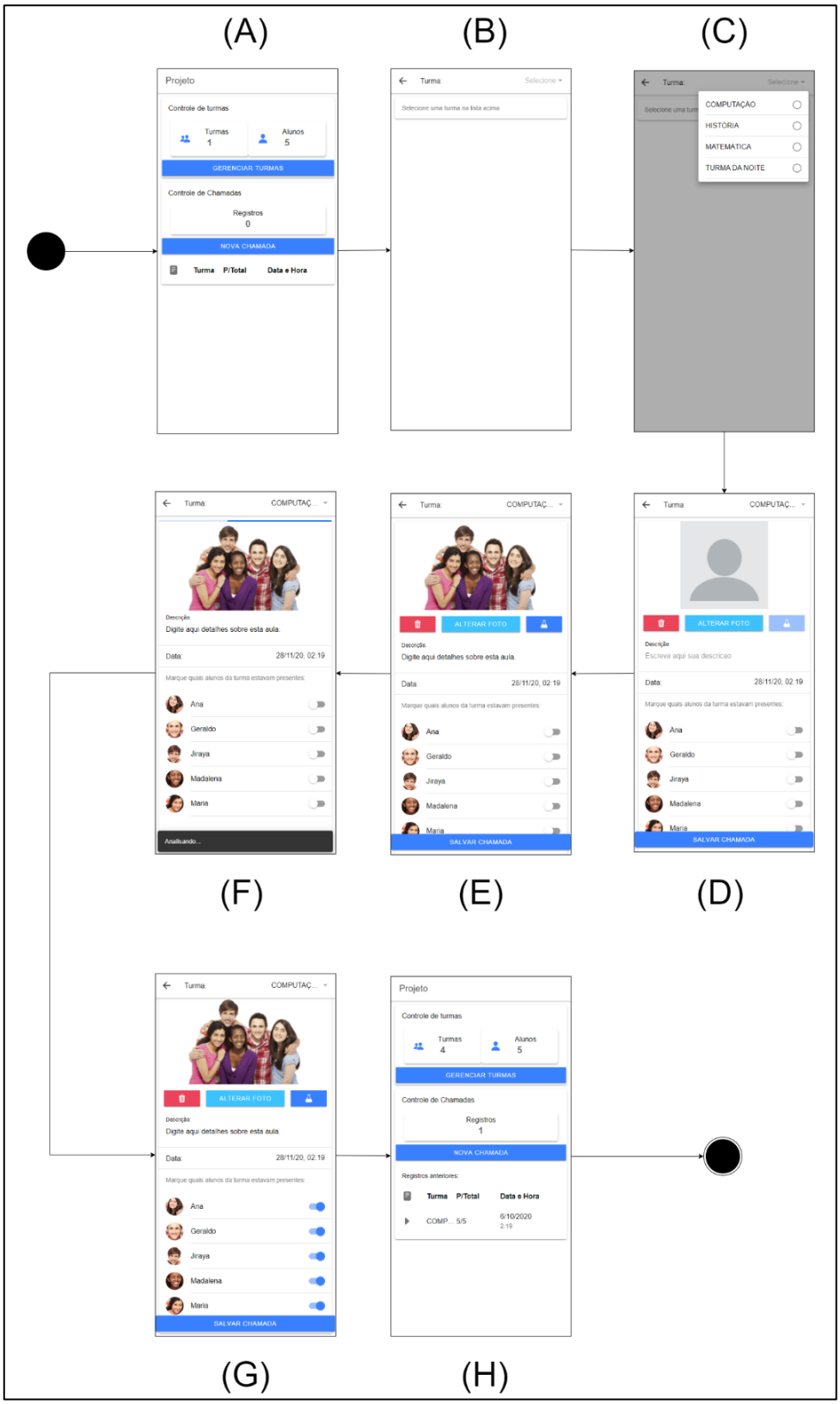
(E) Clique no ícone do lado direito de adicionar uma nova foto para analisar;

(F) Aguarde o processo de análise;

(**G)** Ao analisar ele irá marcar presente em todos os alunos que foram reconhecidos na foto, agora poderá clicar no botão Salvar Chamada;

(H) Voltará para a tela inicial com a nova chamada inseria na lista, note que existe **P/Total**, significa a Quantidade de Presentes / Total de alunos da turma.

Figura 16 - Fluxo de Manter Chamada



Fonte: do autor.

3.5. TREINAMENTO DO MODELO

O modelo foi treinado com o dataset WIDERFACE, um conjunto de 32 mil imagens contendo mais de 393 mil rostos com alto grau de variabilidade em escala, posições e intensidades de luz, conforme apresentado na **Figura 17** abaixo.

Figura 17 - Exemplos de imagens escolhidas no dataset WIDERFACE

Fonte: http://shuoyang1213.me/WIDERFACE/.

Para o treinamento do modelo foi utilizada uma Rede Neural Convolucional através de uma aprendizagem profunda (*Deep Learning*) supervisionada, em que as imagens utilizadas no treinamento já possuíam sua classe pré-determinada.

A estratégia de treinamento utilizada foi executar 61 eventos, sendo que para cada evento de treinamento foi selecionado aleatoriamente 40% das imagens para treinamento dos pesos, 10% para validação e 50% direcionadas para testes.

O modelo treinado ficou disponibilizado no formato de JSON para ser carregado localmente pela aplicação através de métodos próprios desenvolvidos pela biblioteca de reconhecimento facial escolhida (**APÊNDICE A**).

3.6. MÉTRICAS DE AVALIAÇÃO

Para avaliar o desempenho do classificador, foram selecionados dois conjuntos de imagens para verificar os seguintes casos:

- **Reconhecimento de rostos:** foram selecionadas 4 imagens contendo um total de 92 rostos que não foram classificados anteriormente pelo algoritmo, validando a performance do em determinar se é ou não um rosto.

- **Identificação da pessoa:** foram selecionadas 7 (sete) imagens contendo uma considerável quantidade de rostos, com objetivo de encontrar apenas 1 (um), validando a capacidade do comparador de encontrar o rosto certo dentre muitos outros em uma mesma imagem.

Como forma de medição do classificador, foram considerados os seguintes critérios, conforme Alves (A., 2018, p. 83-84):

- **Verdadeiro Positivo (VP):** Corretamente classificado com a classe de interesse;
- **Verdadeiro Negativo (VN):** Corretamente classificado, porém não sendo a classe de interesse;
- **Falso Positivo (FP):** Incorretamente classificado com a classe de interesse;
- **Falso Negativo (FN):** Incorretamente classificado e não sendo a classe de interesse.

Estes critérios são fundamentais para a construção da matriz de confusão, da qual a partir dela poderão ser feitos os cálculos para determinar a precisão, sensibilidade e acuraria do classificador:

		Valores Preditos	
		SIM	**NÃO**
Valores Reais	**SIM**	VP	FP
	NÃO	FN	VN

$$Precisão(\%) = \frac{VP}{VP + FP} x100$$

$$Sensibilidade(\%) = \frac{VP}{VP + FN} x100$$

$$Acurácia(\%) = \frac{VP + VN}{VP + VN + FP + FN} x100$$

4. RESULTADOS E DISCUSSÕES

A biblioteca de reconhecimento facial (**APÊNDICE A**) apresenta, em suas definições, que houve um valor de acurácia em **99,38%** apresentado em seus testes**.** No entanto, para comprovar a proximidade do valor desta acurácia, foram feitos vários testes com imagens em diferentes situações no contexto deste projeto.

4.1. RECONHECIMENTO DE ROSTOS

Para todas as imagens aqui apresentadas foram mantidos os padrões já estabelecidos na biblioteca, sendo de **0,5 para o valor mínimo de probabilidade para o classificador determinar que é um rosto.** Este valor possui baixa probabilidade pois precisa considerar diferentes posições, rotações e até a ausência de algum elemento do rosto como parte do olho fora da imagem, nariz de lado, dentre outras condições adversas.

Nos dados a seguir estão contabilizados todos os rostos coletados nas imagens apresentadas, gerando esta matriz de confusão e suas métricas.

	Quantidade	Detecção	Não detecção
Detecção	86	85	1
Não detecção	6	5	1
Total:	92		

Precisão	94,44%
Sensibilidade	98,84%
Acurácia	93,48%

Foi notado que a acurácia apresentada acima aumenta a cada novo teste que é feito com mais rostos detectados, aproximando cada vez mais do valor de acurácia apresentado pela biblioteca de reconhecimento facial implementada no projeto.

A seguir estão as imagens que foram processadas para gerar suas respectivas matrizes de confusão, as quais foram geradas através dos resultados pós processamento do classificador e identificação dos rostos.

	Quantidade	Detecção	Não detecção
Detecção	16	16	0
Não detecção	2	1	1
Total:	18		

Precisão	94,12%
Sensibilidade	100,00%
Acurácia	94,44%

	Quantidade	Detecção	Não detecção
Detecção	23	22	1
Não detecção	2	2	0
Total:	25		

Precisão	91,67%
Sensibilidade	100,00%
Acurácia	91,67%

	Quantidade	Detecção	Não detecção
Detecção	15	15	0
Não detecção	1	1	0
Total:	16		

Precisão	93,75%
Sensibilidade	100,00%
Acurácia	93,75%

	Quantidade	Detecção	Não detecção
Detecção	32	32	0
Não detecção	1	1	0
Total:	33		

Precisão	96,97%
Sensibilidade	100,00%
Acurácia	96,97%

4.2. IDENTIFICAÇÃO DA PESSOA

Será feita a identificação do rosto de uma pessoa em várias imagens em situações diferentes, o objetivo é apresentar o grau de precisão no processo de identificação do rosto correto. Neste caso, para obter o resultado é feito o cálculo da distância euclidiana em 128 coordenadas X e Y que determinam as características do rosto que são gerados na face (conforme a imagem abaixo do aluno escolhido).

Em uma escala de 10, o valor recomendado pela biblioteca como resultante da distância euclidiana é 0,6, **porém**, **ao fazer diversos testes foi comprovado o valor determinado 0,5 por não apresentar tantos falsos positivos quanto o valor anterior.**

Ao testar os comparados nas imagens abaixo, obteve-se os seguintes resultados (os resultados modificam e a acurácia aumenta à medida que vai fazendo mais testes):

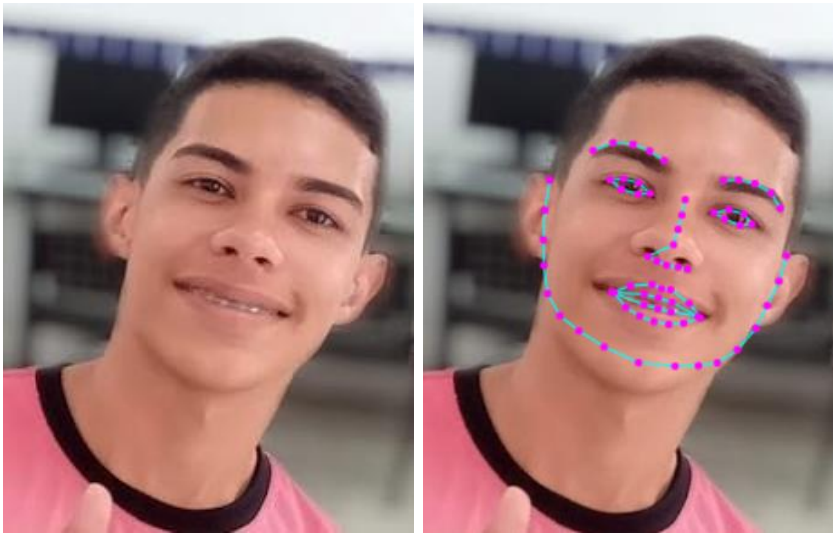
	Quantidade	Identificação correta	Incorreta
Identificação correta	6	6	0
Incorreta	1	1	0
Total:	7		

Precisão	85,71%
Sensibilidade	100,00%
Acurácia	85,71%

Assim como no processo de reconhecimento de rostos em imagens, neste caso de identificação da pessoa a acurácia também aumenta a cada novo teste que é feito. No entanto, verificando os casos apresentados pela matriz de confusão acima, notou-se que de 7 (sete) imagens de teste, 6 conseguiram identificar corretamente.

Para esta única ocorrência onde não foi possível a identificação da pessoa, ocorreu em uma imagem onde a pessoa escolhida estava com parte das mãos na frente do rosto, além da resolução da imagem não favorecer no reconhecimento de sua face.

O rosto escolhido para fazer os testes foi este:

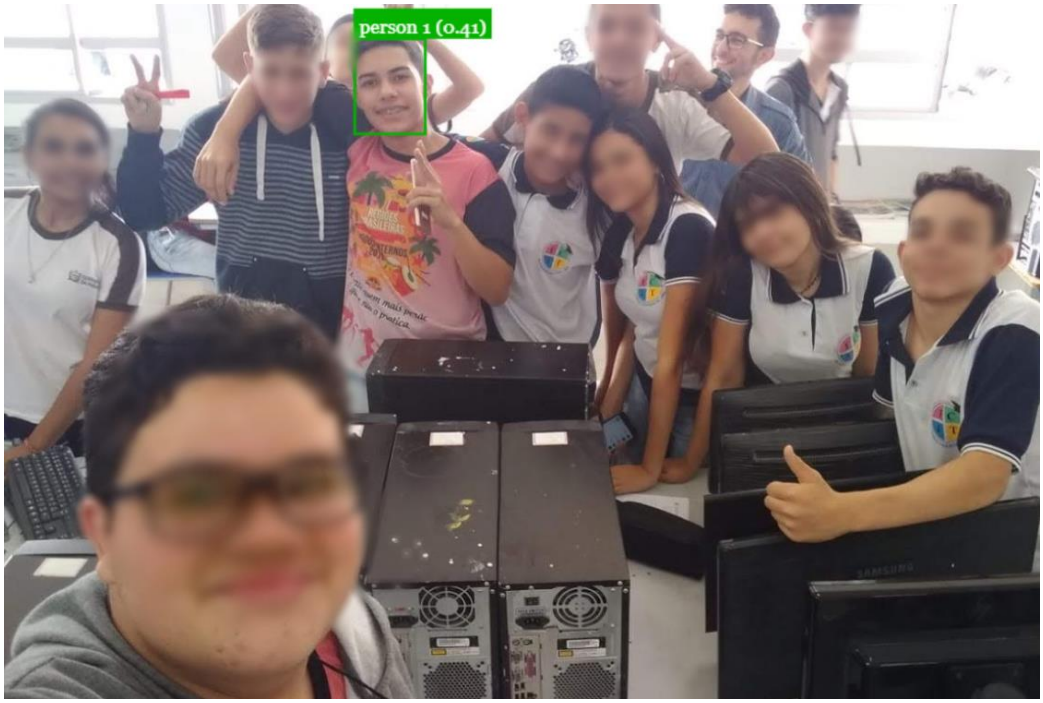


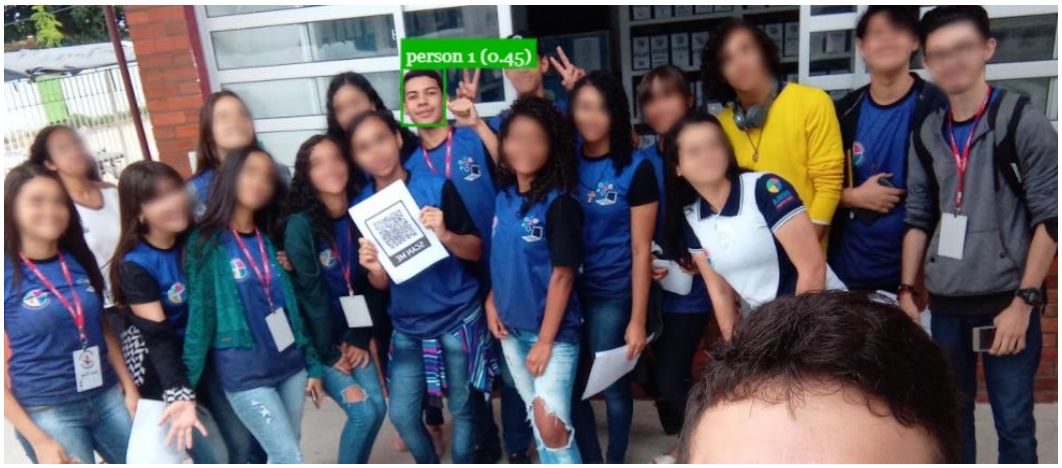
A seguir serão apresentadas as imagens de testes de verificação de face, caso seja positiva a identificação da pessoa o algoritmo irá marcar com um quadrado verde, junto do rótulo “person 1” e o resultante do cálculo da distância euclidiana para a imagem em questão, porém, quando não for possível o reconhecimento da face será marcado de vermelho manualmente pelo autor deste trabalho para representar onde o algoritmo deveria ter identificado.


PERIGO
person 1 (0.43)


person 1 (0.49)


person 1 (0.42)


person 1 (0.41)


person 1 (0.45)


person 1 (0.49)

5. CONCLUSÃO

Para este projeto foi construído um protótipo de software no qual o usuário (que neste caso pode ser um professor) faz o controle de suas turmas e alunos, podendo registrá-los como também registrar novas chamadas de suas aulas. Além disso, o recurso diferencial do protótipo é sua capacidade de realizar operações utilizando Redes Neurais Convolucionais para traçar os descritores (características) de uma face – no momento do cadastro do aluno – e identificar quais indivíduos estão presentes em uma determinada foto – durante o registro de uma nova chamada.

O propósito maior, além de criar esta ferramenta que proporciona uma maior facilidade para este contexto apresentado, é mostrar que os Smartphones atuais possuem sim capacidade de realizar operações complexas devido seu considerável poder de processamento, além de ter uma boa câmera capaz de tirar fotos com resolução suficiente para o algoritmo identificar padrões definidos pelos modelos. Vale ressaltar que este procedimento de reconhecimento e identificação dos rostos das pessoas é feito totalmente pelo próprio smartphone, ou seja, não depende de nenhum servidor externo para fazer tal processo.

Nos testes feitos neste projeto, foram consideradas as seguintes situações:

- **Reconhecimento de rostos:** para este caso foram testadas 4 imagens contendo um total de 92 rostos, dos quais 86 destes foram reconhecidos, dando uma acurária (razão de acerto entre os rostos reconhecidos e o total de rostos nas imagens) de **93,48%**, com uma precisão de **94,44%** e uma sensibilidade de **98,84%**, além disso, foi notado que à medida que mais imagens são acrescentadas e mais testes são feitos, estes valores – e principalmente a acurácia – aumentam seu valor;
- **Identificação da pessoa:** já neste caso, como foi feito o teste apenas para encontrar a pessoa certa através do seu rosto no meio de tantos outros, a contabilização foi feita a partir da sua quantidade de acertos, na qual foram testadas 7 imagens e destas apenas 6 delas identificaram corretamente a pessoa, apresentando uma acurária e precisão de 85,71%, e uma sensibilidade de 100%.

Ao analisar estes valores, notou-se que no processo de reconhecimento de rostos em uma imagem há uma maior tolerância quanto a variabilidade de posições, níveis de oclusão, rotações dos padrões que se deseja encontrar, até quando parte do rosto está coberto por algo em sua frente o algoritmo é capaz de reconhecer.

Já no caso do algoritmo responsável pela identificação da pessoa través do rosto mostrou uma tolerância não tão alta, pois no momento que parte do rosto da pessoa foi coberto por algo, o algoritmo não foi capaz de reconhecer aquela pessoa.

Por fim, o algoritmo de reconhecimento facial se mostrou bastante eficaz, pois apresentou uma taxa de acerto de mais de 90% dos casos de reconhecimento dos rostos e pouco mais de 85% da identificação dos rostos na imagem.

Este valor um pouco mais baixo na identificação dos rostos é algo aceitável pois o algoritmo é adaptável para aceitar mais de uma foto para a mesma pessoa, podendo acrescentar mais conjuntos de descritores da mesma, o que irá aumentar o seu grau de precisão e acurária para identificar a pessoa desejada.

Como se trata de um protótipo experimental que utiliza um modelo já treinado, esta mesma ferramenta poderá futuramente reconhecer padrões de outros tipos de modelos, podendo ser adaptado não só para reconhecimento facial, mas também para reconhecimento de outros elementos ou formas geométricas específicas, desde que já possua um modelo treinado, como por exemplo, treinar um modelo que represente padrões de lesões em tomografias para o algoritmo ser capaz de identificar tais padrões. Além disso, o mesmo algoritmo implementado no protótipo pode ser adaptado para outras ferramentas, como plugins de reconhecimento facial para navegadores, pois considerando o cenário da pandemia mundial neste ano de 2020, onde os professores precisam lecionar suas aulas remotamente, uma proposta futura bastante viável é um sistema capaz de realizar chamadas de forma automática, identificando o aluno através do reconhecimento facial e em certos períodos de tempo durante a aula, o próprio sistema refaz a chamada para verificar se os alunos ainda estão presentes.

REFERÊNCIAS

ALVES, Alanne V. da S. **Classificação de Lesões de Lábio a partir do Processamento de Imagens Térmicas**. UEPB: Campina Grande, 2018.

ALVES, Gabriel T. M. **Um Estudo das Técnicas de Obtenção de Forma a partir de Estéreo e Luz Estruturada para Engenharia**. Rio de Janeiro: PUC, 2005. Disponível em: <https://web.tecgraf.puc-rio.br/press/publication/Alves2005/Alves2005.pdf>. Acesso em: 22 nov. 2020.

ALVES, Gisely. **Entendendo Redes Convolucionais (CNNs)**. 2018. Disponível em: <https://medium.com/neuronio-br/entendendo-redes-convolucionais-cnns-d10359f21184>. Acesso em: 27 nov. 2020.

AZEVEDO, Lucas P. de. **Aplicação de Redes Neurais Artificiais no Processo de Classificação de Orquídeas do Gênero Cattleya**. IFMG: Sabará, 2016. Disponível em: <https://www.ifmg.edu.br/sabara/biblioteca/trabalhos-de-conclusao-de-curso/tcc-documentos/TCCLucasAzevedo.pdf>. Acesso em: 24 nov. 2020.

Data Science Academy. **17 casos de uso de Machine Learning.** 2018. Disponível em: <http://datascienceacademy.com.br/blog/17-casos-de-uso-de-machine-learning/>. Acesso em: 31 out. 2020.

Expert Academy. **Reconhecimento de Emoções com TensorFlow 2.0 e Python**. 2020. Disponível em: <https://iaexpert.academy/courses/reconhecimento-emocoes-tensorflow20-python/>. Acesso em: 23 nov. 2020.

FERREIRA, Elias da Costa. **Tensores e Aplicações**. Macapá: UFAM, 2013. Disponível em: <https://www2.unifap.br/matematica/files/2017/07/tcc-2013-elias-pereira-tensores-e-aplica%C3%A7oes.pdf>. Acesso em: 01 out. 2020.

FERREIRA, Paulo Afonso. **O avanço da tecnologia e as transformações na sociedade**. Agência de Notícias CNI, 2017. Disponível em: <https://noticias.portaldaindustria.com.br/artigos/paulo-afonso-ferreira/o-avanco-da-tecnologia-e-as-transformacoes-na-sociedade/>. Acesso em: 31 out. 2020.

FILHO, Cezar Bastos. **Reuso de Software**. Universidade Estadual de Londrina: Londrina, 2013. Disponível em: <http://www.uel.br/cce/dc/wp-content/uploads/VersaoPreliminarTCC-CezarBastos.pdf>. Acesso em: 25 nov. 2020.

GONZALEZ, Rafael C.; WOODS, Richard E. **Digital Image Processing**. 4ª ed, edição global. New York: Pearson, 2018.

GOODFELLOW, Ian; et al. **Deep Learning**. MIT Press, 2016. Disponível em: <https://www.deeplearningbook.org/>. Acesso em: 24 nov. 2020.

JÚNIOR, Audemar F. R. **Atenuação de Artefato Metálico em Imagem de Tomografia Computadorizada com uso de Deep Learning**. UEPB: Campina Grande, 2018.

MATTOS, Guilherme C. de. **PresentEye: Sistema de Controle de Presença por Reconhecimento Facial**. UFRJ: Rio de Janeiro, 2017. Disponível em: <https://bsi.uniriotec.br/wp-content/uploads/sites/31/2020/05/201712GuilhermeMattos.pdf>. Acesso em: 23 nov. 2020.

MESQUITA, Eduardo de M. **Desenvolvimento de Algoritmos via Visão Computacional para Identificação de Local e posterior Controle de Pouso de um Quadrirrotor Comercial**. Universidade de Brasília: Brasília, 2015. Disponível em: <https://bdm.unb.br/bitstream/10483/15294/1/2015_EduardodeMendoncaMesquita_tcc.pdf>. Acesso em: 24 nov. 2020.

MIRANDA, Bruno de S. **Algoritmos Clássicos de Aprendizado de Máquina Aplicados ao Problema do Reconhecimento de Imagens**. Alegrete: UNIPAMPA, 2011. Disponível em: <http://dspace.unipampa.edu.br/bitstream/riu/1547/1/Algoritmos%20cl%C3%A1ssicos%20de%20aprendizado%20de%20maquina%20aplicados%20ao%20problema%20do%20reconhecimento%20de%20imagens.pdf>. Acesso em: 22 nov. 2020.

MOVIO, Cleber Roberto. **Automação de chamada de sala de aula**. Faculdade Salesiana Dom Bosco de Piracicaba: Piracicaba, 2015. Disponível em: <https://fastformat.co/contests/submissions/5/pdf>. Acesso em: 23 nov. 2020.

PONTI, Moacir A.; COSTA, Gabriel B. Paranhos da. **Como funciona o Deep Learning**. SBC, 1ª ed. USP, 2017. Disponível em: <https://sites.icmc.usp.br/moacir/papers/Ponti_Costa_Como-funciona-o-Deep-Learning_2017.pdf>. Acesso em: 01 out. 2020.

ROSEMBERG, Carlos; et al. **Prototipação de software e design participativo: uma nova experiência do atlântico**. Instituto Atlântico: Porto Alegre, 2008. Disponível em: <https://www.researchgate.net/publication/220737394_Prototipacao_de_software_e_design_participativo_uma_experiencia_do_atlantico>. Acesso em: 25 nov. 2020.

RUFINO, Marcelo P. **Implementação de uma Rede Neural Artificial para Classificação de Sinais Mioelétricos**. UFPB: João Pessoa, 2017. Disponível em: <http://www.cear.ufpb.br/arquivos/cgee/TCC/TCC_-_Marcelo_Pereira_-_vers%C3%A3o_final.pdf>. Acesso em: 24 nov. 2020.

SHAI, Shalev Shwartz; SHAI, Ben David. **Understanding Machine Learning: From Theory to Algorithms**. Cambridge University Press, 2014. Disponível em: <https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf>. Acesso em: 31 out. 2020.

SHERER, João. **Melhores Celulares custo-benefício de 2020**. Guia dos Melhores, 2020. Disponível em: <https://guiadosmelhores.com.br/melhores-celulares-custo-beneficio/>. Acesso em: 24 nov. 2020.

SILVA, Marcelo C. R. da. **Aprendizagem de Máquina em Apoio a Diagnóstico em Ortopedia**. PUC: Campinas, 2016. Disponível em: <http://tede.bibliotecadigital.puc-campinas.edu.br:8080/jspui/bitstream/tede/902/2/Marcelo%20Cicero%20Ribeiro%20da%20Silva.pdf>. Acesso em: 23 nov. 2020.

SIMÃO, Bárbara; FRAGOSO, Nathalie; ROBERTO, Enrico. **Reconhecimento Facial e o Setor Privado: Guia para a adoção de boas práticas**. InternetLab/IDEC, São Paulo, 2020. Disponível em: <https://idec.org.br/sites/default/files/reconhecimento_facial_diagramacao_digital_2.pdf> Acesso em: 23 nov. 2020.

TEIXEIRA, Tiago. **Controle de Fluxo de Pessoas Usando RFId**. IFSC: São José, 2011. Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/images/f/fa/TCC_TiagoTeixeira.pdf>. Acesso em: 23 nov. 2020.

TUTORIALSPOINT. **Ionic Tutorial**. 2018. Disponível em: <https://www.tutorialspoint.com/ionic/ionic_tutorial.pdf>. Acesso em: 26 nov. 2020.

ANEXOS

ANEXO A – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO PARA PESQUISA

UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS III – CAMPINA GRANDE
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIAS DA COMPUTAÇÃO

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO PARA PESQUISA

Eu Carlos Antonio Jose da Silva, concordo na utilização do rosto de Valdenia Vieira Ferreira, presente no tópico de **Resultados e Discussões** deste documento, para fins de coleta de dados no trabalho de título **RECONHECIMENTO FACIAL: UTILIZANDO UM SISTEMA DE CHAMADAS AUTOMÁTICO PELO SMARTPHONE**, tendo como responsável o aluno Aleksandro da Costa Fabrício, do curso de Ciências da Computação, da Universidade Estadual da Paraíba, Campina Grande – PB, que pode ser contactado pelo telefone **(83) 9.9413-4827** e e-mail **alekfabricio@gmail.com**. Além disso, tenho ciência de que a utilização do rosto apenas para apresentação dos resultados neste trabalho e durante sua defesa, como recurso necessário para a conclusão do referente Trabalho de Conclusão de Curso. Além disso, tenho ciência de que posso revogar minha autorização quando desejar.

Assinatura do Responsável

Campina Grande – PB, 30 de 11 de 2020

ANEXO B – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO DO GESTOR ESCOLAR

SECRETARIA DE EDUCAÇÃO E CULTURA
2ª REGIÃO DE ENSINO
ECIT ALFREDO PESSOA DE LIMA
DEC. 5.147 DE 07/11/1970
RUA LUIZ FERREIRA [illegible] MELO, SN
SOLÂNEA - PB

Somos todos PARAÍBA Governo do Estado | GOVERNO digital

SECRETARIA DE ESTADO DA EDUCAÇÃO E DA CIÊNCIA E TECNOLOGIA
ECIT ALFREDO PESSOA DE LIMA
SOLÂNEA (PB)

CARTA DE ANUÊNCIA

Declaramos para os devidos fins, que aceitamos o pesquisador **Aleksandro da Costa Fabrício** a desenvolver o seu projeto de pesquisa **RECONHECIMENTO FACIAL: UTILIZANDO UM SISTEMA DE CHAMADAS AUTOMÁTICO PELO SMARTPHONE**, que está sob à orientação do Prof. Dr. Robson Pequeno de Souza, com o objetivo de "desenvolver um protótipo para smartphone que possua execução de chamada de presença por meio de reconhecimento facial, independente de internet ou qualquer conexão para identificação do aluno".

Esta autorização está condicionada ao cumprimento do pesquisador aos requisitos da Resolução 466/12 CNS e seus complementares, comprometendo-se o mesmo a utilizar os dados dos sujeitos da pesquisa, exclusivamente para fins científico, mantendo o sigilo e garantindo a não utilização das informações em prejuízo das pessoas e/ou das comunidades.

Solânea – PB, em 31/11/2020

Assinatura do Gestor

Valdecl Alves Diniz
Gestor Escolar
Mat. 145552-4 Aut. 8702

APÊNDICES

APÊNDICE A – BIBLIOTECA FACE-API.JS

A biblioteca **face-api.js**, desenvolvida por **Vicent Mühler**[2], é uma API desenvolvida para detecção e reconhecimento de faces em aplicações com suporte para javascript, tendo como base a API do **tensorflow.js**. Essa API pode ser encontrada em seu repositório https://github.com/justadudewhohacks/face-api.js.git.

A face-api.js conta com diversos algoritmos que utilizam modelos de detecção de rosto através de redes neurais, onde computa as localizações de cada face em uma imagem e retorna as cas delimitadoras, junto com sua probabilidade para cada face. Dentre estes modelos estão:

- **SSD Mobilenet V1** (utilizado neste projeto): Este modelo implementa um SSD (Single Shot Multibox Detector), baseado em Mobile Net V1. O tamanho do modelo quantizado é cerca de 5,4 MB (ssd_mobilenetv1_model), o qual foi treinado a partir do dataset WIDERFACE;
- **Tiny Face Detector:** Se trata de um detector de rosto em tempo real de alto desempenho, sendo mais rápido, menor e consome menos recursos em comparação ao detector SSD Mobilenet V1, porém possui uma menor precisão na detecção de rostos. Este modelo é destinado principalmente para soluções com dispositivos de recursos limitados. Seu tamanho é quantizado em 190 KB (tiny_face_detector_model). Este modelo foi resultante de um treinamento através de um dataset com aproximadamente 14 mil imagens, substituindo convoluções regulares por convoluções separáveis em profundidade;
- **MTCNN (Multi-task Cascaded Convolutional Neural Networks):** É um modelo apresentado como experimental por existir outros detectores de rosto com melhor desempenho. Se trata de um detector facial alternativo ao SSD Mobilenet V1 e Tini Face Detector, com mais opções de configuração, como por exemplo, podendo ser capaz de detectar uma ampla gama de tamanhos de faces. O MTCNN é um RNC em cascata de 3 estágios, que retorna simultaneamente 5 pontos de referência de rosto junto com as caixas delimitadoras e pontuações para cada rosto. Além disso, seu tamanho é de 2 MB;
- **68 Point Face Landmark Detection Models:** Este pacote implementa um detector de pontos de referência facial de 68 pontos, sendo leve e rápido, mas preciso. Seu modelo padrão tem apenas 350 KB (face_landmark_68_model) e sua versão reduzia tem apenas 80 KB (face_landmark_68_tiny_model). Ambos os modelos empregam as ideias de

[2] Twitter do Vicent Mühler: https://twitter.com/justadudewhohax

convoluções separáveis em profundidade, totalmente conectados. Os modelos foram treinados com um dataset de aproximadamente 35 mil imagens;

- **Face Recognition Model:** Para o reconhecimento do rosto, uma arquitetura semelhante ao ResNet-34 é implementada para calcular um descritor de rosto (um vetor de recursos com 128 valores) a partir de qualquer imagem facial, que é usada para descrever as características do rosto de uma. O modelo não se limita ao conjunto de rostos usados para treinamento, ou seja, pode identificar qualquer rosto de qualquer pessoa. Pode ser calculada a similaridade de duas faces arbitrárias comparando seus descritores de face, por exemplo, calculando a distância euclidiana. Os pesos deste modelo foram treinados até atingir uma precisão de 99,38% no benchmark LFW (Labeled Faces in the Wild) para reconhecimento facial. Possui um tamanho quantizado de aproximadamente 6,2 M (face_recognition_model);
- **Face Expression Recognition Model:** Se trata de um modelo de reconhecimento de expressão facial leve, rápido e com precisão razoável. Seu tamanho é quantizado em 310 KB e emprega convoluções separáveis em profundidade e densamente conectados. Foi treinado em uma variedade de imagens de conjuntos de dados disponíveis publicamente, bem como imagens extraídas da web.